Amlogic

模型转换运行用户指南

Revision: 0.9

Release Date: 2021-08-10

Copyright

© 2021 Amlogic. All rights reserved. No part of this document may be reproduced. Transmitted, transcribed, or translated into any language in any form or by any means with the written permission of Amlogic.

Trademarks

Amlogic, and other Amlogic icons are trademarks of Amlogic companies. All other trademarks and registered trademarks are property of their respective companies.

Disclaimer

Amlogic may make improvements and/or changes in this document or in the product described in this document at any time.

This product is not intended for use in medical, life saving, or life sustaining applications.

Circuit diagrams and other information relating to products of Amlogic are included as a means or illustrating typical applications. Consequently, complete information sufficient for production design is not necessarily given. Amlogic makes no representations or warranties with respect to the accuracy or completeness of the contents presented in this document.

Contact Information

- Website: www.amlogic.com
- Pre-sales consultation: contact@amlogic.com
- Technical support: support@amlogic.com

变更说明

版本 0.9（2021-08-10）

第 9 版。

1、添加对新 IP 的量化指导

2、添加 perchannel 量化对应指导以及限制条件

版本 0.8（2021-01-15）

第 8 版。

3、添加环境安装注意事项

4、添加量化时使用多张图片的指导

版本 0.7（2020-7-18）

第 7 版。

5、增加多输入模型转换指导

6、删除 IDE 工具使用指导

版本 0.6（2020-05-18）

第 6 版。

增加了 3.5 和 3.6 章节。

版本 0.5（2019-12-20）

第五版。

1. 增加使用 acuity 工具进行 inference 操作指导
2. 增加模型转换过程中量化参数细节指导
3. 移除 FAQ 章节，将对应章节内容移到 Amlogic-NN-FAQ 文档中
4. 增加 darknet、Keras 框架模型导入指导

版本 0.4（2019-07-18）

第四版。

1. 优化环境安装步骤，提供一键式安装方式
2. 模型转换增加模型量化简介以及量化参数选取指导
3. 删除工程代码编译章节，case 代码编译在<Android&Linux 开发指导.docx>文档中体现
4. 删除通用知识章节
5. FAQ 章节中增加量化/反量化指导，以指导客户解决反馈的前处理时间过长的问题

版本 0.3（2019-07-08）

第三版。

1. 添加 ONNX 模型转换命令
2. 模型转换工具添加 whl 包安装提醒说明
3. DDK_6.3.3.4 离线模型变更为 nbg 类型，修改对应模块说明
4. DDK_6.3.3.4 模型转换出 case 代码后直接生成 case 代码目录，取消代码整理章节。
5. 增加 case 代码简介章节

版本 0.2（2018-12-20）

第二版。

1. 默认环境安装错误点/遗漏点修改
2. 不再使用 IDE 转换 case 代码，删除对应章节，部分章节介绍进行调整。
3. 将代码转换流程精简，部分流程删除，将问题解答移到第七章节。
4. 增加 IDE 工具使用介绍章节，介绍 case 代码导入，添加 jpeg 库，编译运行等流程。
5. 工程代码章节添加外部人员获取版本方式，增加基于内部代码和外部代码进行 case 代码编译介绍。
6. 添加 FAQ 章节，将常见问题归纳输出。

版本 0.1（2018-11-12）

第一版。

目录

1. 简介

本文将仔细介绍使用 Acuity_tool 工具将模型转换出 case 代码的过程以及细节。指导公司内部或者外部相关开发人员进行模型转换。

当前我们的 NN 芯片只支持 tensorflow、caffe、tflite、darknet、onnx、pytorch, keras 类型的模型。如果开发者是其他类型的模型，需先将模型转换成上述支持的一种。

2. 工具介绍

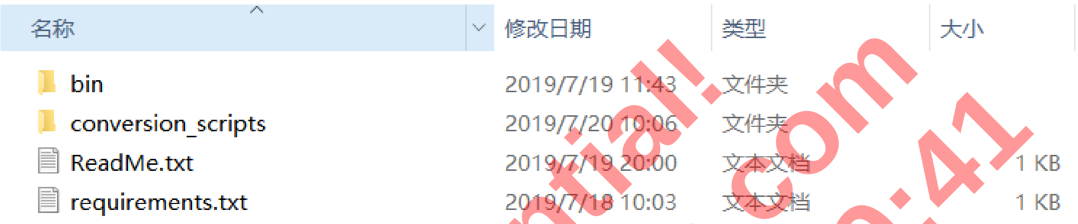
2.1 简介

当前将模型转换出 case 代码需要的工具：

1. acuity-toolkit 模型工具，用于将模型量化，并转换出 case 代码，编译后即可运行。

2.2 模型转换工具

2.2.1 工具目录简介



名称	修改日期	类型	大小
bin	2019/7/19 11:43	文件夹	
conversion_scripts	2019/7/20 10:06	文件夹	
ReadMe.txt	2019/7/19 20:00	文本文档	1 KB
requirements.txt	2019/7/18 10:03	文本文档	1 KB

bin：模型转换需要用到的可执行文件以及 config 配置文件目录

conversion_scripts：

1、模型转换 demo（包含转换脚本程序和 mobilenet_v1.pb 模型）

2、转换只需要在环境安装后之后按照顺序执行三个 sh 脚本即可。

3、转自己的模型，修改三个脚本对应的配置参数再进行转换。

requirements.txt：环境依赖的安装包列表。

ReadMe.txt：简单的文档简介

2.2.2 环境依赖

操作系统	Ubuntu16.04（x64）
Python 版本	Python3.5.2

依赖库	tensorflow==1.13.2 numpy==1.16.4 scipy==1.1.0 Pillow==5.3.0 protobuf==3.9.0 networkx==1.11 image==1.5.5 lmdb==0.93 onnx==1.4.1 h5py==2.9.0 flatbuffers==1.10 matplotlib==2.1.0 dill==0.2.8.2 ruamel.yaml==0.15.81 onnx_tf==1.2.1 ply==3.11 torch==1.2.0

注：上述只是将当前需要的一些库列出来，具体的依赖库以工具内部 acuity_toolkit-binary-xxx 里面的 requirements.txt 为准。

2.2.3 工具安装

Step 1. 环境准备

Ubuntu 16.04 系统 64 位的计算机（如果是虚拟机，RAM 内存设置>4GB），Python 需要的是 3.5.2 版本。

Step 2. 安装 python3 以及 pip 等工具

```
sudo apt-get install python3 python3-pip python3-virtualenv
```

Step 3. 安装相关依赖包

获取 acuity-toolkit 工具包，进入根目录。

执行：for req in $(cat requirements.txt); do pip3 install $req; done

Step 4. Check 环境

```
python3 bin/checkenv.py
```

注：check 成功会有：Env Pass: Env check SUCCESS!!!打印。

3. 模型转换

3.1 模型转换简介

模型转换过程是先将模型量化成 int8、int16 或者 uint8 数据格式，再转成基于我们平台运行的 nbg 格式文件，并导出运行的 case 代码。

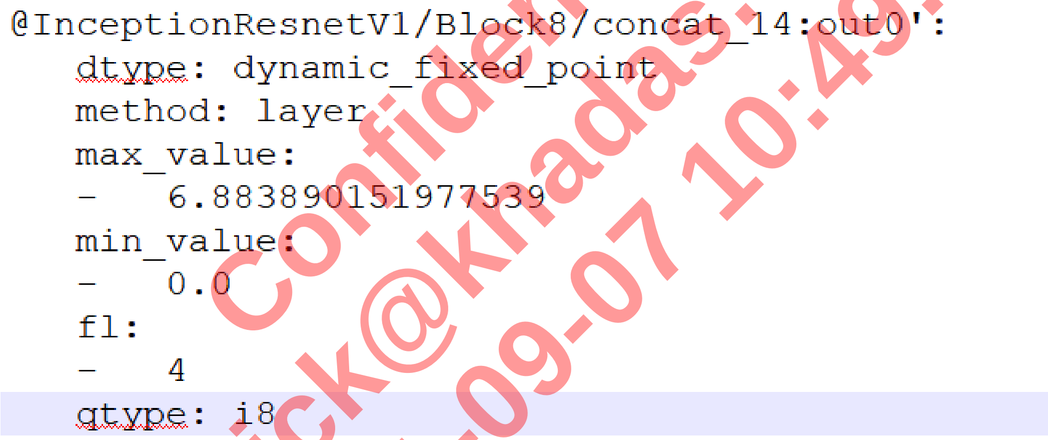
3.2 量化简介

量化：量化是将以往用 32bit 或者 64bit 表达的浮点数用 16bit、8bit 或者更低的 2bit 方式进行存储。

3.2.1 量化详解

Int8：将 float32 数据量化成 int8 的数据，8 个 bit 位，一个为符号位。其他 7 个 bit 位来表示有效数字。需算出 fl 值。fl：表示小数的 bit 位个数。（int16 与 int8 类似）

可以参考 3.4 章节 step2 里面生成的 xxxx.quantize，分析如下图：



```
@InceptionResnetV1/Block8/concat_14:out0':
   dtype: dynamic_fixed_point
   method: layer
   max_value:
   -    6.883890151977539
   min_value:
   -    0.0
   fl:
   -    4
   qtype: i8
```

