Classification Level: Top secret ( )　Secret ( )　Internal ( )　Public (√)

RKNN-Toolkit Trouble Shooting

(Technology Department, Graphic Computing Platform Center)

Mark: [ ] Editing [√] Released	Version	1.6.1
	Author	HPC
	Completed Date	2021-05-21
	Auditor	Vincent
	Reviewed Date	2021-05-21

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd

Revision History

Version no.	Author	Revision Date	Revision description	Auditor
V0.9	HPC	2019-04-01	Initial version release.	Vincent
V1.0	HPC	2019-07-18	Add some questions.	Vincent
V1.1	HPC	2019-08-22	Add some convolution acceleration tips.	Vincent
V1.2	HPC	2019-10-11	Add some questions.	Vincent
V1.3	HPC	2019-12-25	Add some questions, rearrange directory structure.	Vincent
V1.3.2	HPC	2020-04-13	Add some questions.	Vincent
V1.4.0	HPC	2020-08-13	Add some questions.	Vincent
V1.6.0	HPC	2020-12-31	Add some questions.	Vincent
V1.6.1	HPC	2021-05-21	Add some questions.	Vincent

Contents

1. RKNN Toolkit usage related questions

1.1 RKNN Toolkit installation problem

1.1.1 “undefined symbol: PyFPE_jbuf” error occurs when install RKNN Toolkit

The reason of the error is Python environment is not clean, for example, numpy is installed in two different paths. You can re-build a clean Python environment and try again.

1.1.2 “Permission Denied” error occurs when install RKNN Toolkit on Toybrick

The reason is there is no root authority. Need to add ‘--user’ option for installation.

1.1.3 Is it possible to install the RKNN Toolkit in the virtual machine and use the simulator for inference?

Yes. The virtual machine operating system must be Ubuntu16.04 or Ubuntu18.04, and the CPU architecture is x86_64.

1.2 Model configuration problem

1.2.1 Why does channel_mean_value of rknn.config function have 4 values? If it is rgb image, does it still have 4 values?

channel-mean-value of rknn.config: used to set the preprocessing command line parameter. It includes four values (M0 M1 M2 S0). The first three values are mean value parameters and the last value is Scale parameter. If the input data have three channels (Cin0, Cin1, Cin2), the output data will be (Cout0,Cout1, Cout2) after preprocessing. The calculating process is as below:

Cout0 = (Cin0 - M0)/S0

Cout1 = (Cin1 - M1)/S0

Cout2 = (Cin2 - M2)/S0

For example, if need to formulate the input data into [-1, 1], you can set this parameter as (128 128 128 128);

If need to formulate the input data into [0, 1], you can set this parameter as (0 0 0 255).

Note: This parameter has been replaced with mean_values and std_values since version 1.4.0, and officially removed from version 1.6.0.

1.2.2 When the input image is gray picture with single channel, how to set rknn.config interface?

Before version 1.3.0, Please refer to the answer of 1.2.1, when the input image is single channel, only ”Cout0 = (Cin0 - M0)/S0” is used, so you can set as (M0, 0, 0, S0), while the values of M1 and M2 are not used.

Start from version 1.3.0, the channel_mean_value of single channel input should be set to (M0, S0).

Starting from 1.4.0, single channel images can be set to mean_values=[[M0]], std_values=[[S0]], or channel_mean_value=’M0, S0’. Since version 1.6.0, the channel_mean_value parameter has been removed, please use mean_values and std_values

1.2.3 How to set scale parameter of rknn.config function? That is to compress the input range into a certain scope, e.g. from (0-255) to (0-1).

Refer to the answer of 1.2.1

1.2.4 How to set “channel_mean_value” when input channel large than 3?

Before version 1.3.0, for example, when the input shape is 1x25x25x96(NHWC format), you don’t need to set channel_mean_value or reorder_channel. The default value of mean will set to 0, scale will set to 1.

Start from version 1.3.0, for example, when the input shape is 1x25x25x4(NHWC format), the channel_mean_value should be set to (M0, M1, M2, M3, S0). When the input shape is 1x25x25x96(NHWC format, channel is large than 4), you don’t need to set channel_mean_value or reorder_channel. The default value of mean will set to 0, scale will set to 1.

Starting from version 1.4.0, when the dimension is between 5 and 128, no setting is required, but the maximum number of channels cannot exceed 128, otherwise an error will be reported.

Starting from version 1.6.0, if not set, the mean value is set to 0, and the scale value is set to 1. Otherwise, you need to set it yourself, and channel_mean_value will be removed, please use mean_values/std_values instead.

1.2.5 What is the execution order of channel_mean_value and order_channel?

The internal processing of RKNN is to do channel conversion first, and then do mean and scale processing. Therefore, for example, for Caffe's model, reorder_channel is set to "2 1 0", mean needs to be set in the order of BGR.

1.3 Model conversion problem

1.3.1 Troubleshooting steps when loading model errors

First confirm whether the original deep learning framework can load the model and perform correct inference.

Secondly, please upgrade RKNN Toolkit to the latest version. If the model has a layer (or OP) that is not supported by RKNN Toolkit, by turning on the debug log switch, you can see which layer or OP is not supported by RKNN Toolkit in the log. This type of error log usually contains "Try match xxx failed" or "Not match xxx", etc. If there is a problem of incorrect shape processing when the model is converted, you can find hints in the log such as concat/add/matmul dimension mismatch, or calculation of output tensor error.

If the first step fails, please check the original model for problems. If it still cannot be converted after upgrading to the latest version of RKNN Toolkit, or if some OPs does not supported, please report the version of the tool used and the detailed log when the conversion problem occurs to the Rockchip NPU development team.

1.3.2 What deep learning framework does the RKNN Toolkit support? Whether to support all versions of these deep learning frameworks?

Deep learning frameworks supported by the RKNN Toolkit include TensorFlow, TensorFlow Lite, Caffe, ONNX and Darknet.

It corresponds to the version of each deep learning framework as follows:

RKNN Toolkit	TensorFlow	TF Lite	Caffe	ONNX	Darknet	Pytorch	MXNet	Keras
1.0.0	>=1.10.0, <=1.13.2	Schema version = 3	1.0	Release version 1.3.0	Last commit: 810d7f7	Not Support	Not Support	Not Support
1.1.0	>=1.10.0, <=1.13.2	Schema version = 3	1.0	Release version 1.3.0	Last commit: 810d7f7	Not Support	Not Support	Not Support
1.2.0	>=1.10.0, <=1.13.2	Schema version = 3	1.0	Release version 1.4.1	Last commit: 810d7f7	Not Support	Not Support	Not Support
1.2.1	>=1.10.0, <=1.13.2	Schema version = 3	1.0	Release version 1.4.1	Last commit: 810d7f7	Not Support	Not Support	Not Support
1.3.0	>=1.10.0, <=1.13.2	Schema version = 3	1.0	Release version 1.4.1	Last commit: 810d7f7	>=1.0.0, <=1.2.0	>=1.4.0, <=1.5.1	Not Support
1.3.2	>=1.10.0, <=1.13.2	Schema version = 3	1.0	Release version 1.4.1	Last commit: 810d7f7	>=1.0.0, <=1.2.0	>=1.4.0, <=1.5.1	Not Support
1.4.0	>=1.10.0, <=1.13.2	Schema version = 3	1.0	Release version 1.4.1	Last commit: 810d7f7	>=1.0.0, <=1.2.0	>=1.4.0, <=1.5.1	Not Support

1.6.0	>=1.10.0, <=2.0.0	Schema version = 3	1.0	Release version 1.6.0	官方最新提交：810d7f7	>=1.0.0, <=1.6.0	>=1.4.0, <=1.5.1	>=2.1.6-tf

Note:

1. In compliance with semver, SavedModels written with one version of TensorFlow can be loaded and evaluated with a later version of TensorFlow with the same major release. So in theory, the pb file generated by TensorFlow before version 1.14.0, RKNN Toolkit 1.0.0 and later are supported. For more information on TensorFlow version compatibility, please refer to the official link:
   https://www.tensorflow.org/guide/versions
2. RKNN Toolkit uses the TF Lite schema commits in link:
   https://github.com/tensorflow/tensorflow/commits/master/tensorflow/lite/schema/schema.fbs
   Commit hash: 0c4f5dfea4ceb3d7c0b46fc04828420a344f7598.
   Because TF Lite schema may not compatible with each other, TF Lite models with older or newer schema may not be loaded successfully.
3. There are two caffe protocols RKNN Toolkit uses, one based on the officially modified protocol of berkeley, and one based on the protocol containing the LSTM layer. The protocol based on the official revision of berkeley comes from this link: https://github.com/BVLC/caffe/tree/master/src/caffe/proto, commit hash is 21d0608. On this basis RKNN Toolkit have added some OPs. The protocol containing the LSTM layer refers to: https://github.com/xmfbit/warpctc-caffe/tree/master/src/caffe/proto, commit hash is bd6181b. These two protocols are specified by the proto parameter in the load_caffe interface.
4. The relationship between ONNX release version and opset version, IR version refers to the official website description:
   https://github.com/microsoft/onnxruntime/blob/master/docs/Versioning.md

ONNX release version	ONNX opset version	Supported ONNX IR version

1.3.0	8	3
1.4.1	9	3

5. Darknet official Github link: https://github.com/pjreddie/darknet. Our current conversion rules are based on the latest commit of the master branch (commit value: 810d7f7).
6. RKNN Toolkit currently mainly supports the Keras version with TensorFlow as the backend. The tested Keras version is the Keras built-in TensorFlow.

