密级状态：绝密(　　)　　秘密(　　)　　内部(　　)　　公开(√)

RKNN-Toolkit2 问题排查手册

（技术部，图形计算平台中心）

文件状态： [　] 正在修改 [√] 正式发布	当前版本：	1.5.0
	作　　者：	HPC
	完成日期：	2023-5-18
	审　　核：	熊伟
	完成日期：	2023-5-18

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd

更新记录

版本	修改人	修改日期	修改说明	核定人
1.4.2	HPC	2023-2-10	初稿	熊伟
1.5.0	HPC	2023-5-18	修改 whl 包命令规范	熊伟

目录

1 概述

以下文档如无特殊说明，则仅适用于 RKNN-Toolkit2。由此工具转换出的模型适用于 RK3566、RK3568、RK3588、RK3588S、RV1103、RV1106、RK3562 平台。

参考本文档时，请确认环境中的 RKNN-Toolkit2 版本，否则部分内容可能不适用。

2 RKNN-Toolkit2 和 RKNPU2 的版本关系

RKNN-Toollkit2 是基于 Python 语言实现的模型转换工具，可将其他训练框架导出的模型转为 RKNN 模型，并提供较为有限的推理接口，协助用户测试模型转换效果。工程链接为：https://github.com/rockchip-linux/rknn-toolkit2

RKNPU2 是板端的组件，提供 NPU 驱动，并基于 C 语言提供模型加载、模型推理等功能。相比 RKNN-Toolkit2 工具，RKNPU2 的 C API 推理接口用法更灵活、性能更优。工程链接为：https://github.com/rockchip-linux/rknpu2

RKNN-Toolkit2，RKNPU2 具有一致的版本号。不同版本之间可能存在不兼容问题，为了避免造成不必要的麻烦，推荐用户使用同一版本的 RKNN-Toolkit2、RKNPU2；版本的更新通常包含 bug 修复与性能优化，建议用户使用最新版本。

3 RKNN-Toolkit2 安装问题

本章主要覆盖常见的工具安装问题。

3.1 RKNN-Toolkit2 依赖的环境限制太严格，导致无法成功安装

在所有依赖库都已安装、但部分库的版本和要求不匹配时，可以尝试在安装指令后面加上--no-deps 参数，取消安装 python 库时的环境检查。如：

```
pip install rknn-toolkit2*.whl --no-deps
```

3.2 ONNX 依赖说明

RKNN-Toolkit2 的 ONNX 模型加载功能，依赖 ONNX。由于 ONNX 各版本之间的兼容性较好，目前没发现版本导致的问题。

推荐使用的 ONNX 版本是 1.10.0。

3.3 PyTorch 依赖说明

RKNN-Toolkit2 的 PyTorch 模型加载功能，依赖于 PyTorch。PyTorch 的模型分为浮点模型和已量化模型（包含 QAT 及 PTQ 量化模型）。对于 PyTorch 1.6.0 导出的模型，建议将 RKNN-Toolkit2 依赖的 PyTorch 版本降级至 1.6.0 以免出现加载失败的问题。对于已量化模型（QAT、PTQ），我们推荐使用 PyTorch 1.10 导出模型，并将 RKNN-Toolkit2 依赖的 PyTorch 版本升级至 1.10。另外在加载 PyTorch 模型时，建议导出原模型的 PyTorch 版本，要与 RKNN-Toolkit2 依赖的 PyTorch 版本一致。

推荐使用的 PyTorch 版本为 1.6.0、1.9.0 或 1.10 版本。

3.4 TensorFlow 依赖说明

RKNN-Toolkit2 的 TensorFlow 模型加载功能依赖于 TensorFlow。由于 TensorFlow 各版本之间的兼容性一般，其他版本可能会造成 RKNN-Toolkit2 模型加载异常，所以在加载 TensorFlow 模型时，建议导出原模型的 TensorFlow 版本，要与 RKNN-Toolkit2 依赖的 TensorFlow 版本一致。

对于 TensorFlow 版本引发的问题，通常会体现在 load_tensorflow 阶段，且出错信息会指向依赖的 TensorFlow 路径。

推荐使用的 TensorFlow 版本为 2.6.2。

3.5 RKNN-Toolkit2 安装包命名规则

以 1.4.0 版本的发布件为例，RKNN-Toolkit2 wheel 包命令规则如下：

```
rknn_toolkit2-1.4.0+22dcfef4-cp36-cp36m-linux_x86_64.whl
```

- rknn_toolkit2：工具名称。
- 1.4.0：版本号。
- 22dcfef4：提交号。
- cp<xx>-cp<xx>m：适用的 Python 版本，例如 cp36-cp36m 表示适用的 Python 版本是 3.6。
- linux_x86_64：系统架构。

请按照自己所用的操作系统、CPU 架构和 Python 版本安装对应的工具包，否则安装会失败。

3.6 RKNN-Toolkit2 是否有 ARM 版本

RKNN-Toolkit2 没有 ARM 版，如果需要在 ARM 上使用 Python 接口进行推理，可以安装 RKNN-Toolkit-lite2，该工具有 ARM 版。

3.7 bfloat16 依赖库安装不上

bfloat16 的依赖库安装出错：

```
    bfloat16.cc:2013:57: note:   expected a type, got ‘bfloat16’
    bfloat16.cc:2013:57: error: type/value mismatch at argument 2 in templa
e, class Functor> struct greenwaves::{anonymous}::UnaryUFunc’
    bfloat16.cc:2013:57: note:   expected a type, got ‘bfloat16’
    bfloat16.cc:2015:67: error: ‘>>’ should be ‘> >’ within a nested templa
        RegisterUFunc<BinaryUFunc<bfloat16, bfloat16, ufuncs::NextAfter>>(
                                                                          ^
    bfloat16.cc:2015:58: error: type/value mismatch at argument 1 in templa
e, class Functor> struct greenwaves::{anonymous}::BinaryUFunc’
        RegisterUFunc<BinaryUFunc<bfloat16, bfloat16, ufuncs::NextAfter>>(
                                                                         ^
    bfloat16.cc:2015:58: note:   expected a type, got ‘bfloat16’
    bfloat16.cc:2015:58: error: type/value mismatch at argument 2 in templa
e, class Functor> struct greenwaves::{anonymous}::BinaryUFunc’
    bfloat16.cc:2015:58: note:   expected a type, got ‘bfloat16’
    error: command 'gcc' failed with exit status 1
```

更换阿里源即可。

4 模型转换常用参数说明

本章节主要覆盖模型转换阶段常用参数的使用说明。

4.1 根据模型确定参数

模型转换时，config 和 build 接口会影响模型转换结果。load_onnx，load_tensorflow 指定输入输出节点，会影响模型转换结果。load_pytorch，load_tensorflow 指定输入的尺寸大小会影响模型转换结果。

可以参考以下基本思路进行模型转换：

1. 准备量化数据，提供 dataset.txt 文件。

2. 确定模型要使用的平台，如 RK3566、RV1106 等，并填写 config 中的 target_platform 参数。

3. 当输入是 3 通道的图像，且量化数据采用的是图片格式（如 jpg、png 格式）时，需要确认模型的输入是 RGB 还是 BGR，以决定 config 接口中 quant_img_RGB2BGR 参数的值。

4. 确认模型训练时候的归一化参数，以决定 config 接口中的 mean_values/std_values 参数的值。

5. 确认模型输入的尺寸信息，填入 load 接口相应参数中，如 load_pytorch 接口中的 input_size_list 参数。

6. 确认模型要量化比特数，以决定 config 接口中的 quantized_dtype 参数的值。**不对模型进行量化或加载的是已量化模型时可以忽略此步骤**。

7. 确认模型量化时使用的量化参数优化算法，以决定 config 接口中 quantized_algorithm 参数的值。**不对模型进行量化或加载已量化模型时可以忽略此步骤**。

8. 确认是否对模型进行量化，以决定 build 接口中 do_quantization 参数的值。选择对模型进行量化时，需要额外填写 build 接口中的 dataset 参数，指定量化矫正数据。

4.2 各平台对应的模型是否兼容

对于 target_platform 设置的平台参数，兼容性关系如下：

- RK3566、RK3568 平台使用的模型是相互兼容的。
- RK3588、RK3588S 平台使用的模型是相互兼容的。

- RV1103、RV1106 平台使用的模型是相互兼容的。

4.3 量化数据的格式及要求

量化数据的格式有两种选择，一种是图片格式（jpg，png），RKNN-Toolkit2 会调用 OpenCV 接口进行读取；另一种是 npy 格式，RKNN-Toolkit2 会调用 numpy 接口进行读取。

