# 一 image classification

1. 训练

|  |
| --- |
| python train\_image\_classifier.py --train\_dir=satellite/train\_dir --dataset\_name=satellite --dataset\_split\_name=train --dataset\_dir=satellite/data --model\_name=inception\_v3 --checkpoint\_path=satellite/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 |

|  |
| --- |
| python eval\_image\_classifier.py --checkpoint\_path=satellite/train\_dir --eval\_dir=satellite/eval\_dir --dataset\_name=satellite --dataset\_split\_name=validation --dataset\_dir=satellite/data --model\_name=inception\_v3 |

# 二 object detection

第一步, 准备数据集.

下载:

|  |
| --- |
| wget http://host.robots.ox.ac.uk/pascal/VOC/voc2012/VOCtrainval\_11-May-2012.tar |

用voc2012.整理成tfrecord模型,以便tf可以识别使用.

|  |
| --- |
| python create\_pascal\_tf\_record.py --data\_dir voc/VOCdevkit/ --year=VOC2012 --set=train --output\_path=voc/pascal\_train.record  python create\_pascal\_tf\_record.py --data\_dir voc/VOCdevkit/ --year=VOC2012 --set=val --output\_path=voc/pascal\_val.record |

拷贝label, 这个是human 可读的label描述标签.

|  |
| --- |
| cp data/pascal\_label\_map.pbtxt voc/ |

在voc中新建 pretrain, 构建pretrain模型.

|  |
| --- |
| wget http://download.tensorflow.org/models/object\_detection/faster\_rcnn\_inception\_resnet\_v2\_atrous\_coco\_11\_06\_2017.tar.gz |

第二步,训练

|  |
| --- |
| **export PYTHONPATH=$PYTHONPATH:`pwd`:`pwd`/slim**  chmod a+x ../research/bin/protoc  ../research/bin/protoc object\_detection/protos/\*.proto --python\_out=.  /home/julyedu\_433249/work/tf\_base/research/bin/protoc object\_detection/protos/\*.proto --python\_out=. |

|  |
| --- |
| python train.py --train\_dir voc/train\_dir/ --pipeline\_config\_path voc/voc.config |

第三步, 保存模型.

|  |
| --- |
| python3 export\_inference\_graph.py --input\_type image\_tensor --pipeline\_config\_path voc/voc.config --trained\_checkpoint\_prefix voc/train\_dir/model.ckpt-48 --output\_directory voc/export |

导出的模型是voc/export/frozen\_inference\_graph.pb 文件。

三 tensorflow训练过程中对变量值的打印的方法

目的是在训练过程中打印变量的值,不仅仅是打印tensor的情况.

一般使用print打印某个tensor时候.只能打印该tensor确定的shape信息,而且有些shape的维度和输入的图像有关系,会显示出?号的不确定含义.

如何能够在训练过程中实时打印某些变量的值呢?

第一步, 在根目录下构建一个文件tfprint.py.

|  |
| --- |
| # tf print using var  import tensorflow as tf  **tfp\_similarity\_matrix = tf.placeholder(tf.float32)**  # 需要定义一个tensor. |

第二步,在train函数中.

|  |
| --- |
| --- a/dl\_object\_detection/object\_detection/trainer.py  +++ b/dl\_object\_detection/object\_detection/trainer.py  @@ -31,7 +31,7 @@ from object\_detection.core import standard\_fields as fields  from object\_detection.utils import ops as util\_ops  from object\_detection.utils import variables\_helper  from deployment import model\_deploy  -  **+from object\_detection import tfprint**  slim = tf.contrib.slim  @@ -297,7 +297,7 @@ def train(create\_tensor\_dict\_fn, create\_model\_fn, train\_config, master, task,  slim.learning.train(  - train\_tensor,  **+ [train\_tensor,tfprint.tfp\_similarity\_matrix], # 添加训练时要操作的tensor对象.**  logdir=train\_dir,  master=master,  is\_chief=is\_chief, |

第三步,在待打印的变量的文件中.添加如下.

|  |
| --- |
| from object\_detection import tfprint  …  def \_match(self, similarity\_matrix):    **tfprint.tfp\_similarity\_matrix = tf.Print(similarity\_matrix,["argmax\_matcher's input : similarity\_matrix\n",similarity\_matrix],message="[trainning info]")** |

summarize=

是显示多少个单位的数据.默认是3个.

**转换成tflite的**

|  |
| --- |
| toco --graph\_def\_file=voc/export/frozen\_inference\_graph.pb --output\_file=voc/export/frozen\_inference\_graph.lite --input\_format=TENSORFLOW\_GRAPHDEF --output\_format=TFLITE --input\_shape=1,299,299,3 --input\_array=image\_tensor --output\_array=detection\_boxes,detection\_scores,detection\_classes,num\_detections --inference\_type=FLOAT --input\_data\_type=FLOAT  freeze\_graph --input\_graph=/tmp/mobilenet\_v1\_224.pb --input\_checkpoint=/tmp/checkpoints/mobilenet-10202.ckpt --input\_binary=true  --output\_graph=/tmp/frozen\_mobilenet\_v1\_224.pb  --output\_node\_names=MobileNet/Predictions/Reshape\_1            python export\_inference\_graph.py --input\_type image\_tensor --pipeline\_config\_path voc/voc.config --trained\_checkpoint\_prefix voc/train\_dir/model.ckpt-1582 --output\_directory voc/export  python export\_inference\_graph.py --input\_type image\_tensor --pipeline\_config\_path voc/mob.config --trained\_checkpoint\_prefix voc/mob\_train\_dir/model.ckpt-0 --output\_directory voc/mob\_export |

