

作者: Farry 方禹 日期: 2018-12-28

该教程适用于 ubuntu 16.04 操作系统

python 3.5

tensorflow 1.4.0

opencv 3.2.0

代码地址: <https://github.com/MaybeShewill-CV/lanenet-lane-detection>

1. 目录树

在 lanenet 文件夹中目录结构如下:

lanenet

-.idea

--vcs.xml

-config

 --__init__.py

 --global_config.py

-data

 --training

 ---binary

 ---images

 ---labels

 ---test

 ---testlabel

 ---train.txt

 ---val.txt

 ---test.txt

-model

-data_provider

 --__init__.py

 --data_processor.py

 --lanenet_data_processor.py

 --lanenet_hnet_data_processor.py

-encoder_decder_model

 --__init__.py

 --cnn_basenet.py

 --dense_encoder.py

 --fcn_decoder.py

 --vgg_encoder.py

-lanenet_model

 --__init__.py

 --lanenet_binary_segmentation.py

 --lanenet_cluster.py

 --lanenet_discriminative_loss.py

 --lanenet_hnet_loss.py

 --lanenet_hnet_model.py

```
--lanenet_instance_segmentation.py
--lanenet_merge_model.py
--lanenet_postprocess.py
-tboard
  --tusimple_lanenet
    ---vgg
-tools
  --test_lanenet.py
  --train_lanenet.py
-_config.yml
```

2.环境配置

根据 requirements.txt 中的要求，按照需要更新和安装环境（如果代码可以运行建议不更新环境）

如果需要更新环境，可以在 terminal 中输入以下指令：

```
pip3 install -r requirements.txt
```

3.准备训练图片和标签图片

该模型训练需要三类图片分别是原始图片、二值化分割图像（255 表示车道区域，0 表示其他的）和实例图像（标签图像）。

原始图像放在 data 目录下的 images 目录下

二值化分割图像放在 data 目录下的 binary 目录下

实例图像（标签图像）放在 data 目录下的 labels 目录下

然后需要在 data 目录下放 train.txt 和 val.txt 两个文件夹，两个文件中每一行存放的是原始图像存放的路径（包含图像名称）、二值化图像存放的路径（包含图像名称）

和实例图像（标签图像）存放的路径（包含图像名称）。这里顺序要与教程中的一致。这两个文件所指向的图片分别是训练集和测试集。

举 例 ： /XXX/lanenet/data/training/images/170927_063811892_Camera_5.jpg
/XXX/lanenet/data/training/binary/170927_063811892_Camera_5_bin.png
/XXX/lanenet/data/training/labels/170927_063811892_Camera_5_bin.png

两个 txt 文件中内容的格式是一样的。

4.global_config.py

该文件在 config 文件夹下，用来设置模型的相关参数其中需要注意的是

#分类的类别个数

```
__C.TRAIN.CLASSES_NUMS = 2
```

#训练时输入图片的高度

```
__C.TRAIN.IMG_HEIGHT = 256
```

#训练时输入图片的宽度

```
__C.TRAIN.IMG_WIDTH = 512
```

5.data_provider

该文件夹下存放的都是处理模型的输入数据的脚本

其中 lanenet_data_processor.py 为训练模型提供输入图片数据

6.encoder_decoder_model

该文件夹中存放的是用于编码和解码的模型

这里使用的是 `vgg_encoder.py` 和 `fcn_decoder.py`

未测试:

更换编码和解码的模型

7.lanenet_model

该文件夹下存放的是 `lanenet` 模型的重要操作

`train.py` 中主要用到的是 `lanenet_merge_model`

8.tboard

该文件夹用来存放模型训练时产生的 `event` 记录

9.tools

该文件夹下存放的是 `train_lanenet.py` 和 `test_lanenet.py` 两个文件分别用于训练模型和测试模型

10.训练

准备好训练数据以后, 在 `terminal` (终端) 中通过指令跳转到 `lanenet` 目录下

举例: `cd /XXX/lanenet/`

然后输入一下命令开始训练

```
python tools/train_lanenet.py --net vgg --dataset_dir data/training_data_example/
```

`--net` 指定模型的骨干网络

`--dataset_dir` 指定训练数据的目录(该目录为 `train.txt` 所在的目录)

如果训练的过程中出现了中断, 需要接着训练, 可以输入

```
python tools/train_lanenet.py --net vgg --dataset_dir data/training_data_example/
```

```
--weights_path path/to/your/last/checkpoint
```

`--weights_path` 指定之前训练中断时权重文件存放的目录

训练结束后, 训练得到的模型的权重和参数文件在 `model` 目录下的 `tusimple_lanenet` 文件夹中

11.测试

准备测试数据:

测试数据是原始图片, 放在 `data` 目录下的 `training` 目录下的 `testimage` 中 (可以自定义)

准备权重文件:

将训练好的权重文件, 包括:

`checkpoint,lanenet_model.pb,tusimple_lanenet_vgg_2018-10-19-13-33-56.ckpt-2000`

`00.data-00000-of-00001,tusimple_lanenet_vgg_2018-10-19-13-33-56.ckpt-200000.in`

`dex,tusimple_lanenet_vgg_2018-10-19-13-33-56.ckpt-200000.meta`

放在 `model` 文件夹中

测试数据准备完成后，打开 terminal（终端）输入以下命令跳转到 lanenet 目录下：
`cd /XXX/lanenet/`

然后输入如下命令开始用单张图片测试模型：

```
python tools/test_lanenet.py --is_batch False --batch_size 1 --weights_path  
path/to/your/model_weights_file --image_path data/tusimple_test_image/0.jpg
```

--is_batch 是否是批量处理（True 表示是批量处理，False 表示不是批量处理，默认为 True）
--batch_size 批量处理一次读取的图片数量
--weight_path 模型的权重文件所在的目录
--image_path 测试数据的原始图片

输入如下命令开始用批量图片测试模型：

```
python tools/test_lanenet.py --is_batch True --batch_size 2 --save_dir  
data/tusimple_test_image/ret --weights_path path/to/your/model_weights_file  
--image_path data/tusimple_test_image/
```

--save_dir 是存放测试结果的地方（测试结果为划分车道线后的图片）