对于非 RGB/BGR 图片输入的模型，建议使用 numpy 的 npy 格式提供量化数据。

4.4 多输入模型 dataset.txt 文件的填写方式

模型量化需要用 dataset.txt 文件指定量化数据的路径。规则为一行作为一组输入，模型存在多输入时，多个输入写在同一行，并用空格隔开。

如单输入模型，使用两组量化数据：

```
sampleA.npy
sampleB.npy
```

如三个输入的模型，两组量化数据按如下方式填写：

```
sampleA_in0.npy sampleA_in1.npy sampleA_in2.npy
sampleB_in0.npy sampleB_in1.npy sampleB_in2.npy
```

4.5 确认 quant_img_RGB2BGR 参数

当采用图片（jpg，png）作为量化数据时，需要考虑设置 quant_img_RGB2BGR 参数。

模型采用 RGB 图片进行训练时，则 quant_img_RGB2BGR 参数设为 False 或不设置。且在使用 Python inference 接口或 RKNPU2 C API 进行推理时，输入 RGB 图片。

模型采用 BGR 图片进行训练时，则 quant_img_RGB2BGR 参数设为 True。但在使用 Python inference 接口或 RKNPU2 C API 进行推理时，同样需要输入 BGR 图片（quant_img_RGB2BGR 只会影响从量化校正集读入的图像）。

若量化数据采用 numpy 的 npy 格式，则建议不要使用 quant_img_RGB2BGR 参数，避免产生使用混乱的问题。

4.6 mean/std、quant_img_RGB2BGR 的计算顺序问题

因为 quant_img_RGB2BGR 只控制在量化过程中读取校正集图像时是否要进行转换通道，并不会影响其他的步骤。因此对于 RKNN-Toolkit2 的 inference 接口及 RKNPU2 C API，对输入数据都只先进行减均值（mean）、再除标准差（std）的操作，并没有通道转换的操作。

4.7 模型是非 3 通道输入或多输入时，mean/std 的设置问题

RKNN-Toolkit2 中 mean 和 std 的设置格式是一致的。以 mean 为例子。

假设输入有 N 个通道，则 mean_values 的值为[[channel_1, channel_2, channel_3, channel_4, ..., channel_n]]。

存在多输入时，则 mean_values 的值为[[channel_1, channel_2, channel_3, channel_4, ..., channel_n], [channel_1, channel_2, channel_3, channel_4, ..., channel_n]]。

4.8 量化参数矫正算法和量化图片数量的选取

RKNN-Toolkit2 中量化矫正算法 quantized_algorithm 参数提供三种算法进行参数矫正，分别为’normal’、’mmse’和’kl_divergence’，默认使用’normal’。Normal 为常规的量化参数矫正算法；而 MMSE 会迭代中间层的计算结果，对权重数值进行一定范围的裁剪，以获得更高的推理精度。使用 MMSE 不一定能提升量化精度，但相比 Normal 方式，量化时会占用更多的内存、耗费更长的模型转换时间；使用 KL_divergence 量化算法所用时间会比 Normal 多一些，但比 MMSE 会少很多，在某些场景下（feature 分布不均匀时）可以得到较好的改善效果。

建议先使用 Normal 算法，如果量化效果不佳，可尝试使用 MMSE 或 KL_divergence 算法。

使用 Normal 或 KL_divergence 算法时，推荐给出 50-200 组数据进行量化。使用 MMSE 量化时，推荐使用 20-50 组数据进行量化。

4.9 量化模型与非量化模型，推理时输入输出的差异

调用常规 RKNPU2 C API 时（指不使用 pass_through、zero_copy 的方式调用 C API），输入数据的数据类型（如 uint8 数据，float 数据）与模型的量化与否没有关系。输出数据的数据类型可以选择自动处理成 float32 格式，也可以选择直接输出模型推理结果，此时数据类型与输出节点的数据类型一致。使用 Python 推理接口会有点差异，具体关系如下表：

表 4-1 量化与非量化模型输入、输出关系表

模型量化后	Python 推理（rknn.inference）	C API 推理（rknn.run） （非 pass_through、zero_copy）
输入类型是否有限制	无限制。 rknn.inference 的输入为 numpy 数组，本身带有 data type 属性，该输入会自动转成 RKNN 模型需要的数据格式。	无限制。 rknn inputs 的 rknn_tensor_type 参数可以根据实际输入，指定 RKNN_TENSOR_FLOAT32、RKNN_TENSOR_FLOAT16、RKNN_TENSOR_INT8、RKNN_TENSOR_UIN8、RKNN_TENSOR_INT16。指定后，会将输入自动转成 RKNN 模型需要的数据格式。
输出类型是否变化	无变化。 无论模型量化与否，Python 的 inference 接口总是返回 float 类型输出。无法选择其他数据类型。	有变化。 RKNPU2 C API 的 rknn outputs attr，可以设置 want_float=1，得到 float 类型的输出。而量化后，可以设置 want_float=0，此时可以输出最后一个节点的原始输出数据，如 i8 量化时，输出 int8 数据。
输入 format 是否有变化（NCHW，NHWC）	无变化。 无论模型量化与否，rknn.inference 接口的 data_format 参数，可以根据需要可设置为 NCHW 或 NHWC。	无变化。 无论模型量化与否，rknn inputs 结构体的 rknn_tensor_format 参数，可以根据需要设置为 NCHW 或 NHWC。

4.10 是否存在在线编译的模式

RKNN-Toolkit2 只支持导出离线预编译的模型，不支持导出在线编译的模型（RKNN-Toolkit1 支持），因此并不存在离线预编译和在线编译的模式选择。

4.11 RKNN-Toolkit 转出来的 RKNN 模型可以在 RK3566 平台上使用吗？

不可以。

RKNN-Toolkit 转出来的 RKNN 模型适用于 RK1806 / RK1808 / RK3399Pro / RV1109 / RV1126 等平台；RK3566 平台需要用 RKNN-Toolkit2 转出来的 RKNN 模型。RKNN-Toolkit2 转出来的 RKNN 模型适用于 RK3566 / RK3568 / RK3588 / RK3588S / RV1103 / RV1106 / RK3562 等平台。

RKNN-Toolkit 工具的使用说明请参考以下工程：

https://github.com/rockchip-linux/rknn-toolkit

RKNN-Toolkit2 工具的使用说明请参考以下工程：

https://github.com/rockchip-linux/rknn-toolkit2

5 各框架模型加载问题

5.1 RKNN-Toolkit2 支持的深度学习框架和对应版本

请参考《Rockchip_User_Guide_RKNN_Toolkit2_CN》文档的 1.4 章节。

5.2 各框架的 OP 支持列表

RKNN-Toolkit2 对不同框架的支持程度有差异，详细信息可以参考以下目录中的 RKNNToolkit2_OP_Support 文档：

https://github.com/rockchip-linux/rknn-toolkit2/blob/master/doc/

5.3 Caffe 模型转换常见问题

5.4 Darknet 模型转换常见问题

5.5 ONNX 模型转换常见问题

5.5.1 转换时遇到模型 IR 版本过高的错误

具体的错误日志如下：

```
Your model ir_version is higher than the checker`s
```

RKNN-Toolkit2 1.4.0 及之前的版本支持 1.6.0～1.9.0 版本 ONNX 导出的模型，更新的 ONNX 导出的模型可能会报错。

5.5.2 加载模型时出现“Error parsing message”报错

转换 examples/onnx/resnet50v2 模型时，提示加载失败：

```
E load_onnx: Catch exception when loading onnx model: /rknn_resnet_demo/resnet50v2.onnx!
E load_onnx: Traceback (most recent call last):
E load_onnx:   File "rknn/api/rknn_base.py", line 1094, in rknn.api.rknn_base.RKNNBase.load_onnx
E load_onnx:   File "/usr/local/lib/python3.6/dist-packages/onnx/__init__.py", line 115, in load_model
E load_onnx:     model = load_model_from_string(s, format=format)
E load_onnx:   File "/usr/local/lib/python3.6/dist-packages/onnx/__init__.py", line 152, in load_model_from_string
E load_onnx:     return _deserialize(s, ModelProto())
E load_onnx:   File "/usr/local/lib/python3.6/dist-packages/onnx/__init__.py", line 95, in _deserialize
E load_onnx:     decoded = cast(Optional[int], proto.ParseFromString(s))
E load_onnx: google.protobuf.message.DecodeError: Error parsing message
```