解析：qtype 表示量化方式为 int8。最小/最大值范围为（0.0~6.88）

输出结果的最大绝对值为 6.88，只需要 3 个 bit 就可以表示整数位。

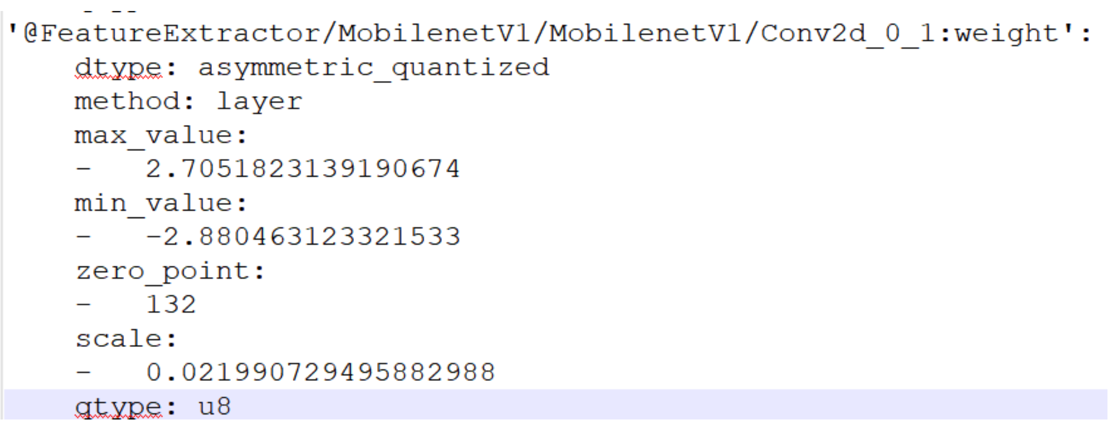
所以用来表示小数的位数 fl=8-1-3=4。

u8：将 float32 数据量化成 unsigned int8 的数据。8 个 bit 位均用来表示数字，没有符号位。即：将最大/最小范围区间映射到 0~255 之间。需计算出两个参数，scale:映射比例 zero_point：零点的位置。计算公式如下图：

scale = (max_value - min_value)/255

zero_point = max_value - max_value/scale

参考 3.4 章节 step2 里面生成的 xxxx.quantize。



```
'@FeatureExtractor/MobilenetV1/MobilenetV1/Conv2d_0_1:weight':
    dtype: asymmetric_quantized
    method: layer
    max_value:
    -   2.7051823139190674
    min_value:
    -   -2.880463123321533
    zero_point:
    -   132
    scale:
    -   0.021990729495882988
    qtype: u8
```

解析：qtype 表示量化方式为 u8。输出的最小/最大值范围为（-2.88~2.7）

量化参数计算：

scale = (2.705 - （-2.88）)/255= 0.0219

zero_point = 255 - 2.705/scale = 132

3.2.2 量化方式选取指导

量化方式选取规则：

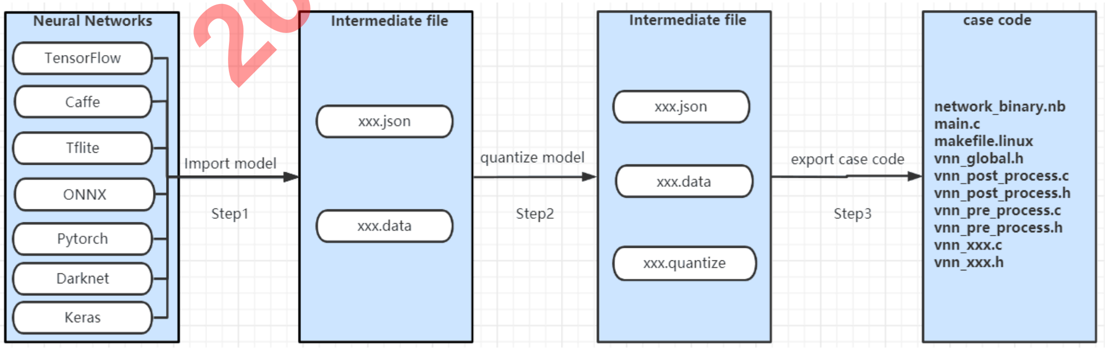
1. 通常情况下，选择 u8/int8 的方式均可。u8 是 google 推荐量化方式。

2. 一般建议不选取 int16，相较 8bit 量化，模型大一倍，并在精度上没有明显的优势。

3. 参考模型前处理后的取值范围选取量化方式。比如，如果模型前处理后的取值范围为（0~255），此时建议直接选取 u8 的量化方式。

4. 先随便选择一种量化方式，执行完量化操作后，查看生成的 xxxx.quantize 文件，确认统计的 max_value/min_value,如果出现绝对值>256 的情况，建议选取 int16 方式量化。如果绝对值出现>128 的情况，建议选取 u8 量化方式，其他情况选择 int8/u8 均可。

3.3 模型转换流程图



3.4 模型转换步骤

当前模型转换过程都是在 acuity-toolkit 目录下进行的。可执行文件均在 bin 目录下。

Step 1. 导入模型

命令：

```
Caffe：
 ./bin/convertcaffe    --caffe-model xx.prototxt
          --caffe-blobs xx.caffemodel --data-output xxx.data --net-output xxx.json

Darknet：
./bin/convertdarknet   --net-input yolov2.cfg   \
          --weight-input yolov2.weights --data-output yolov2.data --net-output yolov2.json

Tensorflow：
./bin/convertensorflow --tf-pb inceptionV1.pb   \
          --inputs input   --outputs InceptionV1/Logits/Predictions/Reshape_1 \
          --net-output inceptionV1.json   --data-output inceptionV1.data     \
          --input-size-list '224,224,3'

Tflite:
./bin/convertflite    --tflite-mode    ssd_big_graph.tflite    \
          --net-output ssd_big_graph.json --data-output ssd_big_graph.data

Onnx：
./bin/convertonnx    --onnx-model    vgg16.onnx    \
          --net-output vgg16.json    --data-output vgg16.data

Keras:
./bin/convertkeras --keras-model simple_CNN.81-0.96.hdf5 \
           --net-output cnn.json --data-output cnn.data

Pytorch:
./bin/convertpytorch --pytorch-model cnn_new.pt --net-output cnn.json \
           --data-output cnn.data --input-size-list '1,28,28'
```

结果：1. 生成两个中间文件 xxx. json 和 xxx. data

注：

1. 转换 tensorflow 模型时，需要知道模型的输入输出节点名称以及输入的 HWC 尺寸。tensorflow 模型可以使用 summarize_graph 或者 tensorboard 来找对应节点以及尺寸

2. --inputs/--outputs, 输入输出节点名称。如果有多个输入或者输出，请使用空格分隔每个输出，例如 “output1 output2 output3”

3. --input-size-list, 如果有多个输入，请使用#分隔每个输入大小，例如：

“224, 224, 3#299, 299, 3#12, 12”

4. 当前只支持 Caffe/Tensorflow/Tflite/Darknet/Onnx/pytorch/keras 等模型，如果是其它类型模型，需要先将模型转成当前支持的类型

5. tensorflow 模型如果存在一些逻辑控制的输入，请参照 Amlogic-NN-FAQ 文档中 3.17 指导进行处理。

Step 2. 对模型进行量化

命令：

```
./bin/tensorzonex    --action quantization \
                     --quantized-dtype    asymmetric_affine-u8    \
                     --channel-mean-value '128 128 128 1'    \
                     --source text --source-file dataset.txt            \
                     --model-input xxx.json -- model-data xxx.data \
                     --reorder-channel '0 1 2'    \

                     --quantized-rebuild
```

结果：根据 dataset.txt 提供的输入图片，进行前向推理计算，统计每层 layer 输出的最大/最小值以及模型权值的最大/最小值，同时计算出每层的量化参数，保存成量化结果文件 xxx.quantize，导出 case 代码时会用到该量化文件。

注：

1. --channel-mean-value 根据训练模型时预处理方式来设置该预处理的命令行参数。包括四个值（m1,m2,m3,scale）.前三个值为均值参数，最后一个值为 scale 参数。对于输入为三通道数据（data1，data2，data3），预处理过程为：

Out1 = (data1-m1)/scale

Out2 = (data2-m2)/scale

Out3 = (data3-m3)/scale

例如：如果前处理需将数据归一化到[-1,1]之间，参数设置为（128 128 128 128）

如果前处理需将参数归一化到[0,1]之间，参数设置为（0 0 0 256）。

如果是单通道参数，设置方式为（m1,0,0,scale）

2. dataset.txt 中是给定的量化输入图片路径，如：/data/test/cat.jpg。建议提供 200 张左右，同时也是模型使用运行场景的图片来进行量化，确保统计出的最大/最小值为实际运行场景的最大/最小值范围，这样量化效果会更佳。

图片数量>100 时，建议添加量化参数--batch-size(默认 100)和--epochs(默认 1)，epochs*batch_size=图片数量。例如 5000 张图片，设置参数为：--batch-size 100 --epochs 50。

如果电脑内存较小，可以将 batch_size 的值设置成较小的值。

3. --quantized-dtype 用于量化的数据类型，支持的类型为：

dynamic_fixed_point-i8（int8 量化）

dynamic_fixed_point-i16（int16 量化）

asymmetric_affine-u8 （uint8 量化，默认）

perchannel_symmetric_affine-i8 （perchannel 量化，当前只有 E8 对应芯片支持）

4. --quantized-rebuild 默认参数，建议携带。在前后使用不同的量化方式进行量化时，如果没有手动删除已生成的 xxx.quantize 文件，在没有携带该参数时，第二次量化未做任何处理。

5. --reorder-channel 设置数摆放据格式。

caffe 模型是 bgr 格式,即:'2 1 0'

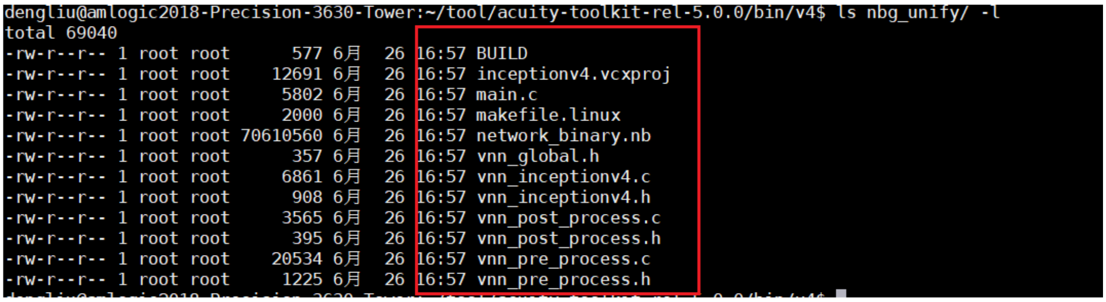
tensorflow 模型为 rgb 格式，即:’ 0 1 2’