网络的输入是: image\_tensor

输出是:

[\_export\_inference\_graph] output\_node\_names:%s **detection\_boxes,detection\_scores,detection\_classes,num\_detections**

Ssd

|  |
| --- |
| # for model in \  ssd\_mobilenet\_v1\_coco\_11\_06\_2017 \  ssd\_inception\_v2\_coco\_11\_06\_2017 \  rfcn\_resnet101\_coco\_11\_06\_2017 \  faster\_rcnn\_resnet101\_coco\_11\_06\_2017 \  faster\_rcnn\_inception\_resnet\_v2\_atrous\_coco\_11\_06\_2017  do \  curl -OL http://download.tensorflow.org/models/object\_detection/$model.tar.gz  tar -xzf $model.tar.gz $model/frozen\_inference\_graph.pb  cp -a $model /opt/graph\_def/ |

curl -OL <http://download.tensorflow.org/models/object_detection/ssd_mobilenet_v1_coco_11_06_2017.tar.gz>

tar -xzf ssd\_mobilenet\_v1\_coco\_11\_06\_2017.tar.gz

python train.py --train\_dir voc/mob\_train\_dir/ --pipeline\_config\_path voc/mob.config

**objectdetection 本身就有一个ssd转tflite的工具.**

python export\_tflite\_ssd\_graph.py --pipeline\_config\_path=voc/mob.config --trained\_checkpoint\_prefix=voc/mob\_train\_dir/model.ckpt-0 --output\_directory=voc/mob\_export --add\_postprocessing\_op=true

python model\_main.py --model\_dir voc/mob\_train\_dir/ --pipeline\_config\_path voc/mob.config

python object\_detection/model\_main.py --model\_dir object\_detection/voc/train\_dir/ --pipeline\_config\_path object\_detection/voc/voc.config

有一个legecy的 train

python legacy/train.py --train\_dir vvv/trainout/ --pipeline\_config\_path vvv/vv.config

detection\_boxes

toco --graph\_def\_file=voc/export/frozen\_inference\_graph.pb --output\_file=voc/export/frozen\_inference\_graph.lite --input\_format=TENSORFLOW\_GRAPHDEF --output\_format=TFLITE --input\_shape=1,299,299,3 --input\_array=image\_tensor --output\_array= box\_encodings --inference\_type=FLOAT --input\_data\_type=FLOAT

可用的toco

|  |
| --- |
| toco --graph\_def\_file=vvv/export/tflite\_graph.pb--output\_file=vvv/litedir/li.lite --input\_format=TENSORFLOW\_GRAPHDEF --output\_format=TFLITE --input\_shape=1,299,299,3 --input\_array=image\_tensor --output\_array='TFLite\_Detection\_PostProcess','TFLite\_Detection\_PostProcess:1','TFLite\_Detection\_PostProcess:2','TFLite\_Detection\_PostProcess:3' --inference\_type=FLOAT --input\_data\_type=FLOAT --allow\_custom\_ops  ~~toco --graph\_def\_file=vvv/export/tflite\_graph.pb--output\_file=vvv/litedir/lili.lite -- --input\_shape=1,300,300,3 --input\_array= normalized\_input\_image\_tensor --output\_array='TFLite\_Detection\_PostProcess','TFLite\_Detection\_PostProcess:1','TFLite\_Detection\_PostProcess:2','TFLite\_Detection\_PostProcess:3' --inference\_type=FLOAT --allow\_custom\_ops~~  toco --graph\_def\_file=vvv/export/tflite\_graph.pb --output\_file=vvv/litedir/lili.lite --input\_shapes=1,**300,300**,3 --input\_arrays=**normalized\_input\_image\_tensor** --output\_arrays=**'TFLite\_Detection\_PostProcess','TFLite\_Detection\_PostProcess:1','TFLite\_Detection\_PostProcess:2','TFLite\_Detection\_PostProcess:3'** --inference\_type=FLOAT --allow\_custom\_ops  toco --input\_file=$OUTPUT\_DIR/tflite\_graph.pb --output\_file=$OUTPUT\_DIR/detect.tflite \  --input\_shapes=1,300,300,3 \  --input\_arrays=normalized\_input\_image\_tensor \  --output\_arrays='TFLite\_Detection\_PostProcess','TFLite\_Detection\_PostProcess:1','TFLite\_Detection\_PostProcess:2','TFLite\_Detection\_PostProcess:3' \  --inference\_type=FLOAT \  --allow\_custom\_ops |

参考:

<https://jefby.github.io/2018/08/20/%E5%B0%86mobilenet-ssd-tensorflow-pb%E8%BD%AC%E6%8D%A2%E4%B8%BAtflite%E7%9A%84%E8%AF%A6%E7%BB%86%E6%AD%A5%E9%AA%A4/>

关键是output\_array是怎么找到的?