1.3.3 Which framework of RKNN Toolkit supports better? Which OPs are supported?

RKNN Toolkit supports different frameworks to some extent. The OP currently supported by each framework can refer to the following link and choose according to the OP used in the model:

https://github.com/rockchip-linux/rknn-toolkit/blob/master/doc/RKNN_OP_Support_V1.6.0.md

1.3.4 Does RKNN Toolkit support model conversion with multiple inputs?

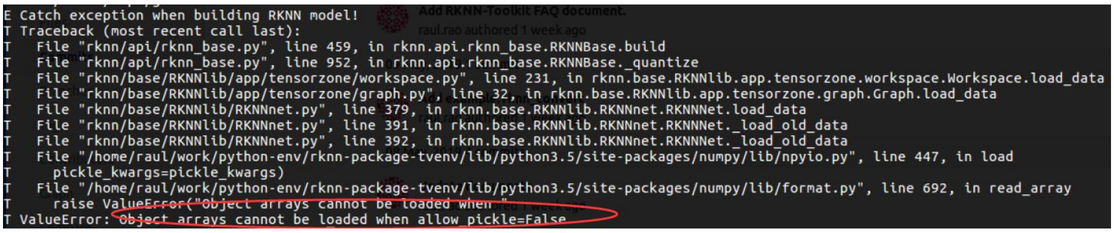
The RKNN Toolkit needs to be upgraded to version 1.2.0 or later.

1.3.5 When will it support to convert pytorch and mxnet model directly to rknn?

Please update RKNN Toolkit to version 1.3.0.

1.3.6 When I load model, the numpy module raises error: Object arrays cannot be loaded when allow pickle=False.

The error message is as follows：



```
E Catch exception when building RKNN model!
T Traceback (most recent call last):
T   File "rknn/api/rknn_base.py", line 459, in rknn.api.rknn_base.RKNNBase.build
T   File "rknn/api/rknn_base.py", line 952, in rknn.api.rknn_base.RKNNBase._quantize
T   File "rknn/base/RKNNlib/app/tensorzone/workspace.py", line 231, in rknn.base.RKNNlib.app.tensorzone.workspace.Workspace.load_data
T   File "rknn/base/RKNNlib/app/tensorzone/graph.py", line 32, in rknn.base.RKNNlib.app.tensorzone.graph.Graph.load_data
T   File "rknn/base/RKNNlib/RKNNnet.py", line 379, in rknn.base.RKNNlib.RKNNnet.RKNNNet.load_data
T   File "rknn/base/RKNNlib/RKNNnet.py", line 391, in rknn.base.RKNNlib.RKNNnet.RKNNNet._load_old_data
T   File "rknn/base/RKNNlib/RKNNnet.py", line 392, in rknn.base.RKNNlib.RKNNnet.RKNNNet._load_old_data
T   File "/home/raul/work/python-env/rknn-package-tvenv/lib/python3.5/site-packages/numpy/lib/npyio.py", line 447, in load
T     pickle_kwargs=pickle_kwargs)
T   File "/home/raul/work/python-env/rknn-package-tvenv/lib/python3.5/site-packages/numpy/lib/format.py", line 692, in read_array
T     raise ValueError("Object arrays cannot be loaded when "
T ValueError: Object arrays cannot be loaded when allow_pickle=False
```

This error is caused by the change in the default value of the allow_pickle parameter of the load file interface after numpy is upgraded to 1.16.3. There are two solutions: one is to reduce the numpy version to version 1.16.2 or lower; the other is to update RKNN Toolkit to version 1.0.0 or later.

1.3.7 How to convert only partial models when converting models?

As of RKNN Toolkit 1.6.0, only Tensorflow supports converting local models. For the TensorFlow model, you can intercept the local model and convert it by specifying the input/output node when loading.

1.3.8 Does the converted RKNN model support encryption?

RKNN Toolkit introduces the model encryption function from version 1.6.0, which is realized by calling interface: export_encrypted_rknn_model.

1.3.9 What is the maximum support of the current RKNN model?

Main limitations: the memory limit of the model conversion platform, the number of nodes of the exported model, and the memory limit of the target device.

1.3.10 If the weight of the convolutional layer is dynamically updated, does RKNN Toolkit support it?

It needs to be updated to version 1.6.0 and later, and the driver must also be updated to version 1.6.0 and later.

1.3.11 Does RKNN Toolkit support the direct construction of RKNN model through op code, like Pytorch and other frameworks?

Not support. Only models generated by other frameworks can be converted into RKNN models.

1.4 Model quantification problem

1.4.1 RKNN Toolkit supported quantization method

RKNN supports two kinds of quantization mechanisms:

- **Quantization-aware training**

Refer to Tensorflow quantization-aware training (https://github.com/tensorflow/tensorflow/tree/master/tensorflow/contrib/quantize), which requires user should have some re-training experience of fine tune. Use rknn.build (do_quantization=False) after the quantized model is loaded through RKNN Toolkit, and now RKNN Toolkit will use the own quantization parameter of the model, so there is no loss on the quantization accuracy. Currently only imported quantified models of TensorFlow and TensorFlow Lite are supported. And TensorFlow's quantized model is limited by TensorFlow functions on Windows and cannot be loaded.

- **Post training quantization**

When use this method, user loads the well-trained float point model, and RKNN Toolkit will do the quantization according to the dataset provided by user. Dataset should try to cover as many input type of model as possible. To make example simple, generally put only one picture. Suggest to put more.

Currently RKNN Toolkit supports three kinds of quantization methods:

✓ asymmetric_quantized-u8（default）

This is the quantization method supported by tensorflow, which is also recommended by Google. According to the description in the article of Quantizing deep convolutional networks for efficient inference: A whitepaper, the accuracy loss of this quantization method is the smallest for most networks.

Its calculation formula is as follows：

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

Where ‘quant’ represents the quantized number; ‘float_num’ represents float; data type of ‘scalse’ if float32; data type of ‘zero-points’ is int32, it represents the corresponding quantized value when the real number is 0. Finally saturate ‘quant’ to [range_min, range_max].

$$range\_\max = 255$$
$$range\_\min = 0$$

Currently only supports the inverse quantization of u8, the calculation formula is as follows:

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

- ✓ dynamic_fixed_point-8

For some models, the quantization accuracy of dynamic_fixed_point-8 is higher than asymmetric_quantized-u8.

Its calculation formula is as follows：

$$quant = round(float\_num * 2^{fl})$$
$$quant = cast\_to\_bw$$

Where ‘quant’ represents the quantized number; ‘float_num’ represents float; ‘fl’ is the number of digits shifted to the left. Finally saturate ‘quant’ to [range_min, range_max].

$$range\_\max = 2^{bw-1} - 1$$
$$range\_\min = -(2^{bw-1} - 1)$$

If ‘bw’ equals 8, the range is [-127, 127].

- ✓ dynamic_fixed_point-16

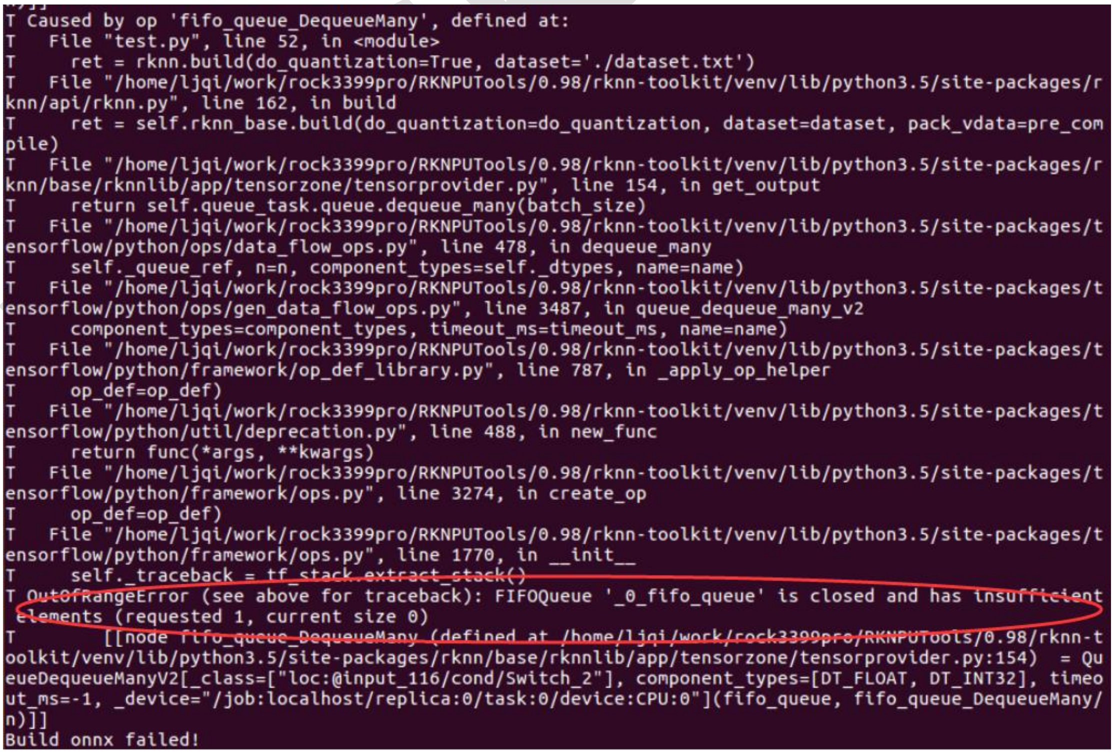
The quantization formula of dynamic_fixed_point-16 is the same as dynamic_fixed_point-8, except bw=16. For RK3399pro/RK1808, there is 300Gops int16 computing unit inside NPU, for some quantized to 8 bit network with relatively high accuracy loss, you can consider to use this quantization method.

