1.snpe 官网下载安装包，各种版本，Downloas-->Software-->Actlive
https://developer.qualcomm.com/sites/default/files/docs/snpe/tutorial_alexnet.html

2.pip 安装依赖包

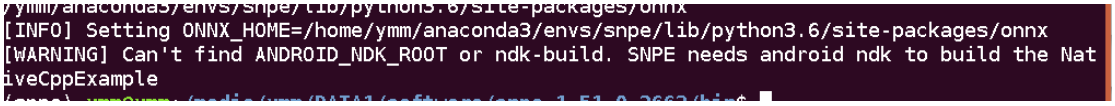
```
Package            Version
------------------ ---------
certifi            2021.5.30
commonmark         0.9.1
cycler             0.11.0
dataclasses        0.8
decorator          4.4.2
docutils           0.18.1
flatbuffers        22.9.24
imageio            2.15.0
Jinja2             3.0.3
kiwisolver         1.3.1
MarkupSafe         2.0.1
matplotlib         3.3.4
networkx           2.5.1
numpy              1.19.5
onnx               1.6.0
onnx-simplifier    0.4.8
onnxruntime        1.10.0
opencv-python      3.4.4.19
Pillow             8.4.0
pip                21.2.2
protobuf           3.6.0
Pygments           2.12.0
pyparsing          3.0.9
python-dateutil    2.8.2
PyWavelets         1.1.1
PyYAML             3.10
rich               12.6.0
scikit-image       0.17.2
scipy              1.4.1
setuptools         59.5.0
six                1.16.0
Sphinx             1.2.2
tifffile           2020.9.3
typing_extensions  4.1.1
```

3.加入环境变量
gedit ~/.bashrc

export LD_LIBRARY_PATH="/media/ymm/DATA1/software/snpe-1.51.0.2663/lib/x86_64-linux-clang:$LD_LIBRARY_PATH"
export SNPE_ROOT=/media/ymm/DATA1/software/snpe-1.51.0.2663
export PATH="/media/ymm/DATA1/software/snpe-1.51.0.2663/bin/x86_64-linux-clang:$PATH"
export PYTHONPATH=$PYTHONPATH:/media/ymm/DATA1/software/snpe-1.51.0.2663/lib/python
export ANDROID_NDK_ROOT=~/Android/Sdk/ndk-bundle

source ~/.bashrc

4.运行 onnx 转 dlc，需要激活 onnx 环境，命令如下：
cd /media/ymm/DATA1/software/snpe-1.51.0.2663/bin
source envsetup.sh -o /home/ymm/anaconda3/envs/snpe/lib/python3.6/site-packages/onnx
此时，可能会出现如下错误：



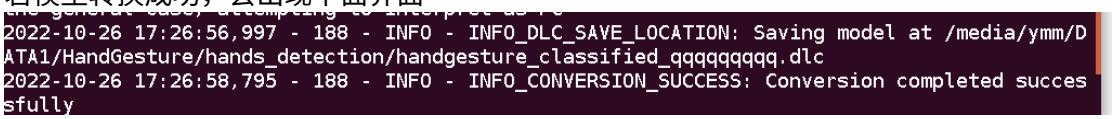
若情况允许可以安装 Android Studio，下载 ANDROID_NDK，指定 ANDROID_NDK_ROOT，如没有，对于模型转换这一步也没影响的。