原因可能是 resnet50v2.onnx 模型损坏导致（如没下载全），需要重新下载该模型，并确保其 MD5 值正确，如：

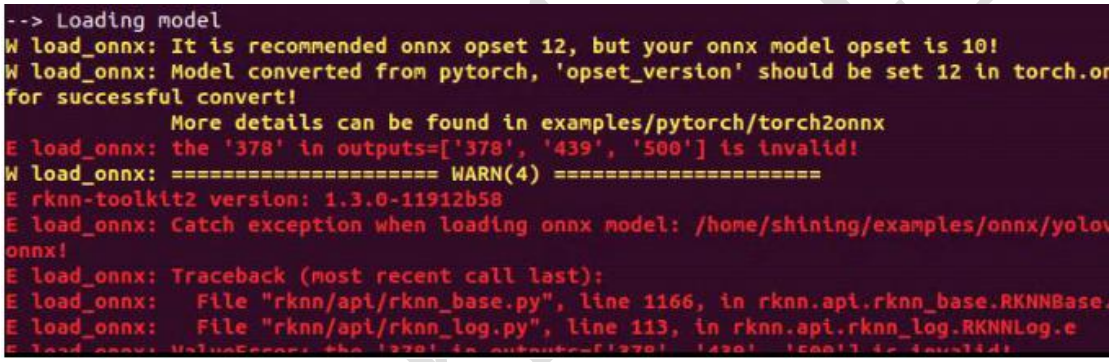
```
22ed6e6a8fb9192f0980acca0c941414    resnet50v2.onnx
```

5.5.3 是否支持动态的输入 shape

RKNN-Toolkit2 不支持动态的输入 shape，比如 onnx 输入维度为[-1, 3, -1, -1]，表示 batch、height 和 width 维度是不固定的，这种情况是不支持的。

5.5.4 自定义输出节点时报错

load_onnx 时传入 outputs 参数进行模型的裁剪，但报如下错误：



日志提示输出节点 378 是无效的，因此 outputs 参数需设置正确的输出节点名称。

5.6 Pytorch 模型转换常见问题

5.6.1 加载 Pytorch 模型时出现 torch._C 没有_jit_pass_inline 属性的错误

错误日志如下：

```
'torch._C' has no attribute '_jit_pass_inline'
```

请将 PyTorch 升级到 1.6.0 或之后的版本。

5.6.2 Pytorch 模型的保存格式

目前只支持 torch.jit.trace 导出的模型。torch.save 接口仅保存权重参数字典，缺乏网络结构信息，无法被正常导入并转成 RKNN 模型。

5.6.3 转换时遇到 PytorchStreamReader 失败的错误

详细错误如下：

```
E Catch exception when loading pytorch model: ./mobilenet0.25_Final.pth!
E Traceback (most recent call last):
……
E       cpp_module = torch._C.import_ir_module(cu, f, map_location, extra_files)
E RuntimeError: [enforce fail at inline container.cc:137]. PytorchStreamReader failed reading zip archive: faild finding central directory frame #0 ……
```

出错原因是输入的 PyTorch 模型没有网络结构信息。

通常 pth 只有权重，并没有网络结构信息。对于已保存的模型权重文件，可以通过初始化对应的网络结构，再使用 net.load_state_dict 加载 pth 权重文件。最后通过 torch.jit.trace 接口将网络结构和权重参数固化成一个 pt 文件。得到 torch.jit.trace 处理过以后的 pt 文件，就可以用 rknn.load_pytorch 接口将其转为 RKNN 模型。

5.6.4 转换时遇到 KeyError 的错误

错误日志如下：

```
E Traceback (most recent call last):
……
E KeyError: 'aten::softmax'
```

出现形如 KeyError: 'aten::xxx' 的错误信息时，表示该算子当前版本还不支持。RKNN-Toolkit2 在每次版本升级时都会修复此类 bug，请使用最新版本的 RKNN-Toolkit2。

5.6.5 转换时遇到“Syntax error in input! LexToken(xxx)”的错误

错误日志如下：

```
WARNING: Token 'COMMENT' defined, but not used
WARNING: There is 1 unused token
!!!!! Illegal character '"'
Syntax error in input! LexToken(NAMED_IDENTIFIER, 'fc', 1, 27)
!!!!! Illegal character '"'
```

该错误的原因有很多种，请按照以下顺序排查：

1）未继承 torch.nn.module 创建网络。请继承 torch.nn.module 基类来创建网络，然后再

用 torch.jit.trace 生成 pt 文件。

2）RKNN-Toolkit2 1.4.0 或之后的版本，torch 建议使用 1.6.0 或 1.9.0，1.10.0 版本。

5.7 TensorFlow 模型转换常见问题

5.7.1 Tensorflow1.x 模型报错

使用 rknn.load_tensorflow API 加载 tensorflow1.x 模型如出现报错提示：

```
E load_tensorflow: Catch exception when loading tensorflow model: ./yolov3_mobilenetv2.pb!
E load_tensorflow: Traceback (most recent call last):
E load_tensorflow: File "/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/importer.py", line 427, in import_graph_def
E load_tensorflow: graph._c_graph, serialized, options) # pylint: disable=protected-access
E load_tensorflow: tensorflow.python.framework.errors_impl.InvalidArgumentError: Node 'MobilenetV2/expanded_conv/depthwise/BatchNorm/cond/Switch_1' expects to be colocated with unknown node 'MobilenetV2/expanded_conv/depthwise/BatchNorm/moving_mean'
E load_tensorflow: During handling of the above exception, another exception occurred:
E load_tensorflow: Traceback (most recent call last):
E load_tensorflow: File "rknn/api/rknn_base.py", line 990, in rknn.api.rknn_base.RKNNBase.load_tensorflow
E load_tensorflow: File "rknn/base/convertor/tensorflow2onnx/tf2onnx/convert.py", line 589, in rknn.base.convertor.tensorflow2onnx.tf2onnx.convert.from_graph_def
E load_tensorflow: File "rknn/base/convertor/tensorflow2onnx/tf2onnx/convert.py", line 590, in rknn.base.convertor.tensorflow2onnx.tf2onnx.convert.from_graph_def
E load_tensorflow: File "rknn/base/convertor/tensorflow2onnx/tf2onnx/convert.py", line 591, in rknn.base.convertor.tensorflow2onnx.tf2onnx.convert.from_graph_def
E load_tensorflow: File "rknn/base/convertor/tensorflow2onnx/tf2onnx/convert.py", line 592, in rknn.base.convertor.tensorflow2onnx.tf2onnx.convert.from_graph_def
E load_tensorflow: File "/usr/local/lib/python3.6/dist-packages/tensorflow/python/util/deprecation.py", line 507, in new_func
E load_tensorflow: return func(*args, **kwargs)
E load_tensorflow: File "/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/importer.py", line
```

```
431, in import_graph_def

E load_tensorflow: raise ValueError(str(e))

E load_tensorflow: ValueError: Node 'MobilenetV2/expanded_conv/depthwise/BatchNorm/cond/Switch_1'

expects to be colocated with unknown node 'MobilenetV2/expanded_conv/depthwise/BatchNorm/moving_mean'
```