1.4.2 If do_quantization is False during model conversion, will it do quantization? What is the quantization accuracy? (because the model is nearly half the size after conversion)

There are two scenarios. When the loaded model is the quantized model, do_quantization=False will use the quantization parameter of the model, for more details please refer to the answer of 1.4.1. When the loaded model is the non-quantized model, do_quantization=False will not do quantization, but will convert the weight from float32 to float16, which will not cause accuracy loss.

1.4.3 When structure RKNN model(invoking build interface), set do_quantization=False can build successfully, but set True will fail to build

The error log is as below:



```
T Caused by op 'fifo_queue_DequeueMany', defined at:
T   File "test.py", line 52, in <module>
T     ret = rknn.build(do_quantization=True, dataset='./dataset.txt')
T   File "/home/ljqi/work/rock3399pro/RKNPUTools/0.98/rknn-toolkit/venv/lib/python3.5/site-packages/r
knn/api/rknn.py", line 162, in build
T     ret = self.rknn_base.build(do_quantization=do_quantization, dataset=dataset, pack_vdata=pre_com
pile)
T   File "/home/ljqi/work/rock3399pro/RKNPUTools/0.98/rknn-toolkit/venv/lib/python3.5/site-packages/r
knn/base/rknnlib/app/tensorzone/tensorprovider.py", line 154, in get_output
T     return self.queue_task.queue.dequeue_many(batch_size)
T   File "/home/ljqi/work/rock3399pro/RKNPUTools/0.98/rknn-toolkit/venv/lib/python3.5/site-packages/t
ensorflow/python/ops/data_flow_ops.py", line 478, in dequeue_many
T     self._queue_ref, n=n, component_types=self._dtypes, name=name)
T   File "/home/ljqi/work/rock3399pro/RKNPUTools/0.98/rknn-toolkit/venv/lib/python3.5/site-packages/t
ensorflow/python/ops/gen_data_flow_ops.py", line 3487, in queue_dequeue_many_v2
T     component_types=component_types, timeout_ms=timeout_ms, name=name)
T   File "/home/ljqi/work/rock3399pro/RKNPUTools/0.98/rknn-toolkit/venv/lib/python3.5/site-packages/t
ensorflow/python/framework/op_def_library.py", line 787, in _apply_op_helper
T     op_def=op_def)
T   File "/home/ljqi/work/rock3399pro/RKNPUTools/0.98/rknn-toolkit/venv/lib/python3.5/site-packages/t
ensorflow/python/util/deprecation.py", line 488, in new_func
T     return func(*args, **kwargs)
T   File "/home/ljqi/work/rock3399pro/RKNPUTools/0.98/rknn-toolkit/venv/lib/python3.5/site-packages/t
ensorflow/python/framework/ops.py", line 3274, in create_op
T     op_def=op_def)
T   File "/home/ljqi/work/rock3399pro/RKNPUTools/0.98/rknn-toolkit/venv/lib/python3.5/site-packages/t
ensorflow/python/framework/ops.py", line 1770, in __init__
T     self._traceback = tf_stack.extract_stack()
T OutOfRangeError (see above for traceback): FIFOQueue '_0_fifo_queue' is closed and has insufficient
 elements (requested 1, current size 0)
T         [[node fifo_queue_DequeueMany (defined at /home/ljqi/work/rock3399pro/RKNPUTools/0.98/rknn-t
oolkit/venv/lib/python3.5/site-packages/rknn/base/rknnlib/app/tensorzone/tensorprovider.py:154)  = Qu
eueDequeueManyV2[_class=["loc:@input_116/cond/Switch_2"], component_types=[DT_FLOAT, DT_INT32], timeo
ut_ms=-1, _device="/job:localhost/replica:0/task:0/device:CPU:0"](fifo_queue, fifo_queue_DequeueMany/
n)]]
Build onnx failed!
```

It is because there is no data in dataset.txt, or the data format is not supported. Recommend to use jpg or npy.

1.4.4 What is the role of dataset during RKNN quantization? Why does quantization need to relate to dataset?

During RKNN quantization, need to find appropriate quantization parameters, such as scale or zero point. These quantization parameters should be selected according to the inference of the actual input.

1.4.5 Upgraded to RKNN Toolkit 1.2.0, there are 200 pictures in dataset.txt, but quantitative correction is quickly completed. The accuracy of the rknn model is very low. Are these pictures used for quantitative correction?

RKNN Toolkit 1.2.0 adjusts the default value of batch_size in config interface. In this version, if you want to use multiple pictures for quantization correction, the value of this parameter should be set to the corresponding number of pictures. If this value is set too large, it may cause program exceptions due to exhaustion of system memory. In this situation, you need to upgrade to version 1.2.1 or later. In version 1.2.1, the default value of batch_size is restored to 100, and multiple quantization correction can be achieved with epochs parameter. The number of images used for quantization correction is the product of batch_size and epochs. For example, if there are 200 pictures in the dataset file, then batch_size is set to 100, epochs is set to 2, or batch_size is set to 200, and epochs is set to 1, all of which can achieve the quantization correction of 200 pictures. But the memory usage peak of theformer is lower than that of the latter. If you only want to use 100 of them, you can set batch_size to 100 and epochs to 1.

1.4.6 The shape of numpy array in dataset is (4, 640, 480), but when building quantized rknn model, the log prompts shape (640, 480, 480), then build failure.

When using numpy array for quantization correction, the data should be arranged in the order of “NHWC”.

1.4.7 Is the size of the image used for quantization correction the same as the size of the model input?

Not required. RKNN Toolkit automatically scales images. Howerver, because zooming can change the image information, it may have some impact on the accuracy, so it is better to use pictures of similar size.

1.4.8 When calling the build interface to build the RKNN model, if do_quantization=True is set, the program will be killed after running for a period of time. Why? How to resolve this problem?

This problem is usually caused by insufficient system memory and too much data is used in quantitative correction.

When you encounter this problem, you can choose a PC with a larger memory to continue the conversion; or set the batch_size in the config interface to a smaller value, such as 8 or 16. The default value of this parameter is 100. If the input is relatively large, it is easy to appear memory Not enough phenomenon.

1.4.9 Does batch_size have a big impact on the results?Does batch_size have a big impact on the results?

Some models will have an impact. Generally, the larger the batch_size setting, the higher the accuracy. But it is still related to the specific model and needs to be experimentally verified.

1.5 Initialize the runtime problem

1.5.1 When using adb to query connected devices, it prompts no permission.

The error message is as follows:

```
test@test:~/ $ adb devices
List of devices attached
* daemon not running; starting now at tcp:5037
* daemon started successfully
1109no permissions (user hpcci is not in the plugdev group); see
[http://developer.android.com/tools/device.html]
1126no permissions (user hpcci is not in the plugdev group); see
[http://developer.android.com/tools/device.html]
```

Solution:

Find script: <sdk>/platform-tools/update_rk_usb_rule/linux/update_rk1808_usb_rule.sh. Execute it. Then try ‘adb devices’ again. If there is no permission, please replug the device or restart the machine. When the permissions are normal, the output is as follows:

```
test@test:~/ $ adb devices
List of devices attached
1109device
1126device
```

1.5.2 When I call rknn_init(), it raises error: RKNN_ERR_MODEL_INVALID.

The error message is as follows：

```
    E RKNNAPI: rknn_init,    msg_load_ack fail, ack = 1, expect 0!
    E Catch exception when init runtime!
    T Traceback (most recent call last):
    T     File "rknn/api/rknn_base.py", line 646, in rknn.api.rknn_base.RKNNBase.init_runtime
    T     File "rknn/api/rknn_runtime.py", line 378, in
rknn.api.rknn_runtime.RKNNRuntime.build_graph
    T Exception: RKNN init failed. error code: RKNN_ERR_MODEL_INVALID
```

There are some situations in which this error occurs:

1. The option to pre_compile=True was used when generating the rknn model. Different versions of the RKNN Toolkit and drivers have a corresponding relationship. It is recommended to upgrade the firmware of the RKNN Toolkit and the board to the latest version.
2. There is no option to use pre_compile=True when generating the rknn model. At this

time, the system firmware is too old. It is recommended to upgrade the firmware of the board to the latest version.

