一、环境

Ubuntu18.04
Python3.6.8(不要低于 Python3.6.5)
Cuda10.0+cudnn7.6(Tensorflow 框架)
Cuda10.2+cudnn7.6(Pytorch 框架)
tensorflow (1.14.0,<2)
tensorrt (7.0)
Pillow
Numpy
Pycuda(>=2017.1.1)
onnx(1.6.0)
wget(>=3.2)
相应的环境对应服务器 192.168.0.106 中的镜像 pytorch/tensorrt:1.5.0-7.0-cuda10.2 和 tensorflow/tensorrt:1.14.0-7.0.0

二、安装 TensorRT

1. 下载 Tensorrt 安装包(选择自己需要的版本)，需要 nvidia 账号，链接 https://developer.nvidia.com/tensorrt。下载完后解压，指令:
sudo tar -xvzf TensorRT-7.0.0.11.Ubuntu-18.04.x86_64-gnu.cuda-10.2.cudnn7.6.tar.gz
2.解压之后在目录下会生成" TensorRT-7.0.0.11"文件夹。进入该文件夹，有各种子文件夹如图 2.1 所示。

```
user@7dee2858f27c:/home/TensorRT-7.0.0.11$ ls
TensorRT-Release-Notes.pdf  bin  data  doc  graphsurgeon  include  lib  python  samples  targets  uff
```

图 2.1

3.安装 Tensorrt，输入指令:
(1)cd python，如下图 2.2 所示

```
user@7dee2858f27c:/home/TensorRT-7.0.0.11/python$ ls
tensorrt-7.0.0.11-cp27-none-linux_x86_64.whl  tensorrt-7.0.0.11-cp35-none-linux_x86_64.whl  tensorrt-7.0.0.11-cp37-none-linux_x86_64.whl
tensorrt-7.0.0.11-cp34-none-linux_x86_64.whl  tensorrt-7.0.0.11-cp36-none-linux_x86_64.whl
```

图 2.2

(2)pip install tensorrt-7.0.0.11-cp36-none-linux_x86_64.whl,选择对应的 python 版本安装
4.安装 uff，在" TensorRT-7.0.0.11"文件夹路径下输入指令:
(1)cd uff
(2)pip install uff-0.6.5-py2.py3-none-any.whl
5.安装 graphsurgeon，在" TensorRT-7.0.0.11"文件夹路径下输入指令:
(1)cd graphsurgeon
(2) pip install graphsurgeon-0.4.1-py2.py3-none-any.whl
6.验证是否安装成功，依次输入以下指令，无报错则安装成功
(1)python
(2)import tensorrt
注：验证过程中可能出现如图 2.3 所示 bug，添加环境变量即可,注意改为自己的 TensorRT-7.0.0.11 路径:
vim ~/.bashrc

export LD_LIBRARY_PATH=/home/TensorRT-7.0.0.11/lib:$LD_LIBRARY_PATH
source ~/.bashrc

```
>>> import tensorrt
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/home/user/.local/lib/python3.6/site-packages/tensorrt/__init__.py", line 1, in <module>
    from .tensorrt import *
ImportError: libnvinfer.so.7: cannot open shared object file: No such file or directory
```

图 2.3

三、TemsorRT 加速模型

3.1 tensorflow 的 pb 模型—>uff 模型

以 TensorRT-7.0.0.11/samples/python/end_to_end_tensorflow_mnist 为例进行操作说明。

1、 在 end_to_end_tensorflow_mnist 文件夹下打开命令窗口，按以下指令顺序操作

(1)cd mkdir models

(2)python model.py

2、模型训练完成后，保存为 models/lenet5.bp

(1)进入 TensorRT-7.0.0.11/graphsurgeon 目录下，会看到 graphsurgeon-0.4.1-py2.py3-none-any.whl 文件，在此目录下打开命令窗口，输入指令:

pip install graphsurgeon-0.4.1-py2.py3-none-any.whl

(2)进入 TensorRT-7.0.0.11/uff 目录下，会看到 uff-0.6.5-py2.py3-none-any.whl 文件，在此目录下打开命令窗口，输入指令:

pip install uff-0.6.5-py2.py3-none-any.whl

(3) 经过上述 2 个命令，已经成功安装 convert-to-uff 。此时再次进入到 end_to_end_tensorflow_mnist 目录下，打开命令窗口输入指令:

convert-to-uff models/lenet5.pb，此时在 models 目录下会生成 lenet5.uff 文件

(4)使用命令 cd home/TensorRT-7.0.0.11/data/mnist(视自己的文件放置路径进行调整)，在 mnist 文件夹下存在 download_pgms.py 文件，在该路径下输入指令:

python download_pgms.py 成功运行后生成了 pgm 文件如图 2.1 所示。

```
root@afad06ba8e5d:/home/TensorRT-7.0.0.11/data/mnist# ls
0.pgm  5.pgm  README.md          lenet5_custom_pool.uff      mnist.prototxt
1.pgm  6.pgm  deploy.prototxt    lenet5_custom_pool.uff.txt  mnist_lenet.caffemodel
2.pgm  7.pgm  download_pgms.py   lenet5_mnist_frozen.pb      mnist_mean.binaryproto
3.pgm  8.pgm  lenet5.uff         mnist.caffemodel            mnistapi.wts
4.pgm  9.pgm  lenet5.uff.txt     mnist.onnx                  mnistgie.wts
```