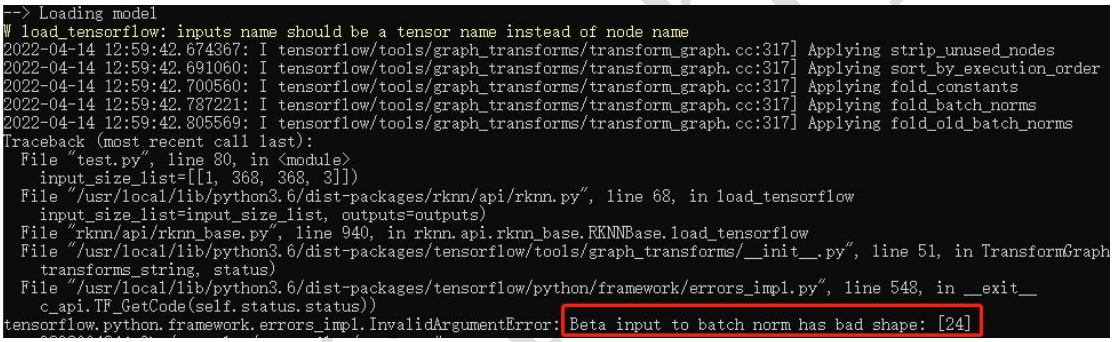
建议：

1）如当前安装的是 1.x 的 TensorFlow，请安装 2.x 的 TensorFlow。

2）更新 RKNN-Toolkit2 / RKNPU2 至最新版本。

5.7.2 TransformGraph 类似的报错

TensorFlow 的模型转成 RKNN 时报错：



原因：

1）该模型直接调用 TensorFlow 原生的 TransformGraph 类进行优化时，也会报上面的错误（RKNN-Toolkit2 里同样会调用 TransformGraph 进行优化，因此也会报同样的错误）。

2）猜测是模型生成时的 TensorFlow 版本与目前安装的版本已经不兼容了。

建议：

使用 1.14.0 的 TensorFlow 版本重新生成该模型，或者寻找其他框架的同类型模型。

5.7.3 “Shape must be rank 4 but is rank 0”报错

加载 pb 模型时：

```
rknn.load_tensorflow(tf_pb='./model.pb',

    inputs=["X","Y"],
```

```
        outputs=['generator/xs'],
        input_size_list=1, INPUT_SIZE, INPUT_SIZE, 3)
```

会产生报错：

```
E load_tensorflow: Catch exception when loading tensorflow model: ./model.pb!
E load_tensorflow: Traceback (most recent call last):
E load_tensorflow: File "/usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/importer.py", line 427, in import_graph_def
E load_tensorflow: graph._c_graph, serialized, options) # pylint: disable=protected-access
E load_tensorflow: tensorflow.python.framework.errors_impl.InvalidArgumentError: Shape must be rank 4 but is rank 0 for 'generator/conv2d_3/Conv2D' (op: 'Conv2D') with input shapes: [], [7,7,3,32].
```

原因可能是该模型是多输入模型，input_size_list 没按规范填写，可以参考 examples/functions/multi_input_test 里的以下用法：

```
rknn.load_tensorflow(tf_pb='./conv_128.pb',
        inputs=['input1', 'input2', 'input3', 'input4'],
        outputs=['output'],
        input_size_list=1, 128, 128, 3], [1, 128, 128, 3], [1, 128, 128, 3], [1, 128, 128, 1)
```

5.8 加载模型出错时的排查步骤

首先确认原始深度学习框架是否可以加载该模型并进行正确的推理。

其次请将 RKNN-Toolkit2 升级到最新版本。如果模型有 RKNN-Toolkit2 不支持的层（或 OP），通过打开调试日志开关，在日志中可以看到哪一个算子是 RKNN-Toolkit2 不支持的。

如果第一步不通过，请先检查原始模型是否有问题；如果升级到最新版本的工具后仍无法转换，或提示有算子不支持，请将所使用的工具的版本和出现转换问题时的详细日志反馈给瑞芯微 NPU 开发团队。

5.9 “Please call rknn.config first”报错提示

转换模型时提示：

```
E load_tflite: Please call rknn.config first!
W load_tflite: ======================= WARN(1) =======================
E rknn-toolkit2 version: 1.3.0-11912b58
E load_tflite: Catch exception when loading tflite model: point_history_classifier.tflite!
E load_tflite: Traceback (most recent call last):
E load_tflite:     File "rknn/api/rknn_base.py", line 1468, in rknn.api.rknn_base.RKNNBase.load_tflite
E load_tflite:     File "rknn/api/rknn_base.py", line 600, in rknn.api.rknn_base.RKNNBase._create_ir_and_inputs_meta
E load_tflite:     File "rknn/api/rknn_log.py", line 113, in rknn.api.rknn_log.RKNNLog.e
E load_tflite: ValueError: Please call rknn.config first!
```

提示“Please call rknn.config first”，表示 rknn.config 接口没有被调用，只需在加载模型之前添加 rknn.config 接口的调用即可。

6 模型量化问题

6.1 量化简介

模型量化后将使用更低的精度（如 int8/int16）保存模型的权重信息，部署后可减少模型的内存空间占用，加快模型推理速度。

RKNN-Toolkit2 目前对量化模型的支持主要有以下两种形式：

- RKNN-Toolkit2 根据用户提供的量化数据集，对加载的浮点模型进行量化，生成量化的 RKNN 模型。
  - 支持的量化精度类型：int8，int16。
  - 量化方式：训练后静态量化。
- 由深度学习框架导出量化模型，RKNN-Toolkit2 加载并利用已有的量化信息，生成量化的 RKNN 模型。
  - 支持的深度学习框架（括号内为主要支持版本，请尽量使用对应版本生成的量化模型）：PyTorch（v1.9.0 或 v1.10.0）、TensorFlow、TFLite。
  - 支持的量化精度类型：int8、uint8。
  - 量化方式：训练后量化，量化感知训练（QAT）。

6.2 QAT 与 PTQ 量化方式

- 训练后量化（PTQ 量化，Post train quantization）

RKNN-Toolkit2 加载用户训练好的浮点模型，然后根据 config 接口指定的量化方法和用户提供的校准数据集（训练数据或验证数据的一个小子集，大约 50~200 张）计算模型中网络层需要的量化参数。

RKNN-Toolkit2 目前支持的量化方法：

- asymmetric_quantized-8（默认量化方法）

这是 TensorFlow 支持的训练后量化算法，也是 Google 推荐的。根据论文《Quantizing deep convolutional networks for efficient inference: A whitepaper》的描述，这种量化方式对精度的

损失最小。

其计算公式如下：

$$quant = round\left(\frac{float\_num}{scale}\right) + zero\_point$$
$$quant = cast\_to\_bw$$

其中 quant 代表量化后的数，float_num 代表浮点数，scale 表示缩放系数（float32 类型），zero-points 代表实数为 0 时对应的量化值（int32 类型），最后把 quant 饱和到[range_min, range_max]，目前只支持 int8 类型，所以 range_max 等于 128，range_min 等于-127。

对应的反量化公式如下：

$$\text{float\_num} = scale(quant - zero\_point)$$

- 量化感知训练（QAT，quantization aware training）

通过量化感知训练可以得到一个带量化权重的模型。RKNN-Toolkit2 目前支持 TensorFlow 和 PyTorch 这两种框架量化感知训练得到的模型。量化感知训练技术细节请参考如下链接：

TensorFlow：https://www.tensorflow.org/model_optimization/guide/quantization/training

PyTorch：https://pytorch.org/blog/introduction-to-quantization-on-pytorch/

这种方法要求用户使用原始框架训练（或 fine tune）得到一个量化模型，接着使用 RKNN-Toolkit2 导入该量化模型（模型转换时需要在 rknn.build 接口设置 do_quantization=False）。此时 RKNN-Toolkit2 将使用模型自身的量化参数，因此理论上几乎不会有精度损失。

注：RKNN-Toolkit2 也支持 PyTorch，TensorFlow，TensorFlow Lite 的训练后量化模型，模型转换方法与量化感知训练模型一样，调用 build 接口时 do_quantization 要设置成 False。

6.3 量化对模型体积的影响

分两种情况，当导入的模型是量化的模型时，do_quantization=False 会使用该模型里面的量化参数。当导入的模型是浮点的模型时，do_quantization=False 不会做量化的操作，但是会把权重从 float32 转成 float16，这块几乎不会有精度损失。这两种情况都减少了模型权重的体积，从而使得整个模型占用空间变小。

6.4 量化数据的作用

RKNN-Toolkit2 在量化过程中，会根据量化数据集，计算推理结果所需要的量化参数。

基于该原因，校准数据集里的数据最好是从训练集或验证集中取一个有代表性的子集（该子集足够覆盖验证集的数据的分布范围即可），建议数量在 50~200 张之间。

如果校准数据集选择的不合适，会导致量化后的数据分布范围与验证集相差较多，那模型的量化精度就会比较差。

6.5 模型量化时，图片是否需要和模型输入的尺寸一致

不需要。RKNN-Toolkit2 会自动对这些图片进行缩放处理。但是缩放操作也可能会使图片信息发生改变，对量化精度产生一定影响，所以最好使用尺寸相近的图片。

6.6 量化校正集是否需要根据 rknn_batch_size 参数进行修改

不需要。rknn.build 的 rknn_batch_size 参数只会修改最后导出的 rknn 模型的 batch 维（由 1 改为 rknn_batch_size），并不会影响量化阶段的流程，因此量化校正集还是按照 batch 为 1 的方式来设置即可。