Step 3. 生成 case 代码

命令：

```
./bin/ovxgenerator    --model-input 190328.json \
            --data-input 190328.data \
            --channel-mean-value '128 128 128 1' \
            --reorder-channel '0 1 2' \
            --export-dtype quantized \
            --optimize VIPNANOQI_PIDOX88   \
            --viv-sdk ../bin/vcmdtools \
            --pack-nbg-unify
```

结果：生成 case 代码，如下图：



```
dengliu@amlogic2018-Precision-3630-Tower:~/tool/acuity-toolkit-rel-5.0.0/bin/v4$ ls nbg_unify/ -l
total 69040
-rw-r--r-- 1 root root      577 6月  26 16:57 BUILD
-rw-r--r-- 1 root root    12691 6月  26 16:57 inceptionv4.vcxproj
-rw-r--r-- 1 root root     5802 6月  26 16:57 main.c
-rw-r--r-- 1 root root     2000 6月  26 16:57 makefile.linux
-rw-r--r-- 1 root root 70610560 6月  26 16:57 network_binary.nb
-rw-r--r-- 1 root root      357 6月  26 16:57 vnn_global.h
-rw-r--r-- 1 root root     6861 6月  26 16:57 vnn_inceptionv4.c
-rw-r--r-- 1 root root      908 6月  26 16:57 vnn_inceptionv4.h
-rw-r--r-- 1 root root     3565 6月  26 16:57 vnn_post_process.c
-rw-r--r-- 1 root root      395 6月  26 16:57 vnn_post_process.h
-rw-r--r-- 1 root root    20534 6月  26 16:57 vnn_pre_process.c
-rw-r--r-- 1 root root     1225 6月  26 16:57 vnn_pre_process.h
```

注：

1. xxx.quantize 文件在执行时没有作为入参，是默认读取的 xx.json, xxx.data。

2. --reorder-channel 设置数据摆放据格式。

caffe 模型是 bgr 格式，即：'2 1 0'

tensorflow 模型为 rgb 格式，即：' 0 1 2'

3. --channel-mean-value 设置参数跟量化时候的保持一致，见 Step2。

4. --optimize 当前设置值为 VIPNANOQI_PIDOX88。当前有六个有效值：

VIPNANOQI_PIDOX7D、VIPNANOQI_PIDOX88

VIPNANOQI_PIDOX99、VIPNANOQI_PIDOXA1

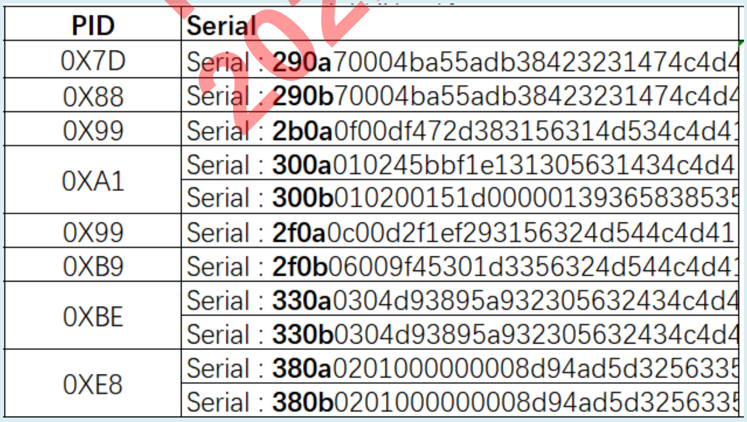
VIPNANOQI_PIDOXB9、VIPNANOQI_PIDOXBE

VIPNANOQI_PIDOXE8

可以使用如下方式来确认应该设置的值，执行：cat /proc/cpuinfo

查看倒第二行数 Serial 对应的数列码**前四个字符（Serial 后面粗体字部分）**

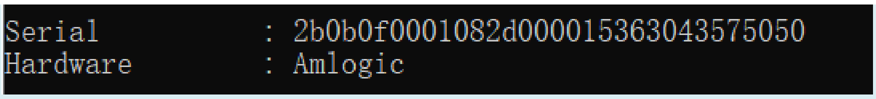
序列号和参数 VIPNANOQI_PIDOX？？ 的对应关系如下图：



PID	Serial
0X7D	Serial : **290a**70004ba55adb38423231474c4d4
0X88	Serial : **290b**70004ba55adb38423231474c4d4
0X99	Serial : **2b0a**0f00df472d383156314d534c4d41
0XA1	Serial : **300a**010245bbf1e131305631434c4d4
	Serial : **300b**0102000151d000001393658385
0X99	Serial : **2f0a**0c00d2f1ef293156324d544c4d41
0XB9	Serial : **2f0b**06009f45301d3356324d544c4d41
0XBE	Serial : **330a**0304d93895a932305632434c4d4
	Serial : **330b**0304d93895a932305632434c4d4
0XE8	Serial : **380a**0201000000008d94ad5d325633
	Serial : **380b**0201000000008d94ad5d325633

例如：

*在串口或者 cmd 窗口执行命令：*cat /proc/cpuinfo 后，Serial 如下图



```
Serial          : 2b0b0f0001082d000015363043575050
Hardware        : Amlogic
```

此时，参数应该设置为：--optimize VIPNANOQI_PID0XB9

5.--viv-sdk 依赖的sdk包，acuity_tool_xxx/bin目录下vcmdtools目录，填写相对路径即可。

6.--pack-nbg-unify 生成nbg 文件。

7、4/5/6三个对应参数设置后，会生成nbg的casedemo，我们通常使用的就nbg case demo。

此时在模型目录下也会生成normal case demo，执行下面两行命令，可以将normal case整理到一个目录中：

mkdir normal_case_demo

mv *.h *.c .project .cproject *.vcxproj *.lib BUILD *.linux *.export.data normal_case_demo

二者区别：

Normal_case：加载模型时，会有在线编译时间，耗时较长。

android平台：支持直接跑normal case

linux平台：不支持直接跑normal case。1.Linux平台需将acuity_tool_xx/bin/vcmdtools目录push到板子data目录，然后设置环境变量：export VIVANTE_SDK_DIR=/data/vcmdtoos。2.还需要将buildroot_sdk_6462目录里buildroot_sdk/build/so/drivers_64_exportdata/对应so push到板子上，设置环境变量：export LD_LIBRARY_PATH=/xxx/xxx/drivers_64_exportdata

NBG case：在线编译这一步在pc端已经完成，板子上能直接加载nb文件，模型加载速度快。

8、nbg case转换成功标志：case代码目录下有network_binary.nb文件。

9、demo是模型转换目录，会在它同级目录下生成demo_nbg_unify目录，此为nbg case demo目录

生成的case代码简介

生成的代码在nbg_unify目录下。

1. network_binary.nb是生成的nbg文件，保存有模型的权值和graph，该文件名称可以根据自已喜好修改。
2. vnn_inceptionv4.c 模型创建和释放实现对应的代码文件
3. vnn_pre_process.c模型前处理文件，将读入图片数据量化成8bit数据，可以复用文件中接口，也可以自行实现量化。
4. vnn_post_process.c 模型后处理文件，当前转出的case代码后处理只是做了top5处理，可以根据模型需要，自行添加后处理流程。

注：

1. 根据step3的log确认生成了哪些文件，需要注意下（如caffe ssd网络会生成一个mbox_priorbox_185.bin文件）

2. 工具包demo目录为提供的模型转换demo：

Demo中执行顺序：

Step1：0_import_model.sh

Setp2：1_quantize_model.sh

Step3：2_export_case_code.sh

3. 当前执行完三步，可将 demo 中的 mobilenet_v1 模型转出 case 代码。

4. 转换自己的模型，可以将模型放到该目录下，然后修改 sh 脚本中的参数即可。

5. Demo 中的 extractoutput.py，执行命令：python extractoutput.py xxx.json 可以生成 outnamelist.txt，里面记录的是 Blob name 与 tensor ID 的 map 关系，在取结果时候，可以参考下。

Step 4. PC 端推理仿真

```
./bin/tensorzonex   --action inference \
                    --source text   \
                    --source-file test.txt \
                    --channel-mean-value '127.5 127.5 127.5 128'   \
                    --model-input xxx.json \
                    --model-data xxx.data \
                    --dtype quantized
```

注：

1. test.txt 里面放置的就是需要在 pc 端推理的图片，一般是给一张图片，确认精度问题时候使用。

2. 在执行 step1 导入模型之后执行 step4，是 pc 推理非量化模型的结果，即（float 结果）。

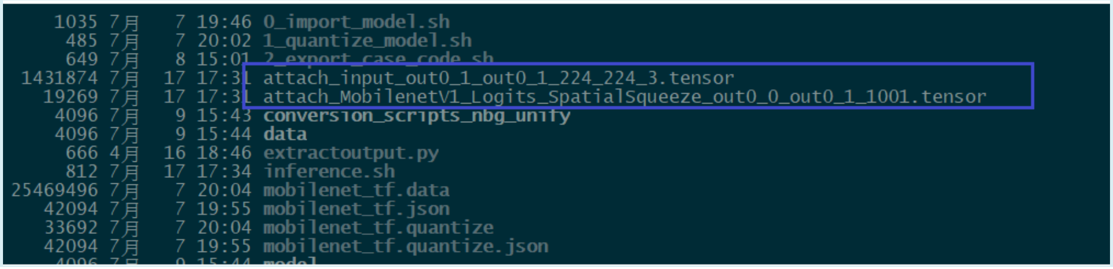
3. 在执行 step2 量化模型之后执行 step4，是 pc 推理量化后的模型结果，执行时候会默认去读取量化后生成的 xxx.quantize 文件，即（quantize 结果）。如果已经执行了量化操作，存在 xxx.quantize 文件，可以设置参数：--dtype float32 来 inference float 的结果。或者是将 xxx.quantize 删掉执行上面命令即可。