图 3.1

(5)再次进入/TensorRT-7.0.0.11/samples/python/end_to_end_tensorflow_mnist 路径下，输入指令 **python sample.py –d /home/TensorRT-7.0.0.11/data/mnist**，若出现如图 2.2 所示画面则运行成功。(/home/TensorRT-7.0.0.11/data/mnist 为上一步生成 pgm 文件的路径)

```
root@afad06ba8e5d:/home/TensorRT-7.0.0.11/samples/python/end_to_end_tensorflow_mnist# python sample.py -d /home/TensorRT-7.0.0.11/data/mnist
WARNING: /usr/src/tensorrt/data/mnist does not exist. Trying /usr/src/tensorrt/data instead.
WARNING: /usr/src/tensorrt/data does not exist. Please provide the correct data path with the -d option.
WARNING: /home/TensorRT-7.0.0.11/data/mnist/mnist does not exist. Trying /home/TensorRT-7.0.0.11/data/mnist instead.
[TensorRT] WARNING: Current optimization profile is: 0. Please ensure there are no enqueued operations pending in this context prior to swit
ching profiles
Test Case: 2
Prediction: 2
```

图 3.2

3.2 Tensorflow 的 weights 模型→trt 模型

以/home/TensorRT-7.0.0.11/samples/python/yolov3_onnx 为例进行操作说明。

1、在 yolov3_onnx 文件夹下打开命令窗口，使用 ls 命令可以看到该文件夹包含的文件如下图 3.3 所示。输入指令: **sudo python2 yolov3_to_onnx.py**，运行完成后会在当前目录下生成 yolov3.cfg 和 yolov3.weights 文件。

```
coco_labels.txt  data_processing.py  onnx_to_tensorrt.py  README.md  requirements.txt  yolov3_to_onnx.py
```

图 3.3

注: 必须使用 python2 进行, 否则会报 bug, 安装包文件时使用的 pip 也必须指向 python2, 使用 pip –V 可以查看 pip 的执行。

2、继续在 yolov3_onnx 文件夹下的命令窗口输入 **sudo python onnx_to_tensorrt.py**，此时对 python 版本无限制，建议使用 python3。注：可能会报类似如图 3.4 所示的 bug，解决方案(修改为自己的路径):

sudo cp /home/TensorRT-7.0.0.11/targets/x86_64-linux-gnu/lib/lib* /usr/lib/

```
user@1ea13e944087:/home/TensorRT-7.0.0.11/samples/python/yolov3_onnx$ sudo python onnx_to_tensorrt.py
Traceback (most recent call last):
  File "onnx_to_tensorrt.py", line 54, in <module>
    import tensorrt as trt
  File "/home/user/.local/lib/python3.6/site-packages/tensorrt/__init__.py", line 1, in <module>
    from .tensorrt import *
ImportError: libnvinfer.so.7: cannot open shared object file: No such file or directory
```

图 3.4

运行成功后如图 3.5 所示，文件目录以及生成 trt 的 engine 文件。

```
user@1ea13e944087:/home/TensorRT-7.0.0.11/samples/python/yolov3_onnx$ sudo python onnx_to_tensorrt.py
Downloading from https://github.com/pjreddie/darknet/raw/f86901f6177dfc6116360a13cc06ab680e0c86b0/data/dog.jpg, this may take a while...
.
100% [..................................................................] 163759 / 163759
Loading ONNX file from path yolov3.onnx...
Beginning ONNX file parsing
Completed parsing of ONNX file
Building an engine from file yolov3.onnx; this may take a while...
Completed creating Engine
[TensorRT] WARNING: Current optimization profile is: 0. Please ensure there are no enqueued operations pending in this context prior to
 switching profiles
Running inference on image dog.jpg...
[[135.14841894 219.59879722 184.3020796  324.0265199 ]
 [ 98.30801929 135.72611092 499.71271304 299.25581993]
 [478.00606108  81.25701899 210.57788522  86.91503643]] [0.99854713 0.99880403 0.93829265] [16  1  7]
Saved image with bounding boxes of detected objects to dog_bboxes.png.
```

图 3.5