3. The parameter target_platform is not set correctly. For example, when the target_platform in the config interface is not set, the generated RKNN model can only be run on RK1806/RK1808/RK3399Pro, but not on RV1109/RV1126. If you want to run on RV1109/RV1126, you need to set target_platform=['rv1109', 'rv1126'] when calling the config interface.
4. If the problem occurs when inferring in a Docker container, it may be because the npu_transfer_proxy process on the host machine is not terminated, resulting in abnormal communication. please exit the Docker container first, terminate the npu_transfer_proxy process on the host, and then enter the container to execute the inference script.
5. It may also be that the model transferred from RKNN Toolkit has some problems. At this time, you can obtain the following information and feed it back to the Rockchip NPU team: If you are using a simulator, set verbose=True when initializing the RKNN object, print a detailed log, and record it. There will be a more detailed log description in the model initialization place The reason for the failure of the model check; if the PC is connected to the development board for debugging, or the model is running on the development board, you can connect the serial port to the development board, and then set the environment variable RKNN_LOG_LEVEL=5, then execute restart_rknn.sh, and then rerun the program. Record the detailed log on the development board.

1.5.3 When I call rknn_init(), error: RKNN_ERR_DEVICE_UNAVAILABLE raised.

The error message is as follows：

```
E RKNNAPI: rknn_init, driver open fail! ret = -9!
E Catch exception when init runtime!
T Traceback (most recent call last):
T File "rknn/api/rknn_base.py", line 617, in rknn.api.rknn_base.RKNNBase.init_runtime
T File "rknn/api/rknn_runtime.py", line 378, in
```

```
rknn.api.rknn_runtime.RKNNRuntime.build_graph
    T Exception: RKNN init failed. error code: RKNN_ERR_DEVICE_UNAVAILABLE
```

Please check it out as follows:

1) Make sure that the RKNN Toolkit and the firmware of devices have been upgraded to the latest version. The correspondence between each version of the RKNN Toolkit and the components of the system firmware is as follows：

RKNN Toolkit	rknn_server	NPU Driver	librknn_runtime
1.0.0	0.9.6/0.9.7	6.3.3.3718	0.9.8/0.9.9
1.1.0	0.9.8	6.3.3.03718	1.0.0
1.2.0	0.9.9	6.4.0.213404	1.1.0
1.2.1	1.2.0	6.4.0.213404	1.2.0
1.3.0	1.3.0	6.4.0.227915	1.3.0
1.3.2	1.3.2	6.4.0.7915	1.3.2
1.4.0	1.4.0	6.4.0.27915	1.4.0
1.6.0	1.6.0	6.4.3.5.293908	1.6.0

The version of these components is queried on RK1808 as follows：

```
    # execute these commands on RK1808
    dmesg | grep -i galcore        # Query the NPU driver version
    strings /usr/bin/rknn_server | grep build        # Query the rknn_server version
    strings /usr/lib/librknn_runtime.so | grep version        # Query the librknn_runtime
version
```

The version information can also be queried through the get_sdk_version interface, where the DRV version corresponds to the version of rknn_server.

2) Make sure the “adb devices” command can get the device, and the target and device_id settings of rknn.init_runtime() are correct.

3) If you use RKNN Toolkit 1.1.0 and above, make sure rknn.list_devices() can get the devices list.

4) If you are using a compute stick or NTB mode for the RK1808 EVB version, make sure you

have called update_rk1808_usb_rule.sh (contained in the RKNN Toolkit distribution) to get read and write access to the USB device.

5) If both ADB equipment and NTB equipment are connected, please make sure that only one type of equipment is in use at the same time. When using the ADB device, the npu_transfer_proxy process needs to be terminated; the ADB device can only be used on the Linux x86_64 platform.

6) If it is a Mac OS platform, please make sure that only one device is in use. If you want to use another device, please manually finish the npu_transfer_proxy process first.

7) If it is a Windows platform, please turn off programs such as 360 Security Guard/Tencent Computer Manager before use, otherwise this error may also occur.

8) If you are running the AARCH64 version of the RKNN Toolkit directly on the RK3399/RK3399Pro, make sure the system firmware has been upgraded to the latest version.

9) If you are using RV1109/RV1126, please check whether Mini Driver is used, Mini Driver does not support online debugging.

10) If it is RK3399Pro, only the development board with Android system firmware can be debugged online on the Linux_X86 PC, and other firmware can be directly logged in to the corresponding system to install the AARCH64 version of RKNN Toolkit for development and testing.

1.5.4 Prompt that dlopen failed during the initial operation of the Windows platform

The error log when the error occurs is as follows:

```
--> Init runtime environment
E Catch exception when init runtime!
E Traceback (most recent call last):
……
E     File “rknn\api\rknn_runtime.py”, line xxx, in
rknn.api.rknn_runtime.RKNNRuntime.__init__,
E     File “C:\Users\xxx\AppData\Local\Programs\Python\Python36\lib\ctypes\__init__.py”,
E              self._handle = _dlopen(self._name, mode)
E OSError: [WinError 126] Cannot find the specified module.
E Current device id is: None
E Devices connected:
E [‘1808s1’]
Init runtime environment failed
```

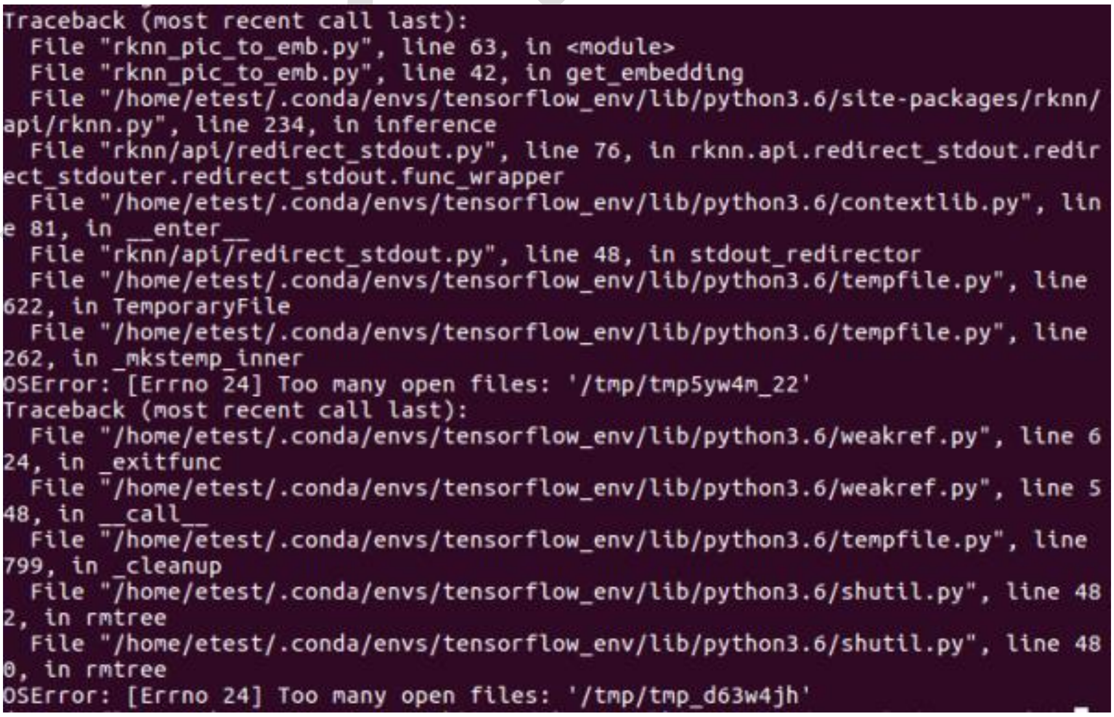
If this error occurs, you need to add the path of librknn_api.so to the system path PATH.

