**第九周作业 目标检测（本地）**

利用slim框架和object\_detection框架，做一个物体检测的模型。具体代码为：https://github.com/Zoushuang86/quiz\_w8

**一、数据集选定**

选择自己的数据集，5个分类（记得修改模型配置文件的分类参数改为5），155张图片，数据不多，训练集和验证集比是7：3。

**二、模型选定**

使用最新的ssd模型和mobilenet模型对物体进行分类和检测，代码参考ssd\_mobilenet\_v1\_coco.config。

**三、文件结构准备**

根据官方文档的推荐的文件结构进行建立。在models/research/目录下分别建立models和data文件夹，分别存放模型相关文件和数据相关文件。

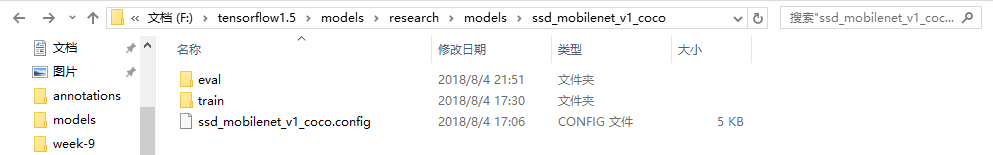


图1 models/ssd\_mobilnet目录结构

其中train文件夹存放训练模型的checkpoint文件，val文件夹存放验证模型的数据，ssd\_mobilnet\_v1\_coco.config是训练COCO数据集的模型配置文件。

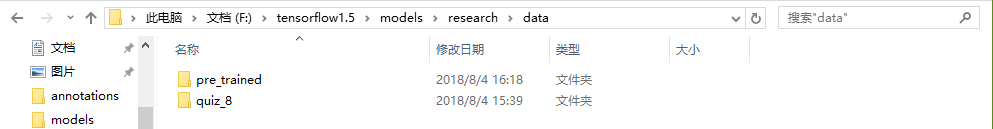


图2 models/research/data目录结构



图3 pre\_trained文件夹存放的checkpoint文件



图4 quiz\_8文件夹存放的训练数据

其中pre\_trained文件夹存放初始化模型的checkpoint文件，quiz\_8文件夹存放训练的数据：annotations文件夹存放了xml类型的注释每张图片的文件，images文件存放了jpg类型的图片，labels\_items.txt是将标签类型和数字映射的文件。

**四、准备数据集**

将数据集编码成tensorflow的record文件。以models官方代码models/research/object\_detection/dataset\_tools/create\_pet\_tf\_record.py为基础进行修改，生成create\_data.py，注意是r1.5版本的。

代码修改注意要点：

1. 本作业的数据集没有mask，所以代码中所有关于faces\_only和mask的代码都要删除。
2. 图片的class需要直接从xml文件的name字段中获取，所以get\_class\_name\_from\_filename函数相关的内容都要删除。
3. 官方教程中提到的label\_map\_path文件，请直接使用数据集中的labels\_items.txt文件，其内容的格式与pet\_label\_map.pbtxt文件相同，但是只有5项。
4. 本数据集xml文件中，filename项有扩展名，这里需要注意。

执行代码：

# From the tensorflow1.5/models/research/ directory

source activate python3

python object\_detection/dataset\_tools/create\_data.py \

--label\_map\_path=./data/quiz\_8/labels\_items.pbtxt \

--data\_dir=./data/quiz\_8/ \

--output\_dir=./data/quiz\_8/

在output\_dir下生成pet\_train.record和pet\_val.record两个文件，如下图：



图5 生成的两个record数据文件

**五、编辑pipline.config文件**

以models/research/object\_detection/samples/configs/ssd\_mobilenet\_v1\_coco.config为基础进行修改(ssd\_mobilenet\_v1\_pet.config对应模型初始化的checkpoint已经找不到)。本文档所在仓库也提供了这个文件。 这个文件里面放了训练和验证过程的所有参数的配置，包括各种路径，各种训练参数（学习率，decay，batch\_size等）。有这个文件，命令行上面可以少写很多参数，避免命令行内容太多。

注意要点：

1. num\_classes， 原文件里面为37,这里的数据集为5。
2. num\_examples， 这个是验证集中有多少数量的图片，请根据图片数量和数据准备脚本中的生成规则自行计算。
3. PATH\_TO\_BE\_CONFIGURED，这个是原文件中预留的字段，一共5个，分别包含预训练模型的位置，训练集数据和label\_map文件位置，验证集数据和label\_map文件位置。这个字段需要将数据以及配置文件等上传到tinymind之后才能确定路径的具体位置。
4. num\_steps，这个是训练多少step，后面的训练启动脚本会用到这个字段，直接将原始的200000改成0。注意不要添加或者删除空格等，后面的训练启动脚本使用sed对这个字段进行检测替换，如果改的有问题会影像训练启动脚本的执行。不通过run.sh本地运行需要将这个数字改成一个合适的step数，改成0的话会有问题。
5. max\_evals，这个是验证每次跑几轮，这里直接改成1即可，即每个训练验证循环只跑一次验证。
6. eval\_input\_reader 里面的shuffle， 这个是跟eval步骤的数据reader有关，如果不使用GPU进行训练的话，这里需要从false改成true，不然会导致错误，详细内容参阅 <https://github.com/tensorflow/models/issues/1936>