6.7 模型量化时，程序运行一段时间后被 kill 掉或程序卡住

在模型量化过程中，RKNN-Toolkit2 会申请较多的系统内存，有可能造成程序被 kill 掉或卡住。

解决方法：增加 PC 内存或增大虚拟内存。

7 模型转换问题

7.1 常见转换 bug 报错的问题

如遇到如下类似转换报错，很可能是由于当前版本存在 bug，可尝试将 RKNN-Toolkit2 更新至最新版本。

7.1.1 infer_shapes 类似错误

```
(op_type:Mul, name:Where_2466_mul): Inferred elem type differs from existing elem type: (FLOAT) vs (INT64)
E build: Catch exception when building RKNN model!
E build: Traceback (most recent call last):
E build: File "rknn/api/rknn_base.py", line 1555, in rknn.api.rknn_base.RKNNBase.build
E build: File "rknn/api/graph_optimizer.py", line 5409, in rknn.api.graph_optimizer.GraphOptimizer.run
E build: File "rknn/api/graph_optimizer.py", line 5123, in rknn.api.graph_optimizer.GraphOptimizer._fuse_ops
E build: File "rknn/api/ir_graph.py", line 180, in rknn.api.ir_graph.IRGraph.rebuild
E build: File "rknn/api/ir_graph.py", line 140, in rknn.api.ir_graph.IRGraph._clean_model
E build: File "rknn/api/ir_graph.py", line 56, in rknn.api.ir_graph.IRGraph.infer_shapes
E build: File "/home/anaconda3/envs/rk2/lib/python3.6/site-packages/onnx/shape_inference.py", line 35, in infer_shapes
E build: inferred_model_str = C.infer_shapes(model_str, check_type)
E build: RuntimeError: Inferred elem type differs from existing elem type: (FLOAT) vs (INT64)
```

或：

```
E build: Traceback (most recent call last):
E build: File "rknn/api/rknn_base.py", line 1643, in rknn.api.rknn_base.RKNNBase.build
E build: File "rknn/api/graph_optimizer.py", line 6256, in rknn.api.graph_optimizer.GraphOptimizer.fuse_ops
E build: File "rknn/api/ir_graph.py", line 285, in rknn.api.ir_graph.IRGraph.rebuild
E build: File "rknn/api/ir_graph.py", line 149, in rknn.api.ir_graph.IRGraph._clean_model
E build: File "rknn/api/ir_graph.py", line 62, in rknn.api.ir_graph.IRGraph.infer_shapes
E build: File "/usr/local/lib/python3.6/dist-packages/onnx/shape_inference.py", line 35, in infer_shapes
```

E build: inferred_model_str = C.infer_shapes(model_str, check_type)

E build: RuntimeError: Inferred shape and existing shape differ in rank: (0) vs (3)

或：

```
(op_type:ReduceMax, name:ReduceMax_18): Inferred shape and existing shape differ in rank: (3) vs (0)
E build: Catch exception when building RKNN model!
E build: Traceback (most recent call last):
E build:   File "rknn/api/rknn_base.py", line 1638, in rknn.api.rknn_base.RKNNBase.build
E build:   File "rknn/api/graph_optimizer.py", line 5499, in rknn.api.graph_optimizer.GraphOptimizer.fold_constant
E build:   File "rknn/api/ir_graph.py", line 750, in rknn.api.ir_graph.IRGraph.make_model
E build:   File "rknn/api/ir_graph.py", line 62, in rknn.api.ir_graph.IRGraph.infer_shapes
E build:   File "/home/wenqiluo/anaconda3/envs/rknn/lib/python3.6/site-packages/onnx/shape_inference.py", line 35, in infer_shapes
E build:     inferred_model_str = C.infer_shapes(model_str, check_type)
E build: RuntimeError: Inferred shape and existing shape differ in rank: (3) vs (0)
```

7.1.2 _p_fuse_two_mul 类似错误

E build: Catch exception when building RKNN model!

E build: Traceback (most recent call last):

E build: File "rknn/api/rknn_base.py", line 1643, in rknn.api.rknn_base.RKNNBase.build

E build: File "rknn/api/graph_optimizer.py", line 6197, in rknn.api.graph_optimizer.GraphOptimizer.fuse_ops

E build: File "rknn/api/graph_optimizer.py", line 204, in rknn.api.graph_optimizer._p_fuse_two_mul

E build: ValueError: non-broadcastable output operand with shape () doesn't match the broadcast shape (3,2)

7.1.3 “Segmentation fault”类似错误

如 picodet 模型转换报错：

```
root@8d7572b2aa6b:/test/pytorch/picodet# python test_hpc203.py
W __init__: Verbose file path is invalid, debug info will not dump to file.
--> config model
pre-process config done
--> Loading model
Load pytorch model done
--> Building model
W build: The output shape [1, 6] of model is wrong, fix it to [11, 6]!
I _fold_constant remove nodes = ['Shape_0', 'Gather_4', 'Shape_1', 'Gather_6', 'Unsqueeze_0', 'Concat_8', 'Cast_3', 'ReduceMin_0', 'Unsqueez
Segmentation fault (core dumped)
```

7.1.4 _p_fuse_mul_into_conv 类似错误

```
I     remove_invalid_reshape: remove node = ['Mul_796_0_expand0', 'Mul_798_0_expand0']
I     fuse_two_reshape: remove node = ['Mul_798_0_expand1']
E build: Catch exception when building RKNN model!
E build: Traceback (most recent call last):
E build:   File "rknn/api/rknn_base.py", line 1643, in rknn.api.rknn_base.RKNNBase.build
E build:   File "rknn/api/graph_optimizer.py", line 6197, in rknn.api.graph_optimizer.GraphOptimizer.fuse_ops
E build:   File "rknn/api/graph_optimizer.py", line 344, in rknn.api.graph_optimizer._p_fuse_mul_into_conv
E build: ValueError: non-broadcastable output operand with shape (1,258,1,256) doesn't match the broadcast shape (80256,258,1,256)
```

7.2 怎么判断各个框架的算子 RKNN 是否支持

直接进行模型的转换，如果不支持会有相关提示。

7.3 转换时提示 Expand 算子不支持

建议：

1）新版本已经支持 CPU 的 Expand，可尝试更新 RKNN-Toolkit2 / RKNPU2 至最新版本。

2）修改模型，采用 repeat 算子来替代 expand 算子。

7.4 转换时提示“Meet unsupported dims in reducesum”

模型转换出现 Meet unsupported dims in reducesum, dims: 6，具体如下：

```
D RKNN: [14:54:19.434] >>>>>> start: N4rknn17RKNNInitCastConstE
D RKNN: [14:54:19.434] <<<<<<<< end: N4rknn17RKNNInitCastConstE
D RKNN: [14:54:19.434] >>>>>> start: N4rknn20RKNNMultiSurfacePassE
D RKNN: [14:54:19.434] <<<<<<<< end: N4rknn20RKNNMultiSurfacePassE
D RKNN: [14:54:19.434] >>>>>> start: N4rknn14RKNNTilingPassE
D RKNN: [14:54:19.434] <<<<<<<< end: N4rknn14RKNNTilingPassE
D RKNN: [14:54:19.434] >>>>>> start: N4rknn23RKNNProfileAnalysisPassE
D RKNN: [14:54:19.434] <<<<<<<< end: N4rknn23RKNNProfileAnalysisPassE
D RKNN: [14:54:19.434] >>>>>> start: OpEmit
E RKNN: [14:54:19.438] Meet unsupported dims in reducesum, dims: 6
Aborted (core dumped)
```

目前 RKNN 不支持 6 维的 OP，大多数情况下只支持 4 维，其他维度的，会有一些限制。

7.5 因 NonMaxSuppression / TopK 等后处理 Op 导致转换报错

1）NonMaxSuppression / TopK 等后处理 Op，RKNN 目前不支持。

2）可以将图的后处理子图部分移除，如：

```
rknn.load_onnx(model='picodet_xxx.onnx', outputs=['concat_4.tmp_0', 'tmp_16'])
```