4. --channel-mean-value 的设置跟量化时候设置的值一样即可。

5. 执行完 inference 之后，会在当前目录保存输入输出对应的 tensor 文件。输入文件为做完前处理之后的 float 数据。输出文件为已经反量化成 float 的数据。一般确认板侧运行的精度问题，建议使用 inference 保存的 tensor 文件作为输入文件。

例如：

Mobilenet 模型执行 inference 之后，保存了输入输出的 tensor 文件。可以使用 inference 时保存的输入 tensor 文件：attach_input_out0_1_out0_1_224_224_3.tensor 作为板子上 demo 执行时的输入文件。



```
    1035 7月    7 19:46 0_import_model.sh
     485 7月    7 20:02 1_quantize_model.sh
     649 7月    8 15:01 2_export_case_code.sh
 1431874 7月   17 17:31 attach_input_out0_1_out0_1_224_224_3.tensor
   19269 7月   17 17:31 attach_MobilenetV1_Logits_SpatialSqueeze_out0_0_out0_1_1001.tensor
    4096 7月    9 15:43 conversion_scripts_nbg_unify
    4096 7月    9 15:44 data
     666 4月   16 18:46 extractoutput.py
     812 7月   17 17:34 inference.sh
25469496 7月    7 20:04 mobilenet_tf.data
   42094 7月    7 19:55 mobilenet_tf.json
   33692 7月    7 20:04 mobilenet_tf.quantize
   42094 7月    7 19:55 mobilenet_tf.quantize.json
    4096 7月    9 15:44 model
```

3.5 扩展参数说明

3.5.1 简介

3.4 章节中介绍的模型转换步骤过程中的参数是当前整理的最简洁的使用参数，一般情况下只使用上述参数即可正常转换出 case demo。当前有一些客户会有更多的需求，需要知道量化和导出命令对应的其他参数情况，本章节对量化命令 tensorzonex 以及导出命令 ovxgenerator 命令的所有参数进行扩展说明，以供使用。

3.5.2 Tensorzonex 扩展参数

此命令行工具(tensorzonex/tensorzonex.py)可用于裁剪、验证和其他操作。所有参数都是可选的，某些参数是执行特定任务所必须的。允许的参数会以粗体显示。模型量化和 PC 端 inference 操作会用到此命令行工具。

```
tensorzonex [-h] [-h] [--action ACTION] [--debug] [--dtype DTYPE]
                            [--device DEVICE] --model-input MODEL_INPUT
                            [--model-data MODEL_DATA]
                            [--model-quantize MODEL_QUANTIZE]
                            [--model-data-format MODEL_DATA_FORMAT]
                            [--validation-output VALIDATION_OUTPUT]
                            [--source SOURCE] [--source-file SOURCE_FILE]
                            [--restart] [--batch-size BATCH_SIZE]
                            [--samples SAMPLES] [--config CONFIG]
                            [--output-num OUTPUT_NUM] [--data-output DATA_OUTPUT]
                            [--epochs EPOCHS] [--optimizer OPTIMIZER] [--lr LR]
                            [--epochs-per-decay EPOCHS_PER_DECAY]
                            [--quantized-dtype QUANTIZED_DTYPE]
                            [--quantized-moving-alpha QUANTIZED_MOVING_ALPHA]
                            [--quantized-algorithm QUANTIZED_ALGORITHM]
                            [--quantized-divergence-nbins QUANTIZED_DIVERGENCE_NBINS]
                            [--quantized-rebuild] [--quantized-rebuild-all]
                            [--quantized-hybrid] [--reorder-channel REORDER_CHANNEL]
                            [--input-fitting INPUT_FITTING]
                            [--input-normalization INPUT_NORMALIZATION]
                            [--channel-mean-value CHANNEL_MEAN_VALUE]
                            [--mean-file MEAN_FILE]
                            [--caffe-mean-file CAFFE_MEAN_FILE] [--random-crop]
                            [--random-mirror] [--random-flip]
                            [--random-contrast RANDOM_CONTRAST]
                            [--random-brightness RANDOM_BRIGHTNESS] [--force-gray]
                            [--task TASK] [--prune-epochs PRUNE_EPOCHS]
                            [--prune-loss PRUNE_LOSS] [--pfps-epochs PFPS_EPOCHS]
                            [--pfps-reduce-target PFPS_REDUCE_TARGET]
                            [--pfps-delta0 PFPS_DELTA0]
                            [--without-update-masked-grad]
                            [--capture-format CAPTURE_FORMAT] [--capture-quantized]
                            [--output-dir OUTPUT_DIR] [--pb-name PB_NAME]
```

Arguments：

General Purpose：

-h help file

--action What to do for the network.(Optional)

The following values are valid for the related activity:

inference - for inference model

measure - for calcuate MACCs and PARAMs

prune - for iterative pruning and retraining

quantization - for Quantization

snapshot - for dump result layer by layer

test - for testing(Default)

train - for training

--batch-size Integer value which specifies the batch size (number of images in a batch). (Optional)

100 (Default).

CAUTION: Set this value smaller if the RAM or GPU cannot support such batch size.

--caffe-mean-file Input caffe mean file. Default is None. Example file:~/IMAGENET_1300_VAL/imagenet_mean.binaryproto

--config Reserved

--debug Set full debug model on.Produces additonal debug output.

Default is off.

--device Sepecify the compute device:

gpu:0 /cpu:0

if more than one GPU is present, specify by incrementing the number e.g, gpu:2.Note:The current tool is cpu,can't use gpu.

--dtype Data type used for calculation.(Optional)

Allowed values are:

float32 - for float32 networks(Use this value to speed quantize.Default)

quantized - for quantized networks.

--epochs Integer value specifying the number of batches of images used to train the model in every PFP-S interation. Each epoch uses

The entire training set. Default is 1

--lr Learning rate in float format.Default is **0.1.** Required when **ACTION**=train

--mean-file Input mean patch and filename.Default is None.

--model-data Neural Network coefficient data input path and filename. Default uses the same patch and replaces the .json of MODEL_INPUT with .data.Rqquired when **ACTION**=quantization or train.

--model-data-format Neural Network model data input format file type.

Valid options are: **zone**(Default) or **acuity**

--model-input Neural Network model data input path and filename.

Required when **ACTION**=prune,quantization,train,inference, test,snapshot,or validate.

--restart When training,do not load the coefficients and retrain.Use this option if you want to start from scratch.

--samples Sample total size.Default is -1.

--source Dataset source type.Currently supported:

sqlite - the source file has a .dsx extension and is a self-contained set of images. The file size will likely be large.

text - the source file has a .txt extension and contains a list of images,one per line,and additional parameters for the image which together make a dataset.

Required when **ACTION**=prune,quantization or train.

--source-file Dataset path and filename. The file extension indicates the dataset:

.dsx (implies source is **sqlite**)

.txt(implies source is a comma delimited **text** file)

Required when **ACTION**=prune,quantization or train.

--validation-output Validation output file.Default is in the tensorzonex binary floder with the name of “va;odatopm.csv”

--pb-name Save Graph and Coefficients in Tensorflow PB format.

Inference Argument:

--output-num Number of outputs(integer value). Default is 5.Only valid for Use **ACTION**=inference.

Quantization Argument:

--model-quantize Neural Network quantized tensors’ description file(Optional)

A *.quantize file. If not specified,the default uses the same path as MODEL_INPUT and replace the .json with .quantize

Please specify this parameter if EXPORT_DTYPE is specified as “quantized”

--quantized-dtype Data type used for quantized.(Optional)

Valid **dtype** values are:

asymmetric_affine-u8 - asymmetric affine unsigned 8bit

dynamic_fixed_point-i8 – dynamic fixed point signed 8bit

dynamic_fixed_point-i16 –ynamic fixed point signed 16bit

perchannel_symmetric_affine-i8 – per-channel quantization for asymmetric affine signed 8 bit

Default is u8

--quantized-algorithm Quantization algorithm(Optional)

Valid values are：

kl_divergence

moving_average

normal(default)

--quantized-moving-alpha If **QUANTIZED_ALGORITHM** is specified as "moving_average", please set this parameter

--quantized-divergence-nbins "kl_divergence", please set this parameter If **QUANTIZED_ALGORITHM** is specified as

--quantized-rebuild Rebuild quantize table containing default specified tensors, mutually exclusive with argument – **quantized-rebuild-all, -- quantized-hybird**

--quantized-rebuild-all Rebuild quantize table containing all tensors, mutually exclusive with argument **--quantized-rebuild, - -quantized-hybrid**

--quantized-hybrid Setup a hybrid quantize network by quantize table.

mutually exclusive with argument **--quantized-rebuild, --quantized-rebuild-all**

This parameter needs to be specified if you want to generate a hybrid mode application

Transformation Arguments:

--channel-mean-value input channel mean value

--force-gray Force channel to gray.Enabled by default.

--input-fittling How to fit the image input to the network.

Crop -crop or pad the image.

Scale -scale the image(Default)

--input-Normalization Normalization method for input into the network:**image,pixel**,or **None**.Default is **None**.

--random-brightness Augment training images with randomized brightness (provide max delta) Format type is float.

--random-contrast Augment training images with randomized contrast (provide min-max contrast).

--random-crop Augment the training images with randomized crop.

--random-flip Augment training images with randomized flip (vertical). Enabled by default.

--random-mirror Augment training images with randomized mirroring (horizontal). Enabled by default.