训练和验证过程次数相关的参数，后面在训练启动脚本中会自动进行处理，这里不需要过多关注，但是实际使用的时候，需要对这些参数进行合适的设置，比如num\_steps参数，后面的训练启动脚本中，每轮运行100个step，同时根据数据集图片总数all\_images\_count和batch\_size的大小，可以计算出epoch的数量，最后输出模型的质量与epoch的数量密切相关。epoch=num\_step\*batch\_size/all\_images\_count。具体的计算留给学员自己进行。

config文件需要跟代码一起上传，运行的时候会先被复制到output文件夹里面。

**训练模型代码：**

# From the tensorflow/models/research/ directory

PATH\_TO\_YOUR\_PIPELINE\_CONFIG=./models/ssd\_mobilenet\_v1\_coco/ssd\_mobilenet\_v1\_coco.config

PATH\_TO\_TRAIN\_DIR=./models/ssd\_mobilenet\_v1\_coco/train/

python object\_detection/train.py \

--logtostderr \

--pipeline\_config\_path=${PATH\_TO\_YOUR\_PIPELINE\_CONFIG} \

--train\_dir=${PATH\_TO\_TRAIN\_DIR}

**验证代码：**

# From the tensorflow/models/research/ directory

PATH\_TO\_YOUR\_PIPELINE\_CONFIG=./models/ssd\_mobilenet\_v1\_coco/ssd\_mobilenet\_v1\_coco.config

PATH\_TO\_TRAIN\_DIR=./models/ssd\_mobilenet\_v1\_coco/train/

PATH\_TO\_EVAL\_DIR=./models/ssd\_mobilenet\_v1\_coco/eval/

python object\_detection/eval.py \

--logtostderr \

--pipeline\_config\_path=${PATH\_TO\_YOUR\_PIPELINE\_CONFIG} \

--checkpoint\_dir=${PATH\_TO\_TRAIN\_DIR} \

--eval\_dir=${PATH\_TO\_EVAL\_DIR}

**六、在Tensorboard查看结果**

启动tensorboard代码：

tensorboard --logdir=${PATH\_TO\_EVAL\_DIR}

从结果中挑出两个具有代表性的验证结果：图6虽然图片模糊了，但是大部分物体都识别了出来，但是还是有水桶没有识别出来。图7也有部分物体没有识别出来，最大的问题在于同一个物体识别了两次，说明模型对边框回归还没有训练好。