3）移除的子图在 cpu 端另行进行处理。

7.6 “invalid expand shape”类似报错

例如 rvm_mobilenetv3_fp32.onnx 转换时出现以下报错：

```
2022-08-17 13:36:50.477462084 [E:onnxruntime:, sequential_executor.cc:333 Execute] Non-zero status code returned while running Expand node. Name:'Expand_294' Status Message: invalid expand shape
E build: Catch exception when building RKNN model!
E build: Traceback (most recent call last):
E build: File "rknn/api/rknn_base.py", line 1638, in rknn.api.rknn_base.RKNNBase.build
E build: File "rknn/api/graph_optimizer.py", line 5529, in rknn.api.graph_optimizer.GraphOptimizer.fold_constant
E build: File "rknn/api/session.py", line 69, in rknn.api.session.Session.run
E build: File "/home/cx/work/tools/Anaconda3/envs/rknn/lib/python3.8/site-packages/onnxruntime/capi/onnxruntime_inference_collection.py", line 124, in run
E build: return self._sess.run(output_names, input_feed, run_options)
E build: onnxruntime.capi.onnxruntime_pybind11_state.InvalidArgument: [ONNXRuntimeError] : 2 : INVALID_ARGUMENT : Non-zero status code returned while running Expand node. Name:'Expand_294' Status Message: invalid expand shape
```

原因：

1）因为 downsample_ratio 的输入值会改变模型中间 feature 的 size，所以说这种图本质上是动态图。

2）RKNN 目前不支持动态图。

建议：

可以将 downsample_ratio 固化为常量（不要作为变量传入），这样 RKNN 就可以支持。

7.7 output_tensor_type 报错提示

模型转换时有如下报错：

```
W __init__: rknn-toolkit2 version: v1.2.3-b4-ac69c24
--> Config model
Traceback (most recent call last):
  File "test.py", line 240, in <module>
    rknn.config(mean_values=[[0, 0, 0]], std_values=[[255, 255, 255]], output_te
nsor_type='int8')
TypeError: config() got an unexpected keyword argument 'output_tensor_type'
```

原因是新版本 RKNN-Toolkit2 已经没有 output_tensor_type 参数，删除 output_tensor_type 该配置参数即可。

7.8 mean_values 报错提示

设置 mean/std 为：

```
rknn.config(mean_values=[128, 128, 128], std_values=[128, 128, 128])
```

时转换模型报错：

```
--> Loading model

transpose_input for input_1: shape must be rank 4, ignored

E load_tflite: The len of mean_values ([128, 128, 128]) for input 0 is wrong, expect 32!
```

原因是可能是模型的输入不是 3 通道图像数据（例如输入 shape 是 1x32，非图像数据），此时：

1）需要根据输入通道个数来设置 mean_values / mean_values。

2）如果模型不需要指定 mean/std，rknn.config 可以不设置 mean_values / std_values（mean/std 一般只对图像输入有效）。

7.9 模型存在 4 维以上 Op 时报错（如 5 维或 6 维）

当模型存在 4 维以上 Op 时（如 5 维或 6 维），会有如下报错：

```
E build: Catch exception when building RKNN model!
E build: Traceback (most recent call last):
E build:   File "rknn/api/rknn_base.py", line 1580, in rknn.api.rknn_base.RKNNBa
se.build
E build:   File "rknn/api/rknn_base.py", line 341, in rknn.api.rknn_base.RKNNBas
e._generate_rknn
E build:   File "rknn/api/rknn_base.py", line 307, in rknn.api.rknn_base.RKNNBas
e._build_rknn
E build: IndexError: vector::_M_range_check: __n (which is 4) >= this->size() (w
hich is 4)
```

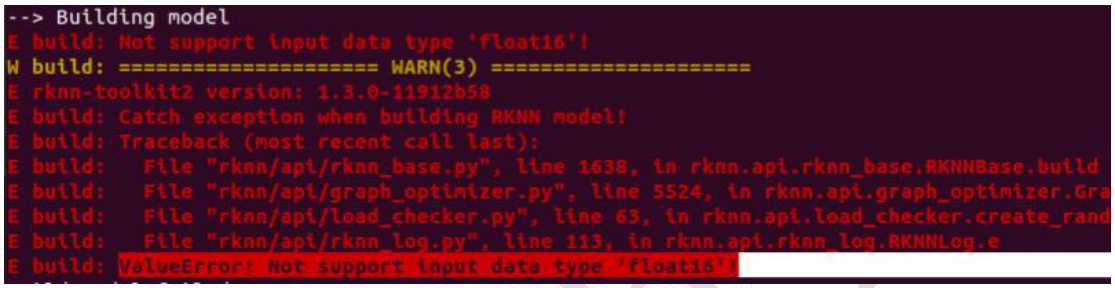
RKNN 目前暂不支持 4 维以上的 OP，可以手工将这些节点去掉。

7.10 Toolkit2 支持动态卷积吗？

RKNN-Toolkit2 以及 RKNPU2 暂不支持动态卷积。

7.11 “Not support input data type 'float16'”报错

Pytorch 训练的权重类型为 float16 的模型，在转换 RKNN 时出现以下报错：



目前 RKNN-Toolkit2 还还不支持 float16 的权重类型的 Pytorch 模型，需改为 float32。

7.12 动态图相关报错

转换模型时，如果出现以下类似报错：

```
E build: ValueError: The Op of 'NonZero' is not support! it will cause the graph to be a dynamic graph!
```

说明该 OP 会导致模型为动态图，需要手工修改模型，用其他 OP 替换或将其移除。

8 模拟器推理及连板推理、板端推理的说明

8.1 术语说明

模拟器推理：RKNN-Toolkit2 在 Linux x86_64 平台提供模拟器功能，可以在没有开发版的情况下进行模型推理，获取推理结果。（该功能输出结果未必与连板或板端一致，更推荐使用连板推理或板端推理）。

连板推理：指在开发板已连接 PC 的情况下，调用 RKNN-Toolkit2 的 Python API 推理模型，获取推理结果。开发板的连接方法，请参考 Rockchip_User_Guide_RKNN_Toolkit2_CN 文档的 3.22 小节)。

板端推理：指在开发板上调用 RKNPU2 的 C API 接口推理模型，获取推理结果。

8.2 模拟器推理结果与连板推理结果不一致

发生此情况时，可能意味着板端的结果不正确。

由于硬件和驱动的差异，模拟器不保证可以和板端获取一模一样的结果。但如果差异实在太大，则大概率是板端驱动 Bug 导致，可以将问题反馈给 RK 的 NPU 团队进行分析和 Debug。

8.3 连板推理的工作原理

使用连板推理时，RKNN-Toolkit2 会在后台启动一个 npu_transfer_proxy 的进程，此进程会与板端的 rknn-server 进行通信，通信时会将模型、模型的输入由 PC 端传至板端，随后调用 RKNPU2 C API 进行模型推理，板端推理完成后将结果回传至 PC 端。

8.4 连板推理与板端推理结果有差异

连板推理是基于 RKNPU2 的 C API 接口实现的，理论上连板推理结果会与 RKNPU2 C API 推理结果一致。当这两者出现较大差异时，请确认输入的预处理、数据类型、数据的排布方式（NCHW，NHWC）是否有差异。

需指出，如差异很小且发生在小数点后 3 位及之后的数值上，则属于正常现象。差异可

能产生在使用不同的库读取图片、转换数据类型等步骤上。

8.5 板端推理的速度比连板推理更快

由于连板推理存在额外的数据拷贝、传输过程，会导致连板推理的性能不如板端的 RKNPU2 C API 推理性能。因此，NPU 实际推理性能以 RKNPU2 C API 的推理性能为准。

8.6 涉及连板调试、连板推理功能时，获取详细的错误日志

连板调试、连板推理时，模型的初始化、推理等操作主要在开发板上完成，此时日志信息主要产生在板端上。

为了获取具体的板端调试信息，可以通过串口进入开发板操作系统。然后执行以下两条命令设置获取日志的环境变量。保持串口窗口不要关闭，再进行连板调试，此时板端的错误信息就会显示在串口窗口上：

```
export RKNN_LOG_LEVEL=5
restart_rknn.sh
```