--reorder-channel Use the same channel reorder value as used for network train/test/inference to change channel to BGR or RGB. A string value: Use "**2 1 0**" for BGR,”**0 1 2**” for RGB

Snapshot Only

Arguments:

--capture-format Define snapshot tensor file data sequence. **nchw** (Default) , **nhwc**

--capture-quantized Define whether or not snapshot tensor file data should be in quantized format.If this parameter is specified, data will be saved to quantized format instead of float.

--output-dir Snapshot data file store path. If not specified, default is storage path is folder ‘./snapshot/’ .

Train Only Arguments:

--epochs-per-decay Integer value indicating the number of epochs required for next decay for learning rate decaying algorithms. Default is 100. Only valid for use with ACTION=train.

--data-output Network coefficient data output file path and filename where trained data should be saved. Default uses the location of the .json file and replaces the .json of MODEL_INPUT with .data. If not set, the original data (which is set using －model-data) is overwritten. Used only with **ACTION**=train, **ACTION**=prune.

--optimizer Stochastic Gradient Descent (SGD) Optimizer:

Static

Momentum (Default).

Only valid for use with **ACTION**=train

3.5.3 Ovxgenerator 扩展参数

此命令行工具(ovxgenerator 或 ovxgenerator.py)获取模型和数据的输入并将其转换为可以使用 ovxlib 构建和运行的应用程序。导出 case demo 那一步会用到此命令。

```
ovxgenerator    [-h] --model-input MODEL_INPUT --data-input DATA_INPUT
                     [--model-quantize MODEL_QUANTIZE]
                     [--model-output MODEL_OUTPUT] [--optimize OPTIMIZE]
                     [--export-dtype EXPORT_DTYPE]
                     [--channel-mean-value CHANNEL_MEAN_VALUE]
                     [--reorder-channel REORDER_CHANNEL]
                     [--save-fused-graph] [--pack-nbg-unify]
                     [--pack-nbg-viplite] [--viv-sdk VIV_SDK]
                     [--build-platform BUILD_PLATFORM]
                     [--target-ide-project TARGET_IDE_PROJECT]
                     [--batch-size BATCH_SIZE] [--force-remove-permute]
```

Required Arguments

--data-input Neural Network coefficient data input filename. Required.

--model-input Neural Network model input filename. Required.

Optional Arguments

-h Show help test on the console

--model-quantize Neural Network quantized tensors’ description file. (Optional).A *.quantize file. If not specified, the default uses the same path as MODEL_INPUT and replaces the .json with .quantize. Please specify this parameter if EXPORT_DTYPE is specified as ‘quantized’

--model-output Neural Network model output name. (Optional) If not specified, the default uses the network name as the model output name.

--optimize Optimize the exporting network according to the specified hardware configuration path or configuration name. (Optional).

none – no optimization

Default (Default) - If a configuration file or configuration name is not specified, it will use default export rule to export application code.If --pack-* is given, please specify a configuration file path or configuration name instead of **none** or **Default**.

--export-dtype Export case to given data type. (Optional)

String values in [‘float’ , ‘quantized’]

float (Default) - specify to export float16 case.

quantized - specify to export quantized case.

--channel-mean-Value Input channel mean value. (Optional)

This parameter should always be given according to running models.The same channel mean value would be used for network train/test/inference/snapshot/validate.

--reorder-channel Use the same channel reorder value as used for network train/test/inference/snapshot/validate to change the channel to BGR or RGB. (Optional) This is a string value and should always be given according to running models

Use "**2 1 0**" for BGR. Use **"0 1 2**" for RGB.

Default is "", which stands for do not do any reorder channel operation at all.

--save-fused-graph Whether to save fused graph. (Optional).A fused graph is a network description file which contains the network structure of exported application code. It is useful for debug.

--viv-sdk SDK dir, if one of --pack-* is given, please specify a folder containing the binary sdk of vSimulator.

--pack-nbg-unify Pack binary graph for unify driver.(Optional) mutually exclusive with other --pack-*.If this feature is enabled, two cases will be generated,a unify case and a nbg_unify case. If no --pack-* is given, only the unify case will be generated.

--batch-size Batch size for exported application code. (Optional)

Integer value in[1,+nan] **1** (Default)

--force-remove-permutes Force remove the head permutation layers inserted by T2C and directly connected to input graph layers; tails permutated layers inserted by T2C and directly connected to

graph output layers.(Optional) This parameter should only

be specified for models in Tensorflow/TensorflowLite

formats, because models in Tensorflow sequence would

trigger T2C.

CAUTION: If specified, please make sure to feed the

correct tensor data to the application. For example, a

Tensorflow network with input of [1,224,224,3](nhwc)

and meet the remove permute condition.

-If this parameter is not specified, the user should feed

tensor with shape [1,224,224,3](nhwc) to the application.

-If this parameter is specified, the user should feed the

tensor with shape [1,3,224,224](nchw) to application.

Similarly, corresponding transpositions are also needed

for every output layer that removed the permutations

in post-process code.

3.6 混合量化

混合量化是对同一个模型的不同 layer 根据精度来使用不同的量化方式。例如，某个模型的某层 layer 的 8bit 量化效果差，导致最终结果误差大，我们可以使用混合量化方式，将对应的 layer 设置不同的量化方式，来提高精度.

Note：避免理解偏差，当前章节直接使用原始的英文介绍流程。

3.6.1 简要流程

1、Use original xxx.data xxx.json file to quantize network into asymmetric_affine-u8 dtype as usual with parameter '--quantized-rebuild', which results in an xxx.quantize file.

2、In the xxx.quantize file, add the layer names and their corresponding quantized_dtype (choose from dynamic_fixed_point-i8, dynamic_fixed_point-i16, asymmetric_affine-u8, float32) to be changed to customized_quantize_layers.

3、Use the xxx.quantize file generated in Step 2, and the original xxx.data xxx.json file to quantize the network into asymmetric_affine-u8 dtype with parameter '--quantized-hybrid'.

4、Use the xxx.data, xxx.quantize.json, and xxx.quantize files generated in Step 3 to export application code as usual. DATACONVERT layers will be inserted into the graph.

3.6.2 详细步骤

Use lenet model as an example to show how to change a layer from quantized dtype to a float dtype:

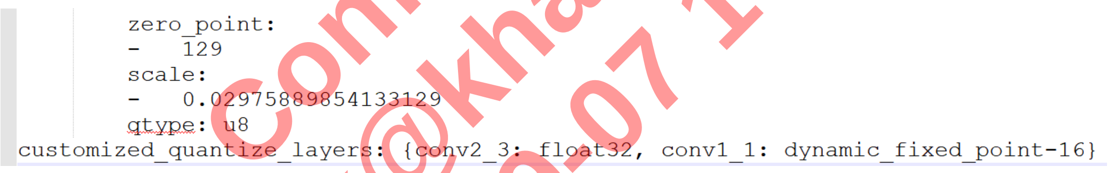
1、Quantize as usual to generate the lenet.quantize file:

```
$ ./tensorzonex --action quantization \
        --dtype float32 \
        --model-input ~/lenet/lenet.json \
        --model-data ~/lenet/lenet.data \
        --quantized-dtype asymmetric_affine-u8 \
        --quantized-rebuild \
        --source text \
        --source-file ~/lenet/dataset.txt \
        --reorder-channel '2 1 0' \
        --channel-mean-value '0 0 0 256':
```

2、Add layer name ‘conv2_3’ and corresponding quantized_dtype ‘float32’ to **customized_quantize_layers** of lenet.quantize.

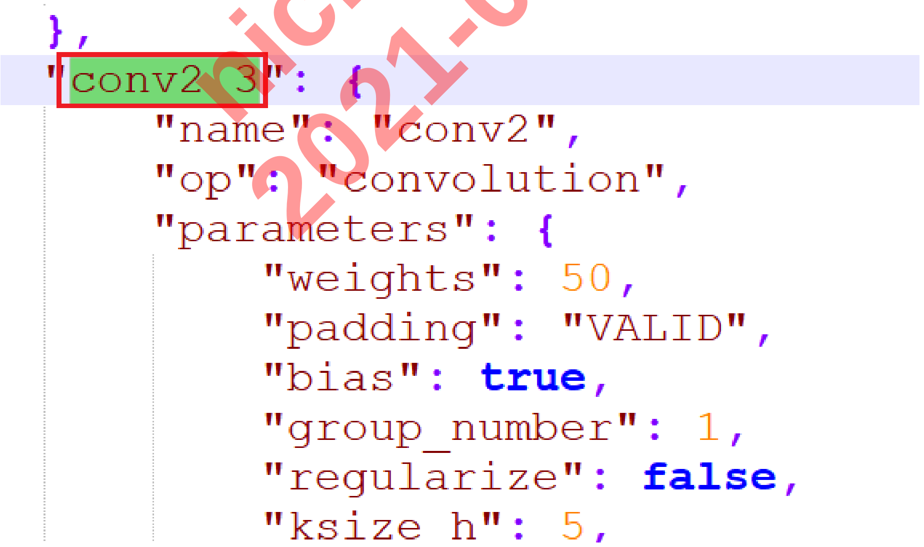
```
customized_quantize_layers: {conv2_3: float32}

Or customized_quantize_layers: {conv2_3: float32, conv1_1: dynamic_fixed_point-16}

CAUTION: if you want to change a connected subgraph of the network to non-quantized layers, set all
the layers in this subgraph to  ‘float32’ , and you can get the layer name from the *.json file.
```

For example：



```
            zero_point:
            -   129
            scale:
            -   0.029758898541331 29
            qtype: u8
customized_quantize_layers: {conv2_3: float32, conv1_1: dynamic_fixed_point-16}
```

The layer name from xxx.json file ， Specific as shown below:



```
  },
  "conv2_3": {
      "name": "conv2",
      "op": "convolution",
      "parameters": {
          "weights": 50,
          "padding": "VALID",
          "bias": true,
          "group_number": 1,
          "regularize": false,
          "ksize_h": 5,
```