First, find the installation path <RKNN_INSTALL_PATH> of RKNN Toolkit through "pip show rknn-toolkit", then add the following two paths to the system PATH: <RKNN_INSTALL_PATH>/rknn/api/lib/hardware/LION/Windows_x64 and <RKNN_INSTALL_PATH>/ rknn/api/lib/hardware/PUMA/Windows_x64

1.6 Model inference problem

1.6.1 rknn.Inference() interface error or stuck happened after multiple invoke

If the error log is similar as below:



```
Traceback (most recent call last):
  File "rknn_pic_to_emb.py", line 63, in <module>
  File "rknn_pic_to_emb.py", line 42, in get_embedding
  File "/home/etest/.conda/envs/tensorflow_env/lib/python3.6/site-packages/rknn/
api/rknn.py", line 234, in inference
  File "rknn/api/redirect_stdout.py", line 76, in rknn.api.redirect_stdout.redir
ect_stdouter.redirect_stdout.func_wrapper
  File "/home/etest/.conda/envs/tensorflow_env/lib/python3.6/contextlib.py", lin
e 81, in __enter__
  File "rknn/api/redirect_stdout.py", line 48, in stdout_redirector
  File "/home/etest/.conda/envs/tensorflow_env/lib/python3.6/tempfile.py", line
622, in TemporaryFile
  File "/home/etest/.conda/envs/tensorflow_env/lib/python3.6/tempfile.py", line
262, in _mkstemp_inner
OSError: [Errno 24] Too many open files: '/tmp/tmp5yw4m_22'
Traceback (most recent call last):
  File "/home/etest/.conda/envs/tensorflow_env/lib/python3.6/weakref.py", line 6
24, in _exitfunc
  File "/home/etest/.conda/envs/tensorflow_env/lib/python3.6/weakref.py", line 5
48, in __call__
  File "/home/etest/.conda/envs/tensorflow_env/lib/python3.6/tempfile.py", line
799, in _cleanup
  File "/home/etest/.conda/envs/tensorflow_env/lib/python3.6/shutil.py", line 48
2, in rmtree
  File "/home/etest/.conda/envs/tensorflow_env/lib/python3.6/shutil.py", line 48
0, in rmtree
OSError: [Errno 24] Too many open files: '/tmp/tmp_d63w4jh'
```

Please update RKNN Toolkit to 0.9.9 or higher version.

1.6.2 rknn.inference() inferring speed slow issue

This issue has two kinds of phenomenon:

1) The speed of forward inferring test is slow, and some picture may take over 0.5s while testing mobilenet-ssd.

2) The time difference between model rknn.inference and rknn.eval_perf() is relatively big, such as:

Theoretical computing time(single picture)	1.79ms	8.23ms	7.485ms	30.55ms
Actual computing time(single picture)	21.37ms	39.82ms	33.12ms	76.13ms

There are two reasons for the issue of slow measured frame rate:

1. Using the method of pc + adb to upload picture is quite slow, as it has high frame rate requirement for network such as 1.79ms theoretically.
2. In the implementation of 0.9.8 and earlier, the inference included some extra time, and the 0.9.9 and later versions have been optimized.

For more real measured frame rate, you can directly use c/c++ api to test on the board.

1.6.3 The first inference of RKNN Toolkit 0.9.9 version is very slow

RKNN Toolkit 0.9.9 version postpones the model loading to the first inference, so the first inference is relatively slow. This issue has been resolved in versions 1.0.0 and later.

1.6.4 Returned outputs of YOLO forward test is [array1 , array2], the length is [10140 , 40560], what is the meaning of the returned value?

The outputs returned by rknn.inference is a list of numpy ndarray, the size and quantity of each model output data are different, users need to look up the corresponding output and analytic rule of models by themselves.

1.6.5 Does rknn.inference() support multiple pictures input at the same time? Or support batch input?

The RKNN Toolkit needs to be upgraded to version 1.2.0 or later. And you need to specify the number of input images when building the RKNN model. For detailed usage, refer to the description of the build interface in <Rockchip_User_Guide_RKNN_Toolkit_V1.2.1_CN.pdf>.

In addition, when rknn_batch_size is greater than 1 (e.g. equal 4), the inference code in python:

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

need modify to:

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

1.6.6 The same picture, why the inference result of rknn_batch_size=1 is different from the result of rknn_batch_size=4?

Because some OPs are optimized when the batch size is 4, the accuracy of the operation is higher, so some model results look a little different.

1.6.7 How to correspond the inference result to the each input when the batch_size is large than 1?

If the inference result corresponding to the input image(shape is [224, 224, 3]) is output(shape is [1, 1001]). When the batch_size is 2, the input is the concat result of two images(shape is [448, 224, 3]), the shape of corresponding inference result is [2, 1001].

1.6.8 The RKNN Toolkit python interface inference results are slightly different from the RKNN API C interface inference. Why?

It is possible that cv2 decoding and jpeg decoding are inconsistent. The python side can use numpy's tofile() method to export the image data to binary, and the C API directly reads the binary image data as input, thus ensuring the comparison under the same input.

1.6.9 The input of the model is single-channel. After reading the graph with opencv, it is used as input to inference, and the result is wrong.

When use the default imread interface of opencv to read the image, it will return a 3-channel graph, so the result is incorrect. Use the following method to read the image and the result will be correct：

```
img = cv2.imread('32x32_gray.jpg', 0)
```

1.6.10 The RKNN Toolkit version is 1.1.0. When inferring, the sum of some models` sotfmax is not 1.

Please update to version 1.2.1 or later.

1.6.11 If you want to run two or more models, how to distinguish which output is which model？

When running two or more models, you need to create multiple RKNN objects. One RKNN object corresponds to a model, which is similar to a context. Each model initializes the model in its own context. Each model initializes the model in its own context, makes inferences, and obtains inference results without interfering with each other. These models are inferred serially on the NPU.

1.6.12 Can four-dimensional data be passed in during inference?

Yes.

1.6.13 Model inference takes a very long time, and the results obtained are all 0. This is why?

If the inference takes more than 20s and the result is all 0, this is usually a GPU hang bug in the NPU. If you encounter this problem, you can try to update the NPU driver to version 1.5.0 or later.

1.6.14 The input data used by the original model is arranged in NCHW format. When using RKNN Toolkit for inference, is it necessary to use NCHW format?

When using the inference interface of RKNN Toolkit for inference, the default data arrangement format is NHWC format, which is the same as the data format read by OpenCV. If you are not using OpenCV to read the image, you must set the data_format parameter of the inference interface according to the format of the actual data. Because the underlying driver of RKNN Toolkit will convert according to the data arrangement of the original model, when calling the inference interface for inference, just fill in the data_format parameter according to the arrangement format of the actual data.

1.6.15 The inference time of the model without quantization is much longer than the model with quantization. Is this normal?

It is normal. NPU floating-point computing power is relatively weak, and for the quantized model, NPU has been optimized. So the performance of the quantized model will be much better than that of the floating-point model. On NPU, it is recommended to use a quantized model.

1.7 Pre-compilation issues

1.7.1 Fail to enable pre_compile=true when using RKNN Toolkit to convert model on the development board

Arm64 version RKNN Toolkit doesn’t support pre_compile so far, if need to open pre_compile, suggest to use x86 version RKNN Toolkit to do the conversion.

Or use the export_rknn_precompile_model interface to export the pre-compiled model from the EVB board.

1.7.2 Pre-compile model generated by RKNN Toolkit 0.9.9 can not run on RK3399Pro which NPU driver version is 0.9.6.

Pre-compiled model generated by RKNN Toolkit 1.0.0 can not run on device installed old driver (NPU driver version < 0.9.6), and pre-compiled model generated by old RKNN Toolkit (version < 1.0.0) can not run on device installed new NPU driver (NPU drvier version == 0.9.6). The driver version number can be queried through the get_sdk_version interface.

1.7.3 When calling rknn.build() with pre_compile=True, it raises an error, it can be successful if it is not set.

The error message is as follows：

```
E Catch exception when building RKNN model!
T Traceback (most recent call last):
T    File "rknn/api/rknn_base.py", line 515, in rknn.api.rknn_base.RKNNBase.build
T    File "rknn/api/rknn_base.py", line 439, in rknn.api.rknn_base.RKNNBase._build
T    File "rknn/base/ovxconfiggenerator.py", line 187, in
rknn.base.ovxconfiggenerator.generate_vx_config_from_files
T    File "rknn/base/RKNNlib/app/exporter/ovxlib_case/casegenerator.py", line 380, in
rknn.base.RKNNlib.app.exporter.ovxlib_case.casegenerator.CaseGenerator.generate
T    File "rknn/base/RKNNlib/app/exporter/ovxlib_case/casegenerator.py", line 352, in
rknn.base.RKNNlib.app.exporter.ovxlib_case.casegenerator.CaseGenerator._gen_special_case
T    File "rknn/base/RKNNlib/app/exporter/ovxlib_case/casegenerator.py", line 330, in
rknn.base.RKNNlib.app.exporter.ovxlib_case.casegenerator.CaseGenerator._gen_nb_file
T AttributeError: 'CaseGenerator' object has no attribute 'nbg_graph_file_path'
```

Please confirm:

1) The system is equipped with the gcc compiler toolchain

2) The name of the model only contains “letters”, “numbers”, “_”. Or you can update RKNN Toolkit to version 1.3.0 or later, the RKNN Toolkit will automatically handle these special characters.

If this is not the case, you can try to export the pre-compiled model from the board using the export_rknn_precompile_model interface.