9 模型评估常见问题

9.1 量化模型精度不及预期

参考 RKNN-Toolkit2 使用说明文档《Rockchip_User_Guide_RKNN_Toolkit2_CN.pdf 的 3.6 章节。

9.2 支持哪些框架的已量化模型

RKNN-Toolkit2 1.4 及之后的版本支持 TensorFlow、TensorFlow Lite 和 PyTorch 框架的已量化模型。

9.3 连板调试时，连接设备失败

连板精度分析（rknn.accuracy_analysis）时出现如下报错：

```
E accuracy_analysis: Connect to Device Failure (-1)
E accuracy_analysis: Catch exception when init runtime!
E accuracy_analysis: Traceback (most recent call last):
E accuracy_analysis: File "rknn/api/rknn_base.py", line 2001, in rknn.api.rknn_base.RKNNBase.init_runtime
E accuracy_analysis: File "rknn/api/rknn_runtime.py", line 194, in rknn.api.rknn_runtime.RKNNRuntime.__init__
E accuracy_analysis: File "rknn/api/rknn_platform.py", line 331, in rknn.api.rknn_platform.start_ntp_or_adb
```

或连板推理（rknn.inference）时出现如下报错：

```
I target set by user is: rk3568
I Starting ntp or adb, target is RK3568
I Device [0c6a9900ef4871e1] not found in ntb device list.
I Start adb...
I Connect to Device success!
I NPUTransfer: Starting NPU Transfer Client, Transfer version 2.1.0 (b5861e7@2020-11-23T11:50:36)
D NPUTransfer: Transfer spec = local:transfer_proxy
D NPUTransfer: ERROR: socket read fd = 3, n = -1: Connection reset by peer
D NPUTransfer: Transfer client closed, fd = 3
E RKNNAPI: rknn_init, server connect fail! ret = -9(ERROR_PIPE)!
E init_runtime: Catch exception when init runtime!
E init_runtime: Traceback (most recent call last):
E init_runtime:   File "rknn/api/rknn_base.py", line 2011, in rknn.api.rknn_base.RKNNBase.init_runtime
E init_runtime:   File "rknn/api/rknn_runtime.py", line 361, in rknn.api.rknn_runtime.RKNNRuntime.build_graph
E init_runtime: Exception: RKNN init failed. error code: RKNN_ERR_DEVICE_UNAVAILABLE
```

原因可能是板端未开启 rknn_server 服务，需要根据以下说明运行 rknn_server 服务才可以正常连板：

https://github.com/rockchip-linux/rknpu2/blob/master/rknn_server_proxy.md

9.4 连板调试时，rknn_init 失败，返回-6 或模型非法的错误

错误信息如下：

```
E RKNNAPI: rknn_init,    msg_load_ack fail, ack = 1(ACK_FAIL), expect 0(ACK_SUCC)!
D NPUTransfer: Transfer client closed, fd = 4
E init_runtime: Catch exception when init runtime!
E init_runtime: Traceback (most recent call last):
E init_runtime:      File "rknn/api/rknn_base.py", line 2011, in
rknn.api.rknn_base.RKNNBase.init_runtime
E init_runtime:      File "rknn/api/rknn_runtime.py", line 361, in
rknn.api.rknn_runtime.RKNNRuntime.build_graph
E init_runtime: Exception: RKNN init failed. error code: RKNN_ERR_MODEL_INVALID
```

出现该错误一般有以下几种情况：

1）在生成 rknn 模型时，不同版本的 RKNN-Toolkit2 和驱动是有对应关系的，建议将 RKNN-Toolkit2 / RKNPU2 和板子的固件都升级到最新的版本。

2）没有正确设置 target_platform。例如不设置 config 接口中的 target_platform 时，生成的 RKNN 模型只能在 RK3566/RK3568 上运行，而不能在 RK3588/RV1103/RV1106/RK3562 上运行。如果要在 RK3588/RV1103/RV1106/RK3562 上运行，则需要在调用 config 接口时设置 target_platform。

3）如果是在 Docker 容器中推理时出现该问题，有可能是因为宿主机上的 npu_transfer_proxy 进程没有结束，导致通信异常。可以先退出 Docker 容器，将宿主机上的 npu_transfer_proxy 进程结束掉，然后再进入容器执行推理脚本。

4）也可能是 RKNN-Toolkit2 转出来的模型本身有问题。这时可以获取如下信息，反馈给瑞芯微 NPU 团队：如果是 PC 连开发板调试，或者在开发板上运行模型，可以串口连到开发板，然后设置环境变量 RKNN_LOG_LEVEL=5，之后执行 restart_rknn.sh，然后再重跑程序，将开发板上的详细日志记录下来。

9.5 连板调试时，rknn_init 失败，返回设备不可用的错误

错误信息如下：

```
E RKNNAPI: rknn_init,  msg_ioctl_ack fail, data_len = 104985, expect 103961!
D NPUTransfer: Transfer client closed, fd = 3
E init_runtime: Catch exception when init runtime!
E init_runtime: Traceback (most recent call last):
E init_runtime:   File "rknn/api/rknn_base.py", line 1916, in rknn.api.rknn_base.RKNNBase.init_runtime
E init_runtime:   File "rknn/api/rknn_runtime.py", line 360, in rknn.api.rknn_runtime.RKNNRuntime.build_graph
E init_runtime: Exception: RKNN init failed. error code: RKNN_ERR_DEVICE_UNAVAILABLE
```

该问题的原因比较复杂，请按以下方式排查：

确保 RKNN-Toolkit2 / RKNPU2 及板子的系统固件都已经升级到最新版本。

以 RK3566 为例，这些组件的版本查询方法如下：

```
# 在 RK3566 (Android)终端里执行：
dmesg | grep rknpu      # 查询 NPU 驱动版本
strings /vendor/bin/rknn_server |grep build      # 查询 rknn_server 版本
strings /vendor/lib64/librknnrt.so | grep version      # 查询 librknnrt 版本

# 在 RK3566 (Linux)终端里执行：
dmesg | grep rknpu      # 查询 NPU 驱动版本
strings /usr/bin/rknn_server |grep build      # 查询 rknn_server 版本
strings /usr/lib/librknnrt.so | grep version      # 查询 librknnrt 版本
```

也可以通过 rknn.get_sdk_version 接口查询版本信息。

另外，需要特别注意：

1） 确保 adb devices 或 rknn.list_devices()都能看到设备，并且 rknn.init_runtime()的 target 和 device_id 设置正确。

2） 如果使用的是 RV1103/RV1106，则不支持连板调试。

9.6 Runtime 出现“Invalid RKNN format”报错

Runtime 上出现以下报错：

```
rknn_init error ret=-1
root@firefly:/home/firefly/rknn_yolov5_demo/build/build_linux_aarch64# ./rknn_yolov5_demo yolov5s.rknn bus.jpg
post process config: box_conf_threshold = 0.50, nms_threshold = 0.60
Read bus.jpg ...
img width = 640, img height = 640
Loading mode...
E RKNN: [06:28:39.048] parseRKNN from buffer: Invalid RKNN format!
E RKNN: [06:28:39.049] rknn_init, load model failed!
rknn_init error ret=-1
root@firefly:/home/firefly/rknn_yolov5_demo/build/build_linux_aarch64#
```

原因：

1）可能是模型转换时的 target_platform 没有设置对，或没有设置（如没有设置默认是 rk3566）。

2）Runtime 版本与 RKNN-Toolkit2 不兼容。

建议：

1）设置正确的 target_platform。

2）RKNN-Toolkit2 与 Runtime 要一起更新到同一个版本。

9.7 rknn.inference 耗时与 rknn.eval_perf 理论速度不一致

因为 rknn.inference 使用 PC+adb 的方式进行连板推理，因此存在着一些固定的数据传输开销，因此与 rknn.eval_perf 理论速度不一致。

对于更真实的帧率，建议直接在开发板上使用 RKNPU2 C API 进行测试。

9.8 rknn.inference 对多 batch 的支持

RKNN-Toolkit2 1.4.0 及之后的版本，需要在构建 RKNN 模型时就指定输入图片的数量，详细用法参考《Rockchip_User_Guide_RKNN_Toolkit2_CN》中关于 build 接口的说明。

另外，当 rknn_batch_size 大于 1（如等于 4 时），python 里推理的调用要由：

```
outputs = rknn.inference(inputs=[img])
```

修改为：

```
img = np.expand_dims(img, 0)
img = np.concatenate((img, img, img, img), axis=0)
outputs = rknn.inference(inputs=[img])
```