3、 Quantize the network into asymmetric_affine-u8 dtype again with parameter ‘--quantizedhybrid’. This will result in two new files lenet.quantize and lenet.quantize.json as well as some dtype_converter layers added into lenet.quantize.json.

```
$ ./tensorzonex --action quantization \
        --dtype float32 \
        --model-input ~/lenet/lenet.json \
        --model-data ~/lenet/lenet.data \
        --model-quantize ~/lenet/lenet.quantize
        --quantized-dtype asymmetric_affine-u8 \
        --quantized-hybrid \
        --source text \
        --source-file ~/lenet/dataset.txt \
        --reorder-channel '2 1 0' \
        --channel-mean-value '0 0 0 256'
```

4、Use the lenet.data, lenet.quantize.json, lenet.quantize files generated by Step 3 to export application code (refer to Section 0) which wil result in some DATACONVERT layers inserted into the graph in the exported case.

```
$ ./ovxgenerator --model-input lenet.quantize.json \
        --data-input lenet.data \
        --model-quantize lenet.quantize
        --export-dtype quantized \
        --channel-mean-value '0 0 0 256' \
        --optimize VIPNANOQI_PID0X88   \
        --viv-sdk ../bin/vcmdtools \
        --pack-nbg-unify

Note: --optimize VIPNANOQI_PID0X88  Refer to chapter 3.4 step3
```

3.7 多输入模型转换

3.7.1 简介

多输入模型与单输入模型在转换过程中的区别在于：量化时需要分别给多个输入对应的量化数据集，使其能够正常量化。多输入模型在转换过程中会多出一步： generate inputmeta，会生成一个 xxx_inputmeta.yml 文件，通过该文件来设置不同输入的量化数据集以及均值参数。

本章节介绍一个超级命令行工具：pegasus。Pegasus 命令行工具集成了 3.4~3.5 章节中介绍的所有功能，还添加了一些额外功能。对于单输入为 4 维度的模型，使用 3.4 章节介绍或者 pegasus 均可以进行转换。但对于多输入模型，只能使用 pegasus 进行转换。

3.7.2 多模型转换

```
Step1 Import Model

        Caffe：

          ./bin/pegasus    import caffe \

                --model     ./model_lw3.prototxt \

                --weights model_lw3.caffemodel \
                --output-model ./model_lw3.json \
               --output-data ./model_lw3.data
        ONNX:
             ./bin/pegasus    import onnx \
             --model      ./model_UnetBased_0620v8-ep44-seg3ch.onnx \
             --inputs "0 1 2" \
             --input-size-list "288,288,3#288,288,3#288,288,3" \
             --outputs "327 328" \
             --output-model ./${NAME}.json \
             --output-data ./${NAME}.data
```

注：当前工具支持的几种模型都能通过这个脚本导入，上面只根据已有模型列了三个，其他框架模型可参照 3.7.3 章节导入

```
Step2 generate inputmeta
          ./bin/pegasus generate inputmeta --model ./${NAME}.json    \
                          --separated-database True
```

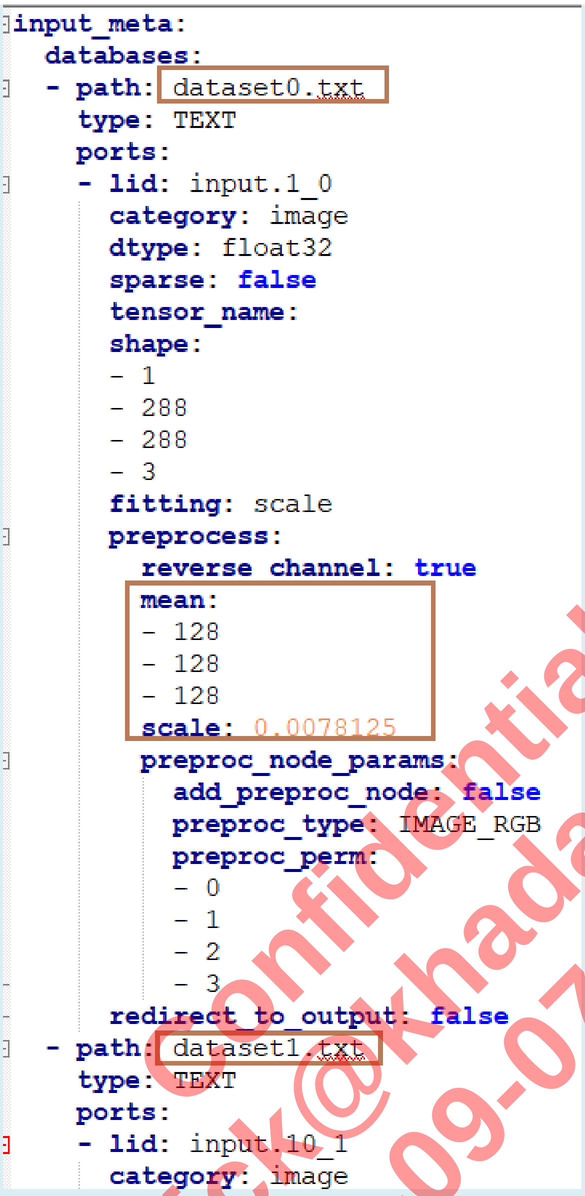
结果：生成${NAME}_inputmeta.yml

注：

1、--separated-database 默认为 false，生成的 xxx_inputmeta.yml 文件里面只生成一个 dataset.txt 的路径，该设置适合多输入的 tensor 尺寸均相同的情况，此时 dataset.txt 一行设置多张图片，空格隔开即可：

```
    echo "space_shuttle_224x224.jpg goldfish_224x224.jpg"    > dataset.txt
```

2、--separated-database 设置为 true 时，，生成的 xxx_inputmeta.yml 文件里面每个输入都有一个对应的 dataset.txt 文件，分开设置不同输入的量化数据集，如多输入的 tensor 尺寸存在不相同的情况下，需要设置成 true。



```
input_meta:
  databases:
  - path: dataset0.txt
    type: TEXT
    ports:
    - lid: input.1_0
      category: image
      dtype: float32
      sparse: false
      tensor_name:
      shape:
      - 1
      - 288
      - 288
      - 3
      fitting: scale
      preprocess:
        reverse_channel: true
        mean:
        - 128
        - 128
        - 128
        scale: 0.0078125
        preproc_node_params:
          add_preproc_node: false
          preproc_type: IMAGE_RGB
          preproc_perm:
          - 0
          - 1
          - 2
          - 3
      redirect_to_output: false
  - path: dataset1.txt
    type: TEXT
    ports:
    - lid: input.10_1
      category: image
```

3、转换过程中的均值也是通过改 yml 文件设置，如上图，mean、scale 对应的是均值设置。与前面 3.4 章节中不同的是 scale 的值是 float 小数，如，1.0/128,此处请注意。

Step3 Quantize Model

```
./bin/pegasus quantize \
    --model ${NAME}.json \
    --model-data ${NAME}.data \
    --with-input-meta model_lw3_inputmeta.yml \
    --quantizer asymmetric_quantized   \
    --qtype uint8 --rebuild
```

注：

1、量化参数通过 xxx_imputmeta.xml 文件来设置。

2、其他参数参照 3.7.3 章节介绍

3、量化数据如果是多张图片，可以使用--batch-size 和 --iterations 来设置，batch-size*iterations = 图片张数。

```
Step4 Export Model:
    ./bin/pegasus  export ovxlib\
        --model ${NAME}.json \
        --model-data ${NAME}.data \
        --model-quantize ${NAME}.quantize \
        --with-input-meta model_lw3_inputmeta.yml \
        --dtype quantized \
        --optimize VIPNANOQI_PID0XB9  \
        --viv-sdk ${ACUITY_PATH}vcmdtools \
        --pack-nbg-unify
```