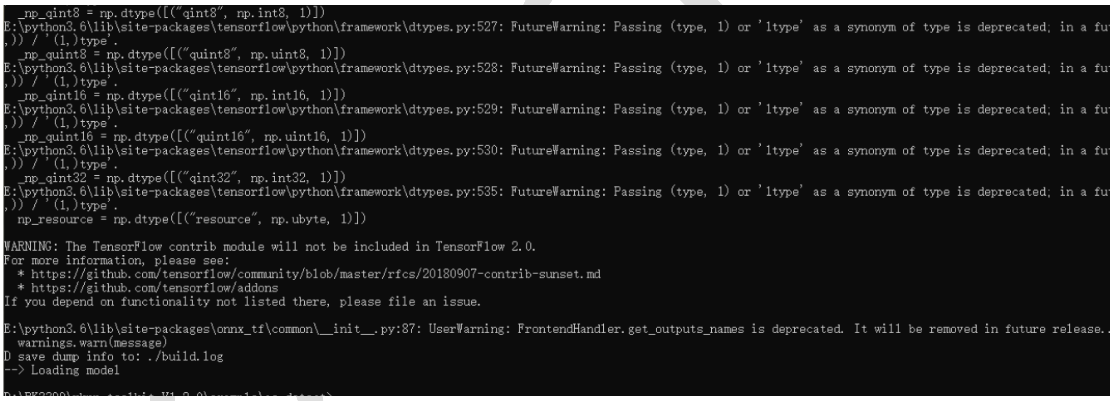
1.8 Log problems

1.8.1 When using the RKNN Toolkit, if the logging module is used in the program to output the log, it will report an error and exit.

Please upgrade to 1.2.1 or later.

1.8.2 Upgraded to RKNN Toolkit 1.2.0，after calling load_xxx interfaces, the program exits directly without any log.

The message is as follows：



```
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
E:\python3.6\lib\site-packages\tensorflow\python\framework\dtypes.py:527: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a fu
,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
E:\python3.6\lib\site-packages\tensorflow\python\framework\dtypes.py:528: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a fu
,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
E:\python3.6\lib\site-packages\tensorflow\python\framework\dtypes.py:529: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a fu
,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
E:\python3.6\lib\site-packages\tensorflow\python\framework\dtypes.py:530: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a fu
,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
E:\python3.6\lib\site-packages\tensorflow\python\framework\dtypes.py:535: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a fu
,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])

WARNING: The TensorFlow contrib module will not be included in TensorFlow 2.0.
For more information, please see:
  * https://github.com/tensorflow/community/blob/master/rfcs/20180907-contrib-sunset.md
  * https://github.com/tensorflow/addons
If you depend on functionality not listed there, please file an issue.

E:\python3.6\lib\site-packages\onnx_tf\common\__init__.py:87: UserWarning: FrontendHandler.get_outputs_names is deprecated. It will be removed in future release..
  warnings.warn(message)
D save dump info to: ./build.log
--> Loading model
```

New environment variables ("PYTHONLEGACYWINDOWSSTDIO") are required when using RKNN Toolkit 1.2.0 on Windows systems, and it should be set with value of 1.

It can also be upgraded to RKNN Toolkit 1.2.1 and later versions, which do not require manual setting of the environment variables.

1.8.3 How to view the detailed log and save it in the log file?

There are two methods:

1. When initializing the RKNN object, set the verbose parameter to True, and set verbose_file to the specified file path;
2. When initializing the RKNN object, set the verbose parameter to True, and redirect the

log to a file when executing the python script, such as "python test.py> test.log 2>&1".

If you want to view the detailed log on the development board, you can connect the serial port to the board and execute the following two commands before online debugging:

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

2 Questions related with quantization accuracy

2.1 The accuracy doesn’t match with original model after quantization, how to debug?

- **Firstly make sure the accuracy of float type is similar to test result of original platform:**

(1) Make rknn.build(do_quantization=False) when the quantized model is loaded by RKNN Toolkit.

(2) Refer to 1.1 to set **channel_mean_value** parameter, which should be same as the parameter used for training model.

(3) Make sure the sequence of the input image channel must be R,G,B while testing. (Whatever the sequence of the image channel is used for training, it must be input by R,G,B while using RKNN to do testing)

(4) Set **reorder_channel** parameter in **rknn.config** function, ’0 1 2’ stands for RBG, ’2 1 0’ stands for BGR, and it must be consistent with the sequence of the image channel used for training.

- **Accuracy test after quantization**

(1) Use multiple pictures to do quantization, to ensure the stability of quantization accuracy.

Set batch_size parameter in rknn.config (recommend to set batch_size = 200) and provide more than 200 images path in dataset.txt for quantization.

If the display memory is not enough, you can set batch_size =1, epochs=200 instead of batch_size = 200 for quantization.

(2) Accuracy comparison, try to use relatively big data set to do testing. Compare the accuracy of top-1, top-5 for classifying network, compare mAP, Recall of data set for checking network, and so on.

(3) If it is face recognition, the results of the float model and the results of the quantitative model cannot be used for feature comparison. For example, two pictures of A1 and A2, the results of using the float model are A1fp, A2fp, the results of running with the quantitative model are A1u8, A2u8. At this time, the Euclidean distance of A1fp and A2fp can be calculated to calculate the similarity of two pictures, and the Euclidean distance of A1u8 and A2u8 can also be calculated to calculate the similarity of two pictures. But can't calculate the Euclidean distance of A1fp and A2u8 to calculate the similarity of two pictures.

- **Floating point model results are incorrect**

1) At present, the PC simulator can support dumping the data of each layer of the network. Before executing the inference script, you need to set an environment variable. The command is as follows: export NN_LAYER_DUMP=1.

2) After execution, the tensor data file of each layer of the network will be generated in the current directory, so that it can be compared with the data of the original framework layer by layer. Note: Some layers will be merged. For example, conv+bn+scale will be merged into a conv. At this time, it needs to be compared with the output of the scale layer of the original model. Please first check whether the mean_values / std_values / reorder_channel in the config is set correctly; whether the image is processed in other ways before inference.

3) If the result of the last layer is all 0, pay attention to the serial port log to see if there is a GPU Hang message, and feed the model back to the Rockchip NPU team for analysis. If there is an error in the middle layer, the parameters, input and output information of the layer can be fed back to the Rockchip NPU team for further analysis. If you find that the results of the first layer are very different when comparing the data, it is usually because the input preprocessing is not set correctly.

4) The intermediate results of each layer of model inference can also be saved on development boards such as RK1808/RV1109/RV1126. The specific method is as

follows: The serial port is connected to the development board, and a directory(such as dump_data) is created under the /userdata directory (the directory space is generally larger) to save the intermediate results. Then enter the directory and set the environment variable (export NN_LAYER_DUMP=1). If you are debugging through a PC connected to the board, you need to restart rknn_server by executing the restart_rknn.sh command. After making the above settings and then running the model inference program, you can dump the intermediate results of each layer. For the RK3399Pro board, you need to connect the NPU serial port on the board. For the specific location, please refer to the location marked with a red box in the figure below:

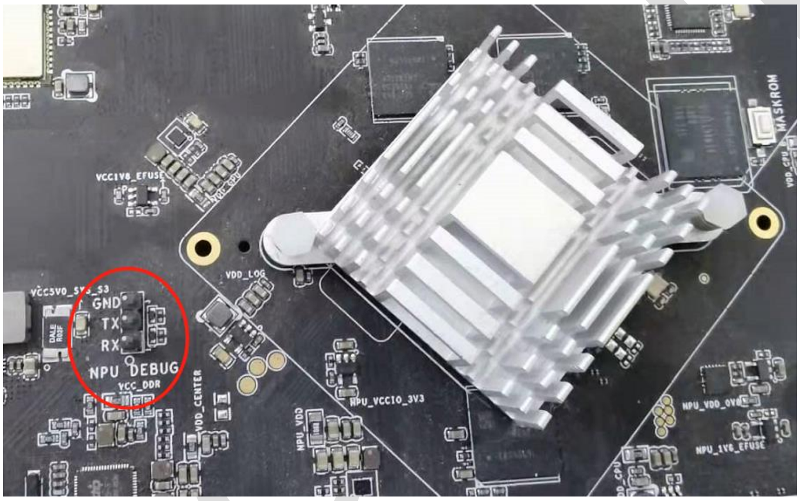


Figure 2-1-1 Serial port of NPU on RK3399Pro

- **Quantized model results are incorrect**

1) Check if the floating-point model is correct or not, please follow the previous section for troubleshooting; if correct, proceed to the next step.

2) Use the accuracy_analysis interface to perform accuracy analysis to find out the layer that caused the decrease in accuracy. You can also manually dump the results of each layer, and compare the results to find the layer with reduced accuracy.

3) There are three situations for these layers with reduced accuracy. First, there is a problem with the driver implementation (the quantization result usually has a huge gap with the floating-point result at this time); the second is that the layer itself is not friendly to quantization; it may also be that some model optimizations lead to a decrease in accuracy, such as replacing add or average pool with conv etc. For the first two

scenarios, you can try to circumvent it with a hybrid quantization method; for the third scenario, you can also try to lower the optimization level (the method of lowering is to set the optimization_level to 2 in the config interface).