export ANDROID_NDK_ROOT=~/Android/Sdk/ndk-bundle

5.进行模型转换
cd /media/ymm/DATA1/software/snpe-1.51.0.2663/bin/x86_64-linux-clang

./snpe-onnx-to-dl -i model.onnx -o model.dlc

若模型转换成功，会出现下面界面

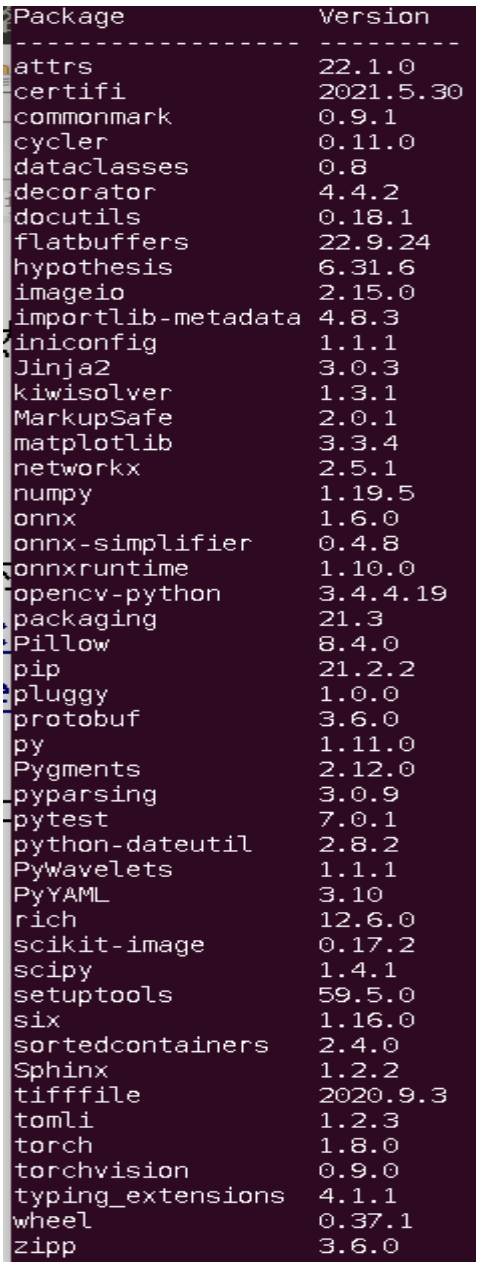


若转换不成功，尝试一下 onnxsim，命令如下：
python3 -m onnxsim model.onnx model_sim.onnx
再次执行 ./snpe-onnx-to-dl -i model_sim.onnx -o model.dlc

若仍然转换不成功，可分析模型算子是否支持，修改算子等。

以上为 snpe-1.51 配置过程，目前 snpe-1.65 支持 snpe-pytorch-to-dlc.。可参考 https://developer.qualcomm.com/sites/default/files/docs/snpe/model_conv_pytorch.html

与以上不同的是：
1）pip list (多了一些依赖库)
pip install hypothesis　pytest　torchvision==0.9



Package	Version
attrs	22.1.0
certifi	2021.5.30
commonmark	0.9.1
cycler	0.11.0
dataclasses	0.8
decorator	4.4.2
docutils	0.18.1
flatbuffers	22.9.24
hypothesis	6.31.6
imageio	2.15.0
importlib-metadata	4.8.3
iniconfig	1.1.1
Jinja2	3.0.3
kiwisolver	1.3.1
MarkupSafe	2.0.1
matplotlib	3.3.4
networkx	2.5.1
numpy	1.19.5
onnx	1.6.0
onnx-simplifier	0.4.8
onnxruntime	1.10.0
opencv-python	3.4.4.19
packaging	21.3
Pillow	8.4.0
pip	21.2.2
pluggy	1.0.0
protobuf	3.6.0
py	1.11.0
Pygments	2.12.0
pyparsing	3.0.9
pytest	7.0.1
python-dateutil	2.8.2
PyWavelets	1.1.1
PyYAML	3.10
rich	12.6.0
scikit-image	0.17.2
scipy	1.4.1
setuptools	59.5.0
six	1.16.0
sortedcontainers	2.4.0
Sphinx	1.2.2
tifffile	2020.9.3
tomli	1.2.3
torch	1.8.0
torchvision	0.9.0
typing_extensions	4.1.1
wheel	0.37.1
zipp	3.6.0

2）环境变量：
export LD_LIBRARY_PATH="/media/ymm/DATA1/software/snpe-1.65.0.3676/lib/x86_64-linux-clang:$LD_LIBRARY_PATH"
export SNPE_ROOT=/media/ymm/DATA1/software/snpe-1.65.0.3676
export PATH="/media/ymm/DATA1/software/snpe-1.65.0.3676/bin/x86_64-linux-clang:$PATH"
export PYTHONPATH=$PYTHONPATH:/media/ymm/DATA1/software/snpe-1.65.0.3676/lib/python
export ANDROID_NDK_ROOT=~/Android/Sdk/ndk-bundle

3）激活环境
source envsetup.sh -o /home/ymm/anaconda3/envs/snpe/lib/python3.6/site-packages/torch
export ANDROID_NDK_ROOT=~/Android/Sdk/ndk-bundle

4)模型转换命令
./snpe-pytorch-to-dlc --input_network resnet18.pt --input_dim input "1,3,224,224" --output_path resnet18.dlc

注意：该部分的 torch 模型为 torch.jit.trace 后的模型，可用 libtorch 部署的模型。样例如下：
```
import torch
import torchvision.models as models
resnet18_model = models.resnet18()
input_shape = [1, 3, 224, 224]
input_data = torch.randn(input_shape)
script_model = torch.jit.trace(resnet18_model, input_data)
script_model.save("resnet18.pt")
```

snpe 模型量化
参考：
https://blog.csdn.net/Guo_Python/article/details/127048810?spm=1001.2014.3001.5501