注：

1、--optimize VIPNANOQI_PID0XB9 参照 3.4 章节 Step3 步骤的设置

2、--viv-sdk ${ACUITY_PATH}vcmdtools 参照 3.4 章节 Step3 步骤的设置

3、其他参数介绍参照 3.7.3 章节

3.7.3 PASUS 扩展参数

```
./bin/pegasus import caffe
    [-h] --model MODEL [--weights WEIGHTS]
    [--proto PROTO] [--output-model OUTPUT_MODEL]
    [--output-data OUTPUT_DATA]

./bin/pegasus import tensorflow
    [-h] --model MODEL --inputs INPUTS
    --input-size-list INPUT_SIZE_LIST –outputs
    [--size-with-batch SIZE_WITH_BATCH]
    OUTPUTS [--mean-values MEAN_VALUES]
    [--std-values STD_VALUES]
    [--predef-file PREDEF_FILE]
    [--subgraphs SUBGRAPHS]
    [--output-model OUTPUT_MODEL]
    [--output-data OUTPUT_DATA]

./bin/pegasus import tflite
    [-h] --model MODEL [--output-model OUTPUT_MODEL]
    [--output-data OUTPUT_DATA] [--outputs OUTPUTS]


 ./bin/pegasus import darknet
      [-h] --model MODEL --weights WEIGHTS
      [--output-model OUTPUT_MODEL]
      [--output-data OUTPUT_DATA]
 ./bin/pegasus import onnx
      [-h] --model MODEL [--inputs INPUTS] [--outputs OUTPUTS]
      [--input-size-list INPUT_SIZE_LIST] [--size-with-batch SIZE_WITH_BATCH]
      [--input-dtype-list INPUT_DTYPE_LIST]
      [--output-model OUTPUT_MODEL]
      [--output-data OUTPUT_DATA]

 ./bin/pegasus import pytorch
      [-h] [--model MODEL]
      [--inputs INPUTS]
      [--outputs OUTPUTS]
      [--input-size-list INPUT_SIZE_LIST]
      [--size-with-batch SIZE_WITH_BATCH]
      [--output-model OUTPUT_MODEL]
      [--output-data OUTPUT_DATA]
      [--config CONFIG]

 ./bin/pegasus import keras
      [-h] --model MODEL
      [--convert-engine {Keras,TFLite}]
      [--inputs INPUTS]
      [--input-size-list INPUT_SIZE_LIST]
      [--outputs OUTPUTS]
      [--output-model OUTPUT_MODEL]
      [--output-data OUTPUT_DATA]

./bin/pegasus generate inputmeta
      [-h] [--input-meta-output INPUT_META_OUTPUT]
      [--separated-database SEPARATED-DATABASE]
      --model MODEL

./bin/pegasus quantize
      [-h] --model MODEL --model-data MODEL_DATA
      [--model-quantize MODEL_QUANTIZE]
      [--batch-size BATCH_SIZE] [--iterations ITERATIONS]
      [--device {GPU,CPU}]
      [--with-input-meta WITH_INPUT_META]
      [--quantizer {dynamic_fixed_point,asymmetric_quantized,asymmetric_affine,
      symmetric_affine,perchannel_symmetric_affine}]
      [--qtype QTYPE] [--hybrid] [--rebuild]
      [--algorithm {normal,moving_average,kl_divergence}]
      [--moving-average-weight MOVING_AVERAGE_WEIGHT]
      [--divergence-nbins DIVERGENCE_NBINS]

./bin/pegasus export ovxlib
      [-h] --model MODEL --model-data MODEL_DATA
      [--model-quantize MODEL_QUANTIZE]
      [--output-path OUTPUT_PATH]
      [--with-input-meta WITH_INPUT_META]
      [--optimize OPTIMIZE] [--dtype {float,quantized}]
      [--save-fused-graph]
      [--pack-nbg-unify] [--pack-nbg-viplite]
      [--viv-sdk VIV_SDK]
      [--build-platform {make}]
      [--target-ide-project TARGET_IDE_PROJECT]
      [--batch-size BATCH_SIZE]

./bin/pegasus inference
      [-h] --model MODEL --model-data MODEL_DATA
      [--model-quantize MODEL_QUANTIZE]
      [--batch-size BATCH_SIZE] [--iterations ITERATIONS]
      [--device {GPU,CPU}]
      [--with-input-meta WITH_INPUT_META]
      [--dtype {float32,quantized}]
      [--postprocess {classification_classic,print_topn,dump_results}]
      [--postprocess-file POSTPROCESS_FILE]
```

```
          [--output-dir OUTPUT_DIR]
```

Pegasus Arguments	Description(for gegasus arguments)
Import Caffe Arguments	
-h	Help file.
--model	Caffe model filename. (Required)
--output-model	Acuity net Neural Network layer set output file. (Optional). If not specified, uses the path where the Caffe proto file is located, the output filename is the name defined in proto file.
--output-data	Acuity net Neural Network coefficient data output file. (Optional) If not specified, default uses the path where the Caffe proto file is located, the output filename is the name defined in proto file.
--proto	Switch protocol used for the Caffe binary protocol buffer file (binaryproto.caffemodel) associated with the CAFFE_MODEL. (Optional) **caffe** – to import Caffe models from standard Caffe format protocol (Default) (http://caffe.berkeleyvision.org/) **lstm_caffe** – use networks containing LSTM layer protocol. CAUTION: To import non-standard Caffe models, please put relevant protocol file xxx_pb2.py into the same folder as caffe_pb2.py, and then specify this parameter as ‘xxx’ to import models.
--weights	Caffe weights filename. (Optional) Default assumes path and filename will be the same as that of the **CAFFE_MODEL**. If the blob file does not exist, random fake data will be generated.
Import Tensorflow Arguments	
-h	Help file.
--output-model	Acuity net Neural Network layer set output file. (Optional) If not specified, the default uses the path where the Tensorflow protobuf file is located, and the output filename is the same as the ptotobuf file. CAUTION: For quant models, there will be an extra .quantize file (quantize table) generated, after imported, DO NOT quantize model again, just use imported model inference/snapshot/export.
--output-data	Acuity net Neural Network coefficient data output file.

	(Optional) If not specified, the default uses the path where the protobuf file is located, the output file name is the same as the ptotobuf file.
Tensorflow	
--model	Tensorflow frozen protobuf file. (Required) CAUTION: Because every version of Tensorflow may have slight difference about APIs, use of 1.10.x to train and freeze protobuf file is highly recommended. From test reports for a limited number of test cases, models rained and frozen by the following Tensorflow versions are known to work well: 1.4.x, 1.10.x, 1.13.x.
-- inputs	Input points of Tensorflow graph. (Required) -If there is only one input point, enclose the input point name in quotation marks, such as “input_point”. -If there are two or more input points, use a space to separate each input point “input_point_1 input_point_2 input_point_3”.
--input-size-list	Input size list for corresponding input points. (Required) - If there is only one input point, enclose the input point size list in quotation marks, and use a comma to separate the numbers in the size list, such as “224,224,3”. -If there are two or more input points, use a hashtag to separate each input size, such as “224,224,3#299,299,3#12,12”. CAUTION: the given input size of each input should or shouldn’t contain the batch size is determined by --size-with-batch.
--size-with-batch	Describes if the --input-size-list contains the highest batch dimension. (Optional) None (Default) means --input-size-list should be given without batch. Enclose the bool values in quotation marks, and use a comma

to separate the bool values. Take a graph with two input layer as an example, the three input layers with shape “?x224x224x3”, “11x88”, “10”, input-size list and size with batch could be given as:

A:

--input-size-list “224,224,3#11,88#10”

--size-with-batch “False,True,True”

B:

--input-size-list “224,224,3#88#10”

--size-with-batch “False,False,True”

-- outputs

Output points of Tensorflow graph. (Required)

-If there is only one output point, enclose the output point name in quotation marks, such as “output_point”.

-If there are two or more output points, use a space to separate each output point, such as “output_point_1 output_point_2 output_point_3”.

--mean-values

Mean values for Tensorflow quant models. (Optional)

Needs to be provided when converting Tensorflow quant models. Comma-separated list of doubles, each entry in the list should match an entry in '--inputs'.

--std-values

Standard values for Tensorflow quant models. (Optional)

Needs to be provided when converting Tensorflow quant models. Comma-separated list of doubles, each entry in the list should match an entry in '--inputs'.

--predef-file

Pre-define file to import some complex models. (Optional)

To support some control logic, provide a npz format predef-file.

Use np.savez(\'prd.npz', [placeholder name]=predefine_value) to generate the file. If the filename contains characters which are not supported in, use ‘map’ as a keyword to store a dict to map it to normal key value.

Import TFlite Arguments

-h Help file.

--output-model Acuity net Neural Network layer set output file. (Optional)

If not specified, the default uses the path where the tflite model

file is located, and the output filename is the same as the tflite

model.

CAUTION: For quant models, there will be an extra .quantize

file (quantize table) generated, after imported, DO NOT

quantize model again, just use imported model

inference/snapshot/export.

--output-data Acuity net Neural Network coefficient data output file.(optional)

If not specified, the default uses the path where the tflite model

file is located, and the output filename is the same as the tflite

model.

--model Neural Network model input filename. (Required)

CAUTION: Acuity uses the tflite schema commits as in link:

https://github.com/tensorflow/tensorflow/commits/

master/tensorflow/lite/schema/schema.fbs

Commit hash:

0c4f5dfea4ceb3d7c0b46fc04828420a344f7598.

Because the tflite schema may not compatible with each other,

tflite models in older or newer schema may not be imported

successfully.

--outputs Output points of tflite graph. (Optional, available from 5.11.0)

-If there is only one output point, enclose the output point

name in quotation marks, such as “output_point”.

-If there are two or more output points, use a space to

separate each output point, such as “output_point_1

output_point_2 output_point_3”.

Import Darknet Arguments

-h Help file.

--model Darknet model filename. (Required)

--weights Darknet weights filename. (Required)

--output-model Acuity net Neural Network layer set output file. (Optional)
If not specified, the default uses the path where the darknet model file is located.

--output-data Acuity net Neural Network coefficient data output
file.(optional)

If not specified, the default uses the path where the darknet model file is located.

Import Onnx Arguments

-h Help file.

--model model filename. (Required)
CAUTION: Supports onnx models in operator set 1-7.

-- inputs Input points of ONNX graph. (Optional)

--input-size-list Input size list for corresponding input points. (Optional)
None (Default): If there is only one input point, enclose the input point size list in quotation marks, and use a comma to separate the numbers in the size list, such as "224,224,3". If there are two or more input points, use hashtag to separate each input size, such as "224,224,3#299,299,3#12,12"
CAUTION: whether the given input size of each input should or shouldn't contain batch size is determined by --size-with-batch.

--size-with-batch Describe if the --input-size-list contain the highest batch dimension.(Optional, available from 5.11.0, March 2020)
None (Default): --input-size-list should be given without batch.Enclose the bool values in quotation marks, and use a hashtag to separate the bool values.

--input-dtype-list Input tensors dtype for corresponding input points.(Optional)
None (Default): Default input dtype is float. String values in ['float', 'int8', 'uint8', 'int16', 'uint16'].

--output-data Acuity net Neural Network coefficient data output file.
(Optional) If not specified, the default uses the path where The onnx model file is located.

--outputs Output points of ONNX graph. (Optional)

--output_model Acuity net Neural Network layer set output file. (Optional)

If not specified, the default uses the path where the onnx model file is located.

Import Pytorch Arguments

-h Help file.

--model Pytorch model filename. The pytorch model must created from torch.jit.trace(). (Optional)

CAUTION: Supports Pytorch 1.2

--inputs Model inputs.(Optional) Query from model (Default)

Only supports nodes which have one input tensor as input node.

--outputs Model outputs.(Optional) Query from model (Default).

Only supports nodes which have one output tensor as output node.