Note: For the usage of accuracy analysis and hybrid quantization, please refer to the document <Rockchip_User_Guide_RKNN_Toolkit_EN.pdf>.

2.2 How to dump the output of each layer of network

Refer to the previous question.

2.3 Which frameworks` quantized model are currently supported by the RKNN Toolkit?

The RKNN Toolkit currently supports quantized models of the two frameworks TensorFlow and TensorFlow Lite.

2.4 What format is the file saved during the accuracy analysis? In what order? Is it possible to specify the layout of the data?

It is a one-dimensional numpy array, which can be loaded by numpy.loadtxt interface.

The data is storage in the layout of NHWC. And the layout cannot be specified currently.

3 Common issues of Caffe model conversion

3.1 “Deprecated caffe input usage” error occurs during model conversion

It means this model is old version of caffe mode. Need to change input layer into below format.

```
    layer {
        name: "data"
        type: "Input"
        top: "data"
        input_param {
            shape {
                dim: 1
                dim: 3
                dim: 224
                dim: 224
            }
        }
    }
```

3.2 “Message type "caffe.PoolingParameter" has no field named "round_mode"”error occurs during model conversion

round_mode field of Pool layer cannot be recognized, you can change it to ceil_model. For example, if originally it is round_mode: CEIL, then you can delete (ceil_mode is True by default) or change to ceil_mode:True.

3.3 “ValueError("'%s' is not a valid scope name" % name)” error occurs during caffe or other model conversion

The detailed error log is as below:

```
T         raise ValueError("'%s' is not a valid scope name" % name)
T    ValueError: '_plus0_17' is not a valid scope name
```

In this case, it is because layer name '_plusxxx' is not allowed to use _ at the beginning. Need to follow the naming rule of tensorflow:

[A-Za-z0-9.][A-Za-z0-9_.\\-/]* (for scopes at the root)

[A-Za-z0-9_.\\-/]* (for other scopes)

3.4 “Invalid tensor id(1), tensor(@mbox_conf_flatten_188:out0) ” error occurs when Caffe version SSD conversion fails

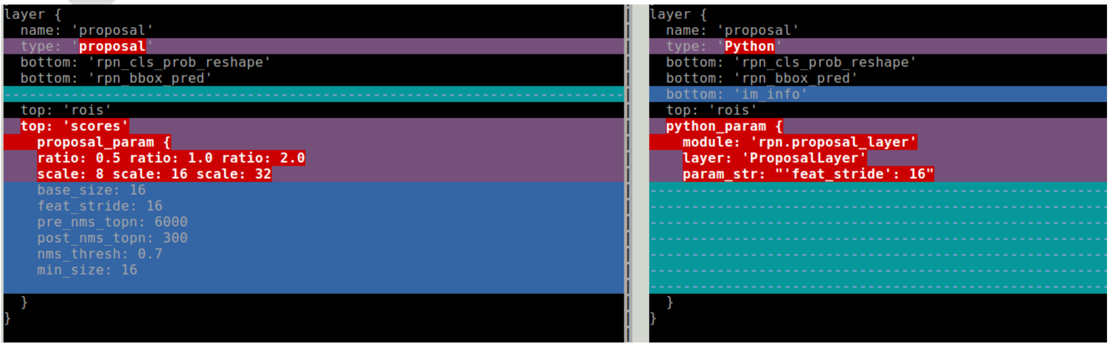
Not support detectionoutput layer, you can delete and then change to CPU.

3.5 There should be three output tensor after Caffe version SSD model deletes detectionoutput, but actually only return two tensor by RKNN inference

The missing tensor is priori box. It is the same during training and inference stage, and for all inputs. In order to improve performance, RKNN Toolkit optimized the relative layer in the model. If want to get the tensor of priori box, you can save the tensor of priori box, or use Caffe to do inference once in training stage.

3.6 “ValueError: Invalid tensor id(1), tensor(@rpn_bbox_pred_18:out0) ” error occurs during py-faster-rcnn model conversion

Comparing with official code, need to change 'proposal' layer of prototxt as below:



```
layer {
  name: 'proposal'
  type: 'proposal'
  bottom: 'rpn_cls_prob_reshape'
  bottom: 'rpn_bbox_pred'
  top: 'rois'
  top: 'scores'
    proposal_param {
    ratio: 0.5 ratio: 1.0 ratio: 2.0
    scale: 8 scale: 16 scale: 32
    base_size: 16
    feat_stride: 16
    pre_nms_topn: 6000
    post_nms_topn: 300
    nms_thresh: 0.7
    min_size: 16
  }
}
```

```
layer {
  name: 'proposal'
  type: 'Python'
  bottom: 'rpn_cls_prob_reshape'
  bottom: 'rpn_bbox_pred'
  bottom: 'im_info'
  top: 'rois'
  python_param {
    module: 'rpn.proposal_layer'
    layer: 'ProposalLayer'
    param_str: "'feat_stride': 16"
  }
}
```

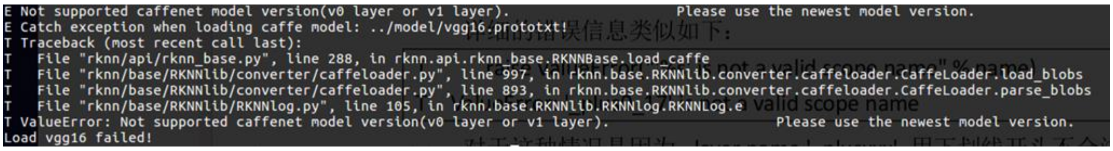
```
layer {
  name: 'proposal'
```

```
        type: 'proposal'
        bottom: 'rpn_cls_prob_reshape'
        bottom: 'rpn_bbox_pred'
        top: 'rois'
        top: 'scores'
          proposal_param {
          ratio: 0.5 ratio: 1.0 ratio: 2.0
          scale: 8 scale: 16 scale: 32
          base_size: 16
          feat_stride: 16
          pre_nms_topn: 6000
          post_nms_topn: 300
          nms_thresh: 0.7
          min_size: 16
        }
      }
```

3.7 Does RKNN Toolkit support upsample of Caffe?

Upsample before 1.4.0 needs to be replaced by Resize, after 1.4.0, the Upsample layer is directly supported.

3.8 “E Not supported caffenet model version(v0 layer or v1 layer)” error occurs during model conversion



```
E Not supported caffenet model version(v0 layer or v1 layer).          Please use the newest model version.
E Catch exception when loading caffe model: ../model/vgg16.prototxt!
T Traceback (most recent call last):
T   File "rknn/api/rknn_base.py", line 288, in rknn.api.rknn_base.RKNNBase.load_caffe
T   File "rknn/base/RKNNlib/converter/caffeloader.py", line 997, in rknn.base.RKNNlib.converter.caffeloader.CaffeLoader.load_blobs
T   File "rknn/base/RKNNlib/converter/caffeloader.py", line 893, in rknn.base.RKNNlib.converter.caffeloader.CaffeLoader.parse_blobs
T   File "rknn/base/RKNNlib/RKNNlog.py", line 105, in rknn.base.RKNNlib.RKNNlog.RKNNLog.e
T ValueError: Not supported caffenet model version(v0 layer or v1 layer).          Please use the newest model version.
Load vgg16 failed!
```

The main reason is that the version of the caffe model is too old and needs to be updated. The update method is as follows (take VGG16 as an example):

1) Download Caffe source code from https://github.com/BVLC/caffe.git

2) Compile Caffe

3) Convert the model to a new format

```
./build_release/tools/upgrade_net_proto_text vgg16_old/vgg16.prototxt vgg16_new/vgg16.prototxt
./build_release/tools/upgrade_net_proto_binary vgg16_old/vgg16.caffemodel vgg16_new/vgg16.caffemodel
```

4 Common issues of Tensorflow model conversion

4.1 “AttributeError: ‘NoneType’ object has no attribute op” error occurs during Google official ssd_mobilenet_v2 model conversion

One possible reason is that input node is not correct. You can modify as below:

```
rknn.load_tensorflow(tf_pb='./ssd_mobilenet_v2_coco_2018_03_29/frozen_inference_graph.pb',
                     inputs=['FeatureExtractor/MobilenetV2/MobilenetV2/input'],
                     outputs=['concat', 'concat_1'],
                     input_size_list=[[INPUT_SIZE, INPUT_SIZE, 3]])
```

4.2 “Cannot convert value dtype ([‘resource’, ‘u1’]) to a Tensorflow Dtype” error occurs during SSD_Resnet50_v1_FPN_640x640 model conversion

Need to update RKNN Toolkit to version 0.9.8 or higher.

4.3 On RKNN Toolkit 1.0.0，is the output shape of RKNN model converted from TensorFlow changed?

Versions prior to 1.0.0 will convert output shape from "NHWC" to "NCHW". Starting from this version, the shape of the output will be consistent with the original model, and no longer convert from "NHWC" to "NCHW". Please pay attention to the location of the channel when performing post processing.

4.4 When converting the model, the error "E ValueError: NodeDef mentions attr 'explicit_paddings' not in Op <name = Conv2D;" appears.