完整示例请参考：

examples/functions/batch_size/

9.9 运行多个 RKNN 模型

运行两个或多个模型时，需要创建多个 RKNN 对象。一个 RKNN 对象对应一个模型，类似一个上下文。每个模型在各自的上下文里初始化模型，推理，获取推理结果，互不干涉。这些模型在 NPU 上推理时是串行进行的。

9.10 模型推理的耗时非常长，而且得到的结果错误

如果推理耗时超过 20s，且结果错误，这通常是 NPU 出现了 NPU Hang 的 BUG。如果遇到该

问题，可以尝试更新 RKNN-Toolkit2 / RKNPU2 到最新版本。

9.11 非量化模型推理慢

NPU 浮点算力比较弱，针对量化模型 NPU 有更好的优化，所以量化模型性能会比浮点模型好很多。在 NPU 上，建议用量化的模型。

9.12 模型输入为 3 维情况下，连板推理结果错误

模型的输入为 3 维情况下，如出现 Simulator 的仿真结果正确，但连板推理结果错误的情况。原因可能是当前 NPU 的输入 3 维支持还不完善，后面会完善 3 维的支持。

建议：

1）先将模型输入改为 4 维。

2）更新 RKNN-Toolkit2 / RKNPU2 至最新版本进行尝试。

9.13 连板端推理结果错误，并且每次都不一致

ONNX 模型转 RKNN 后，用 Simulator 的仿真结果正确，并且每次结果都一致，但在连板推理时结果错误，并且每次都不一致。这种问题可能是板端 npu 内核驱动 bug 导致，此时需要更新板端的 NPU 内核驱动，并且需要一并更新最新的 RKNN-Toolkit2 / RKNPU2。

9.14 模型存在较多的 Resize OP 时，出现精度下降问题

当 ONNX 模型里存在较多的 Resize OP 时，转换为 RKNN 后出现精度下降。可能的原因是：

1）精度下降是因为 NPU 目前还不支持硬件级别 Resize（后续会支持），转换工具会将 Reszie 转为 ConvTranspose，会导致一点点的精度丢失。

2）如模型有多个串联的 Resize，则可能会累积了太多误差导致精度下降比较多。

建议：

1）目前尽量避免 Resize 的使用（如将 Resize 改为 ConvTranspose 再进行训练）

2）可以在 rknn.config 里加入 optimization_level=2 的参数，此时 Resize Op 会走 cpu，精度不会掉，但会导致性能下降。

9.15 do_quantization 设为 False 以后推理结果都为 nan

build 接口中的 do_quantization 设为 True 时推理结果没有异常，但设为 False 以后推理结果就都变为 nan 了。原因可能是 do_quantization=False 时，RKNN 模型的运算类型是 fp16 的，但该模型的中间层（如卷积）输出的范围可能超出了 fp16（65536）的范围（如-51597~75642）

建议：

训练的时候需要保证中间层的输出不超过 fp16 的表达范围（一般通过添加 BN 层来做归一化）

9.16 QAT 模型与 RKNN 模型结果不一致

在 Pytroch 框架下使用 QAT 训练了一个分类模型并转为 RKNN 模型，对该模型使用 Pytorch 和 RKNN 分别进行推理，发现得到的结果不一样，原因可能是 Pytorch 的推理没有设置 engine='qnnpack'，因为 RKNN 的推理方式与 qnnpack 更为贴近。

9.17 怎么测试模型运行时候内存占用率

可以使用 rknn.eval_memory 接口，输出的日志里有个 total，就是总的占用大小。

9.18 性能评估时，开启 perf_debug 与关闭该参数，性能数据的差异

开启 perf_debug 时，为了收集每层的信息，会添加一些调试代码，并且可能禁用一些并行的机制，因此耗时比 perf_debug=False 时多一些。

开启 perf_debug 的主要作用是看模型中是否有耗时占比比较多的层，以此为依据来设计优化方案。

9.19 环境用的 docker，之前连板推理正常，重启 docker 后，推理时卡在初始化环境阶段？

因为 docker 重启时 npu_transfer_proxy 类似于异常退出的状态，导致开发板上的 rknn_server 无法检测到上端连接已经断开，这时需要重启下开发板，重置 rknn_server 的连接状态。

10 深度神经网络模型设计建议

10.1 如何设计卷积神经网络，使其能在 RKNN 上实现最佳性能

有以下建议：

- 卷积核设置

推荐在设计的时候尽量使用 3x3 的卷积核，这样可以实现最高的乘加运算单元（MAC）利用率，使得 NPU 的性能最佳。

另外，在 group 卷积情况下，group 的值不要设置得太大，并且 weight 的输入通道尽量 16 对齐。

- 网络稀疏化

当前的神经网络存在过度参数化现象，并且在其设计时会存在很多冗余。NPU 针对稀疏矩阵，有进行跳零计算和内存提取方面的优化。所以建议在设计网络的时候，可以针对性的进行网络稀疏化设计，以利用该技术进一步提高网络性能。

- 去除冗余的 OP

当存在无效的 OP 时，可以去掉该 OP，如输入输出 shape 一致的 Reshape 等。当存在并行的操作一致的 OP 时，可以去掉其中一个，如两个来自同一个输入的 Relu 等。

- OP 融合设计

当存在连续的可融合 OP 时，尽量提前融合在一起（特别是针对 QAT 或 PTQ 的已量化模型），如：

1） Mul+Gemm 时，Mul 可以融合进 Gemm。

2） Gemm+Add 时，Add 可以融合进 Gemm。

3） Conv+BN 时，BN 可以融合进 Conv。

4） ConvTranspose+BN 时，BN 可以融合进 ConvTranspose。

5） Conv+Add 时，Add 可以融合进 Conv。

6） Conv+Mul 时，Mul 可以融合进 Conv。

7） Mul+Conv 时，Mul 可以融合进 Conv。

8） Add+Add 时，两个 Add 可以进行融合。

9） Mul+Mul 时，两个 Mul 时可以进行融合。

10） Mul+Add 时，Mul 和 Add 可以转换成 BN。

11） MatMul+Add 时，MatMul 和 Add 可以转换成 Gemm。

- Resize 改进

如模型存在 Resize 时，因为 NPU 对 Resize 支持较弱，建议将 Resize 转为 ConvTranspose 再进行训练。

- Concat 和 Split 对齐要求

尽量保证 Concat 的各个输入的通道维或 Split 的各个输出的通道维，在 RK3566/RK3568 下是 8 对齐的，在 RK3588/RV1103/RV1106/RK3562 下是 16 对齐。

- 3D 卷积支持

目前 RKNN 不支持 3D 卷积，需要将 3D 卷积替换成等效的 2D 卷积。

- Keepdims 参数

尽量保证 ReduceMean、ReduceMax、ReduceMin、ReduceSum、ArgMin 和 ArgMax 的 keepdims 参数为 1。

- 非 4 维 OP 支持

目前 RKNN 只对 4 维的 OP 支持较好，其他的维度（如 2/3/5 维）支持的并不完善，因此需要尽量将这些其他维度的 OP 改为 4 维。

- OP 限制

大部分 OP 都会有一些限制条件，各个框架下 OP 的支持情况可参考：

https://github.com/rockchip-linux/rknn-toolkit2/blob/master/doc/RKNNToolKit2_OP_Support.md

对于更详细的 OP 限制，可以参考：

https://github.com/rockchip-linux/rknpu2/tree/master/doc/RKNN_Compiler_Support_Operator_List.pdf

11 附录

11.1 参考文档

OP 支持列表：《RKNNToolKit2_OP_Support.md》

快速上手指南：《Rockchip_Quick_Start_RKNN_Toolkit2_CN.pdf》

RKNN-Toolkit2 使用指南：《Rockchip_User_Guide_RKNN_Toolkit2_CN.pdf》

问题排查手册：《Rockchip_Trouble_Shooting_RKNN_Toolkit2_CN.pdf》

以上文档均放在 doc 目录中，也可以访问以下链接查阅：

https://github.com/rockchip-linux/rknn-toolkit2/tree/master/doc

11.2 问题反馈渠道

请通过瑞芯微 Redmine 将问题反馈给 Rockchip NPU 团队。

Rockchip Redmine: https://redmine.rock-chips.com/

注：Redmine 账号需要通过销售或业务人员开通。如果是第三方开发板，请先找原厂反馈问题。