--input-size-list Input size list for corresponding input points. (Optional)

“None” (Default) If there is only one input point, enclose the input point size list in quotation marks, and use a comma to separate the numbers in the size list, such as “3,224,224”. If there are two or more input points, use a hashtag to separate each input size, such as “3,224,224#3,299,299#12,12”

CAUTION: the given input size of each input should NOT contain batch size.

--size-with-batch Describes if the --input-size-list contains the highest batch dimension. (Optional)

None (Default) means --input-size-list should be given without batch.

--output-model Acuity net Neural Network layer set output file.(Optional)

If not specified, the default uses the path where the Pytorch model file is located, and the output filename is the same as the Pytorch model.

--output-data Acuity net Neural Network coefficient data output file.

If not specified, the default uses the path where the Pytorch model file is located, and the output filename is

the same as the Pytorch model.

--config A json file that describes the pytorch model information.

(Optional) It can convert pytorch model only by specifying config file. Do not specify parameter ‘config’ if using parameter ‘pytorch_model’ to convert pytorch model.

Import Keras Arguments

-h Help file.

--model Keras model filename. (Required)

CAUTION: Support Keras model generated by Tensorflow 1.13.2.

--convert-engine Convert engine for Keras model. (Optional)

String value in [‘Keras’, ‘TFLite’] Keras (Default)

--inputs Input points of Keras graph. (Optional)

-If there is only one input point, enclose the input point name in quotation marks, such as “input_point”.

-If there are two or more input points, use a space to separate each input point “input_point_1 input_point_2 input_point_3”.

None (Default) means use all of the header node for import.

--input-size-list Input size list for corresponding input points. (Required)

None (Default) means let converter detect the input shapes. -If there is only one input point, enclose the input point size list in quotation marks, and use a comma to separate the numbers in the size list, such as “224,224,3”.

-If there are two or more input points, use a hashtag to separate each input size, such as “224,224,3#299,299,3#12,12”.

CAUTION: the given input size of each input should NOT contain the batch size.

--outputs Output points of Keras graph. (Optional)

None (Default) means use all of the tail node for import.

-If there is only one output point, enclose the output point name in quotation marks, such as “output_point”.

-If there are two or more output points, use a space to separate each output point, such as “output_point_1 output_point_2 output_point_3”.

--output-model Acuity net Neural Network layer set output file. (Optional)

If not specified, the default uses the path where the Keras model file is located, and the output filename is the same as the Keras model.

--output-data Acuity net Neural Network coefficient data output file. (Optional) If not specified, the default uses the path where the Keras model file is located, and the output filename is the same as the Keras model.

Generate Inputmeta

-h Help file.

--model Acuity model file input. (Required)

--input-meta-output Model input meta output path. (Optional) If not specified, put the generated input meta file in the same folder as Acuity model file.

--separated-database whether to generate more database for multi-inputs network. (Optional) False (Default) means generate one database in input_meta yaml file.

Quantize Arguments

-h Help file.

--model Network model input file. (Required)

--model-data Network coefficient input file. (Required)

--model-quantize Quantized tensor description file(quantize table), such as *.quantize etc. (Optional) Please specify this parameter if DTYPE is specified as ‘quantized’

--batch-size Integer value which specifies the batch size (number of images in a batch). (Optional) 100 (Default).
CAUTION: Set this value smaller if the RAM or GPU cannot support such a batch size.

--iterations Number of iterations to run, integer value. (Optional) 1 (Default) The iteration times for networks with cycle such as lstm, etc.

--device Specify the compute device. (Optional) String value in GPU or CPU.

--with-input-meta

Merge input meta into MODEL. (Optional)

Input meta provides descriptions about every input layer’

s dataset, pre-process, etc. -If there’s no input meta

information in MODEL, this parameter need to be pecified.

-If the input meta information is existed in MODEL, and this parameter is specified, use the specified input meta

to override.

--quantizer

Quantizer type. (Optional) Specify the quantizer to

quantize network tensors. String value options are:

asymmetric_affine (Default)

dynamic_fixed_point

perchannel_symmetric_affine

symmetric_affine

asymmetric_quantized (will be deprecated, use asymmetric_affine instead)

CAUTION: Quantizers without the prefix ‘perchannel_’

are per-layer quantizers. asymmetric_quantized will be

deprecate in a future release. asymmetric_quantized and

asymmetric_affine are equivalent, and asymmetric_affine now

replaces asymmetric_quantized.

--qtype

Quantizer data type. (Optional)

String value options are:

int8 (Default), **int16**, and **uint8**.

The following quantizer + qtype is supported:

asymmetric_affine + uint8

dynamic_fixed_point + int8/int16

perchannel_symmetric_affine + int8

symmetric_affine + int8

asymmetric_quantized + uint8

--hybrid

Setup a hybrid quantize network. (Optional)

Mutually exclusive with –rebuild. In the hybrid quantize

condition, please specify this parameter.

--rebuild (Optional)

Rebuild quantize table with a default quantize rules.

Mutually exclusive with --hybrid. In most scenarios, this

parameter should be given to let Acuity build a default quantize table.

--algorithm Quantization algorithm. (Optional) String value options are: **normal** (Default),**kl_divergence**, **moving_average.**

--moving-average-weight Moving average coefficient. (Optional) Positive float value. 0.01 (Default) Please specify this parameter if ALGORITHM is specified as ‘moving_average.’

--divergence-nbins KL divergence histogram nbins. (Optional) Positive integer value. **1024** (Default) Please specify this parameter if ALGORITHM is specified as ‘kl_divergence.’

Export Ovxlib Arguments

-h Help file.

--model Neural Network model input filename. (Required)

--model-data Neural Network coefficient data input filename. (Required)

--with-input-meta Merge input meta into MODEL. (Required)

Input meta contains the description about every input layer’s dataset, pre-process etc. If there is no input meta information in MODEL, this parameter needs to be specified. If the input meta information already exists in MODEL and this parameter is specified, use this specified input meta to override.

--model-quantize Quantized tensor description file (quantize table), such as *.quantize, etc. (Optional) Please specify this parameter if DTYPE is specified as ‘quantized’

-- output_path output filename or path. (Optional)

--optimize Optimize the exporting network according to the specified hardware configuration path or configuration name. (Optional)

none – no optimization

Default (Default) - If a configuration file or configuration name is not specified, the default export rule will be used to export application code. If --pack-* is given, please specify a configuration file path or configuration name instead of none

or

Default.

--dtype Export case to given data type. (Optional)

String value options are:

float (Default) - specify this value to export a float16 case.

float16 - specify this value to export a float16 case.

float32 - specify this value to export a float32 case.

quantized - specify this value to export a quantized case.

--save-fused-graph Whether to save fused graph. (Optional)

A fused graph is a network description file which contains the network structure of exported application code. It is useful at the debug level.

--pack-nbg-unify Pack binary graph for unify driver. Mutually exclusive with other --pack-*. -If this feature is enabled, three cases will be generated: a unify case, a nbg_unify case and an openvx case. The openvx case is in the nbg_unify case folder, and tnbg_unify and openvx cases share the same .nb file. -If no --pack-* is specified, only the unify case will be generated.

CAUTION: DOES NOT support graph with CPU node.

--pack-nbg-viplite Pack binary graph for VIPLite driver. (Optional)

Mutually exclusive with other --pack-*. -If this feature is enabled, two cases will be generated, a unify case and a nbg_viplite case. -If no --pack-* is specified, only the unify case will be generated.

CACUTION: DOES NOT support graph with CPU node.

--viv-sdk SDK directory.

-If one --pack-* is given, please specify the folder which contains the binary sdk for vSimulator. If VivanteIDE is installed, the SDK path may be something like /home/xxx/Verisilicon/VivanteIDEx.x.x/*cmdtools.

--build-platform (Optional) Build platform for generating network cases for nbg.

Value options are:

make (Default), consistent with the chosen pack parameter --pack-*.

--target-ideproject Target IDE environment for application code. (Optional)

Valid string values are:

linux64 (Default), **win32**

--batch-size Batch size for exported application code. (Optional)

Integer value in[1,+nan] **None** (Default)

CAUTION: if set '–batch-size', using the parameter set,

otherwise set the shape[0] in the input_meta file as batch-size.

Inference Arguments

-h Help file.

--model Network model input file in Acuity format, such as *.json *.yml etc. (Required)

--model-data Network coefficient input file in Acuity format, such as *data etc. (Required)

--model-quantize Quantized tensor description file (quantize table), such as *.quantize etc. (Optional) Please specify this parameter if DTYPE is specified as 'quantized'.

--batch-size Integer value which specifies the batch size (number of images in a batch). (Optional) 100 (Default).

CAUTION: Set this value smaller if the RAM or GPU cannot support such a batch size.

--iterations Running iterations, integer value. (Optional) 1 (Default) The iteration times for networks with cycle such as lstm, etc.

--with-input-meta Merge input meta into MODEL. (Optional) Input meta is description about every input layer's dataset, pre-process, etc. -If there is no input meta information in MODEL, this parameter needs to be specified. -If the input meta information exists in MODEL, and this parameter is specified, use the specified input meta to override.

--dtype Data type used for inference. (Optional) String values in ['float32', 'quantized'] **float32** (Default) - specify this value to inference floa32 model. **quantized** - specify this value to inference quantized model.

--postprocess Postprocess task. (Optional) Built-in post-process task value in ['print_topn', 'dump_result','classification_classic'] classification_classic(Default) ONLY ONE task could be chosen at the same time. Mutually exclusive with --postprocess-file Usage see 0

--postprocess-file Postprocess task configuration file. (Optional) According to POSTPROCESS, the user needs to create a yaml file manually or use pegasus to generate a postprocess-file to define the output tensors' post process tasks. One or more tasks could be given at the same time. This argument is mutually exclusive with –postprocess. For usage see Section 0

--output-dir Output directory of result files. (Optional)