（1）准备量化的输入图片数据：首先从测试集中获取 N 张图片(N 的取值范围推荐在 50~200 之间)，然后将图片转换成 raw 文件，转换脚本可参考

```
#!/usr/bin/python
# coding=utf8
# Author: guopei

import numpy as np
import os
import cv2
import shutil
import random
from tqdm import tqdm


def get_image_path_list(dataset_path_list, N):
    """
    从数据集中随机抽取 N 张图片
    Args:
        dataset_path_list:
        N:

    Returns:

    """
    image_path_list = []
    for dataset_path in dataset_path_list:
        for dir_path, _, file_names in os.walk(dataset_path):
            for file_name in file_names:
                # if 'input' not in file_name:
                #     continue
                img_path = os.path.join(dir_path, file_name)
                image_path_list.append(img_path)
    random.shuffle(image_path_list)
    return image_path_list[:N]


def preprocess(image):
    """
    对图片进行预处理，image 的通道顺序需要是 BGR
    Args:
        image:

    Returns:

    """
```

```
    image = np.array(image)
    # 如果输出是 float32，那么输入也一定要转换成 float32
    image = cv2.resize(image, (528, 784), cv2.INTER_LINEAR)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2YUV)
    image = image[:,:,0]
    image = image.astype(np.float32)
    # 归一化
    image = image / 255.
    return image


def convert(image_path_list, raw_data_dir_save_path):
    """
    把图片转换成 raw 文件
    Args:
        image_path_list:      待转换的图片的路径列表
        raw_data_dir_save_path: raw 文件的保存路径

    Returns:

    """
    if os.path.exists(raw_data_dir_save_path):
        shutil.rmtree(raw_data_dir_save_path)
    os.mkdir(raw_data_dir_save_path)
    raw_data_path_list_file = raw_data_dir_save_path + '_list.txt'
    if os.path.exists(raw_data_path_list_file):
        os.remove(raw_data_path_list_file)

    with open(raw_data_path_list_file, 'w') as f:
        for index, img_path in enumerate(image_path_list):
            raw_name = os.path.basename(img_path).split('.')[0] + '.raw'
            raw_data_path = os.path.join(raw_data_dir_save_path, raw_name)
            f.write(raw_data_path + '\n')
            image = cv2.imread(img_path)
            image = preprocess(image)
            image.tofile(raw_data_path)


if __name__ == '__main__':
    image_path_list = get_image_path_list(
        dataset_path_list=['/home/guopei/jpgs'], N=2)
    convert(image_path_list=image_path_list,
            raw_data_dir_save_path='./img_raws')
```

转换过程中需要注意以下几点：

一定要按照模型对输入数据的要求对图片进行预处理；

一定要将图片转换成 float32；

转化结束后，将得到一个包含所有 raw 文件的文件夹和一个包含所有 raw 文件绝对路径的 list 文件。

（2）模型量化：

```
snpe-dlc-quantize
--input_dlc my.dlc
--input_list quantize_data_list.txt
--output_dlc my_quant.dlc
--optimizations cle --optimizations bc –enable_htp
```

–input_list 就是上一步骤生成的包含所有 raw 文件绝对路径的 list 文件，
–optimizations cle --optimizations bc 对应的量化算法效果最佳，但是这种量化算法对模型结构有一定的限制且量化速度较慢，如果使用这种量化算法出现量化失败，可以将–optimizations cle --optimizations bc 这两个选项去掉，使用 snpe 默认的量化算法。
–enable_htp 选项一定需要， 否则 dsp 运行的初始化非常慢， 这里千万注意，官网没有介绍；

（3） **验证量化后的模型**

snpe-net-run --container my_quant.dlc --input_list ./quantize_data_list.txt --output_dir ./my_dir

将得到的 raw 转换成图片，可参考代码：

```
import numpy as np
import os
import cv2
import shutil


def postProcess(net_output):
    predEdge = (net_output[..., 0:1] * 255).astype(np.uint8)
    predSegment = (net_output[..., 1:2] * 255).astype(np.uint8)
    return predEdge, predSegment
```

```
def convert(raw_data_dir_path, converted_image_dir_save_path):
    if os.path.exists(converted_image_dir_save_path):
        shutil.rmtree(converted_image_dir_save_path)
    os.mkdir(converted_image_dir_save_path)

    count = 0
    for dir_path, _, filenames in os.walk(raw_data_dir_path):
        for filename in filenames:
            if 'raw' not in filename:
                continue
            raw_file_path = os.path.join(dir_path, filename)
            net_output = np.fromfile(raw_file_path, dtype='float32').reshape((384, 384, 2))
            predEdge, predSegment = postProcess(net_output)
                        edge_save_path  =  os.path.join(converted_image_dir_save_path,
str(count).zfill(3) + '_output1.jpg')
                    segment_save_path  =  os.path.join(converted_image_dir_save_path,
str(count).zfill(3) + '_output2.jpg')
            print(edge_save_path)
            count += 1
            cv2.imwrite(edge_save_path, predEdge)
            cv2.imwrite(segment_save_path, predSegment)


if __name__ == '__main__':
    raw_data_dir_path = 'dlc_output_now'
    converted_image_dir_save_path = raw_data_dir_path + '_convert'
    convert(raw_data_dir_path, converted_image_dir_save_path)
```