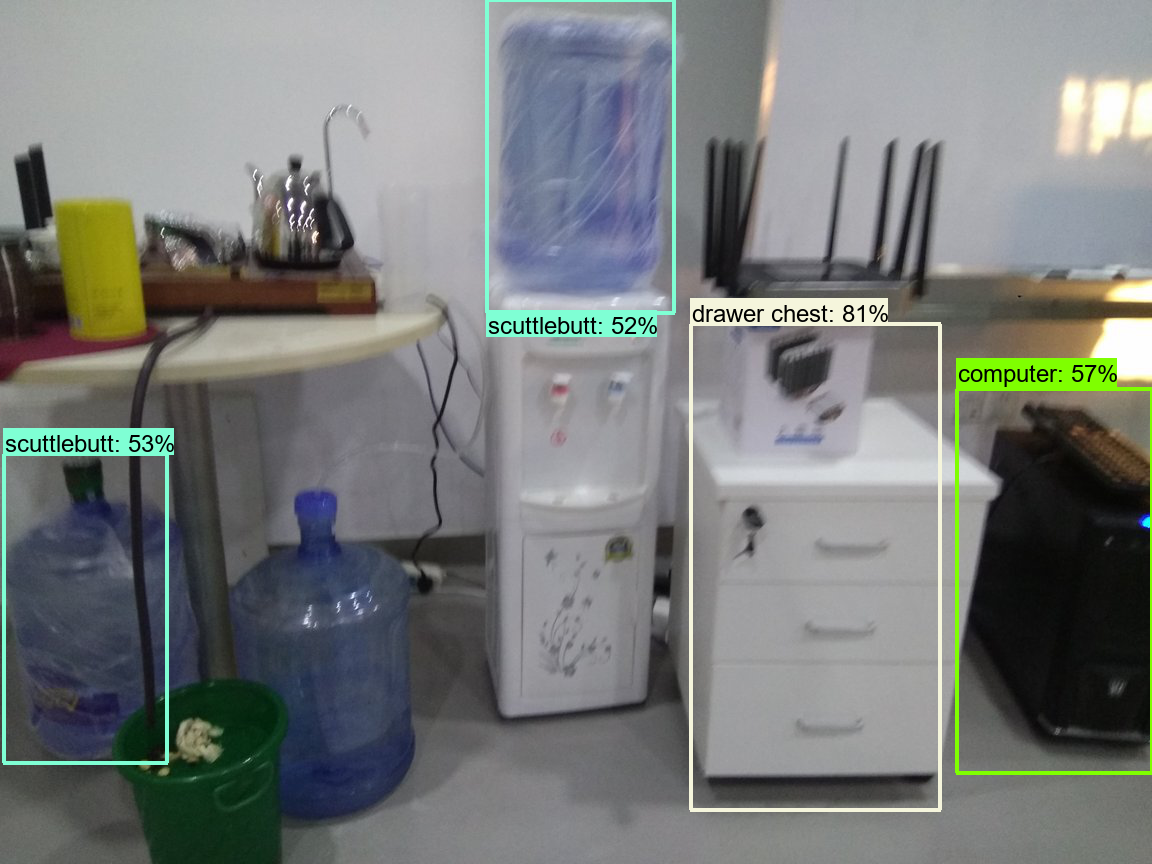


图6 验证图片1

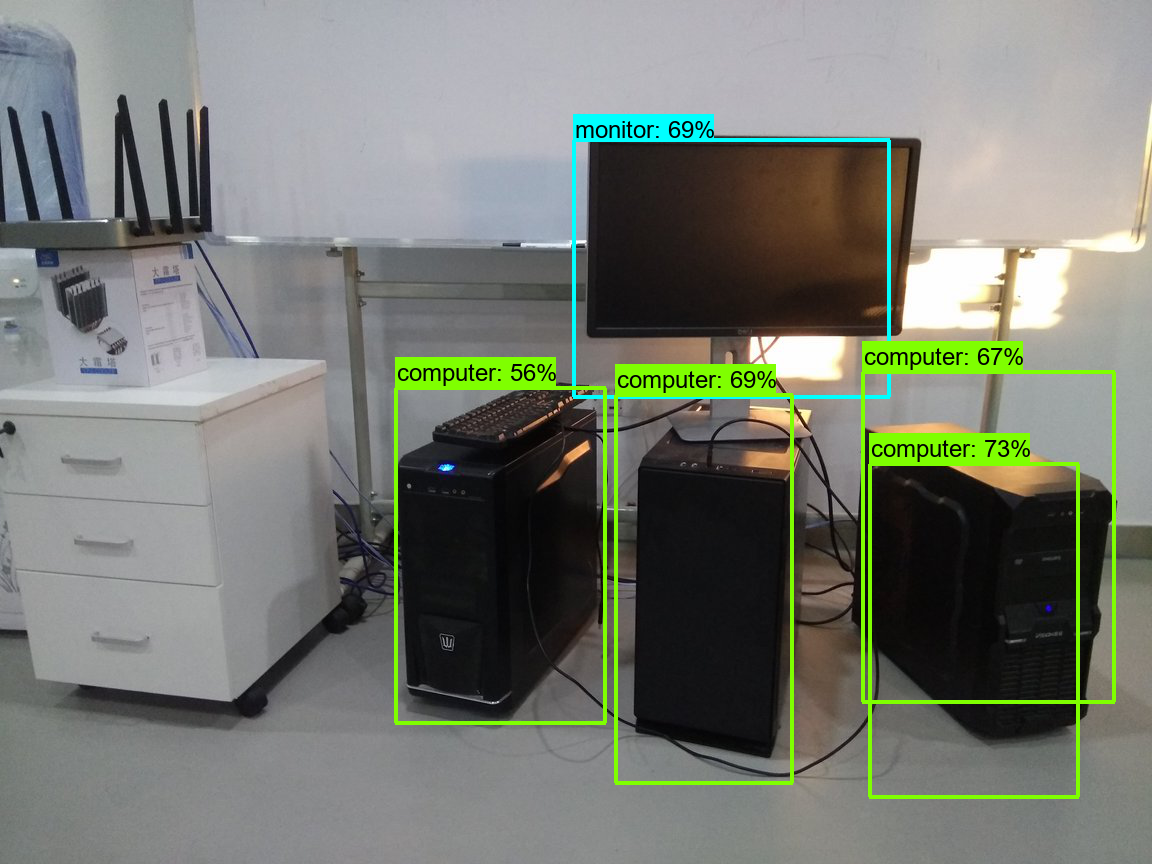


图7 验证图片2