This is because the version of tensorflow used in the model training is greater than or equal to the 1.14.0 version, and when converted to the rknn model, the tensorflow version used is less than or equal to the 1.13.2 version. The solution is to keep the training model and the converted rknn model using the same tensorflow version. For example, either use version >= 1.14.0, or use version <= 1.13.2.

4.5 When loading the model, it prompts the module “RKNNlib.convertert.lite.tflite” has no attribute ‘TransposeOptions’

The RKNN Toolkit version 1.3.2 and earlier does not support the Transpose operation of TFLite. Need to update to version 1.4.0 or later.

5 Common issues of Pytorch model conversion

Before version 1.3.0, RKNN Toolkit indirectly supports Pytorch through ONNX, so need to convert Pytorch to ONNX first. Start from version 1.3.0, RKNN Toolkit supports loading Pytorch model directly. If issue occurs during conversion, please update RKNN Toolkit to the latest version first.

5.1 How to solve the problem of "torch._C" has no attribute '_jit_pass_inline' when loading the Pytorch model?

Please update version of torch to 1.5.0 or 1.5.1 or 1.6.0.

5.2 When using the load_pytorch interface to load a pytorch model, can the pytorch model only use the model saved by torch.jit.trace? Can the model saved with torch.save(net.state_dict()) be used?

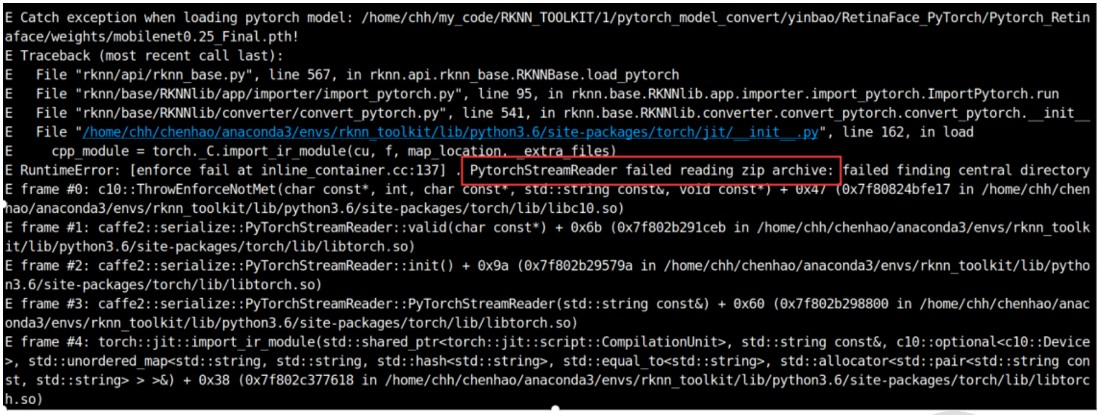
Only models saved in torch.jit.trace can be used. The torch.save() saves only the parameters of the model, and does not have a network structure. With only one network parameter, RKNN Toolkit cannot construct the corresponding network.

5.3 "assert(tsr.op_type == 'Constant') " error occurs during conversion

This issue is introduced after pytorch 0.4.5 version. In your model, if there is something like "x = x.view(x.size(0), -1)" , need to change to "x = x.view(int(x.size(0)), -1)" .

5.4 "PytorchStreamReader failed reading zip archive" error occurs during conversion

The detailed error is as follows:



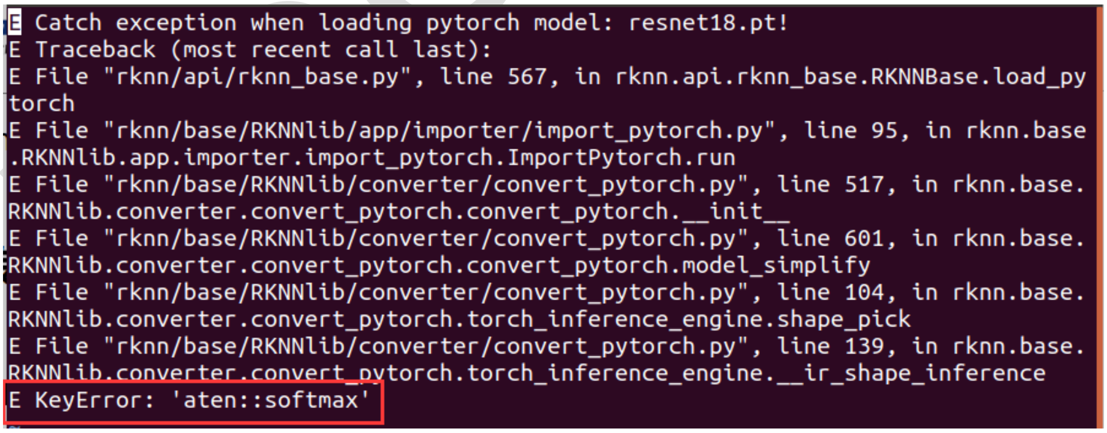
```
E Catch exception when loading pytorch model: /home/chh/my_code/RKNN_TOOLKIT/1/pytorch_model_convert/yinbao/RetinaFace_PyTorch/Pytorch_Retin
aface/weights/mobilenet0.25_Final.pth!
E Traceback (most recent call last):
E   File "rknn/api/rknn_base.py", line 567, in rknn.api.rknn_base.RKNNBase.load_pytorch
E   File "rknn/base/RKNNlib/app/importer/import_pytorch.py", line 95, in rknn.base.RKNNlib.app.importer.import_pytorch.ImportPytorch.run
E   File "rknn/base/RKNNlib/converter/convert_pytorch.py", line 541, in rknn.base.RKNNlib.converter.convert_pytorch.convert_pytorch.__init__
E   File "/home/chh/chenhao/anaconda3/envs/rknn_toolkit/lib/python3.6/site-packages/torch/jit/__init__.py", line 162, in load
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: failed finding central directory
E frame #0: c10::ThrowEnforceNotMet(char const*, int, char const*, std::string const&, void const*) + 0x47 (0x7f80824bfe17 in /home/chh/chen
hao/anaconda3/envs/rknn_toolkit/lib/python3.6/site-packages/torch/lib/libc10.so)
E frame #1: caffe2::serialize::PyTorchStreamReader::valid(char const*) + 0x6b (0x7f802b291ceb in /home/chh/chenhao/anaconda3/envs/rknn_toolk
it/lib/python3.6/site-packages/torch/lib/libtorch.so)
E frame #2: caffe2::serialize::PyTorchStreamReader::init() + 0x9a (0x7f802b29579a in /home/chh/chenhao/anaconda3/envs/rknn_toolkit/lib/pytho
n3.6/site-packages/torch/lib/libtorch.so)
E frame #3: caffe2::serialize::PyTorchStreamReader::PyTorchStreamReader(std::string const&) + 0x60 (0x7f802b298800 in /home/chh/chenhao/anac
onda3/envs/rknn_toolkit/lib/python3.6/site-packages/torch/lib/libtorch.so)
E frame #4: torch::jit::import_ir_module(std::shared_ptr<torch::jit::script::CompilationUnit>, std::string const&, c10::optional<c10::Device
>, std::unordered_map<std::string, std::string, std::hash<std::string>, std::equal_to<std::string>, std::allocator<std::pair<std::string con
st, std::string> > >&) + 0x38 (0x7f802c377618 in /home/chh/chenhao/anaconda3/envs/rknn_toolkit/lib/python3.6/site-packages/torch/lib/libtorc
h.so)
```

This error is because the model you are converting contains only weights but no network structure.

Usually the pth file contains only weights and no network structure information. The correct step is to define a net, then load the pth weights with net.load_state_dict(), and finally use torch.jit.trace() to freeze the network structure and weights into a pt file, and then use rknn.load_pytorch() to convert this pt file. For details, please refer to examples / pytorch / resnet18. Usually it can’t be converted if you only has a pth file.

5.5 “E KeyError: 'aten::xxx'”error occurs during conversion

The detailed error is as follows:

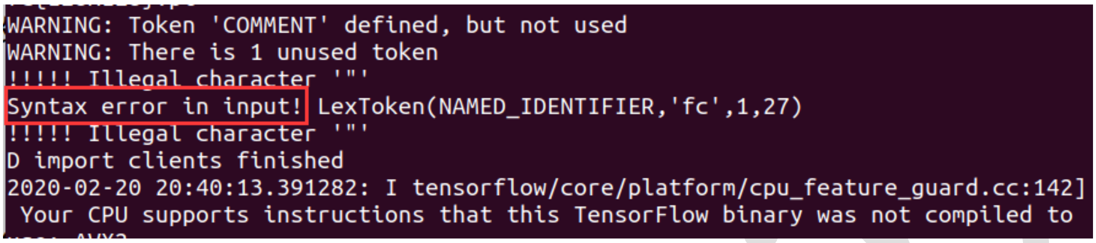


```
E Catch exception when loading pytorch model: resnet18.pt!
E Traceback (most recent call last):
E File "rknn/api/rknn_base.py", line 567, in rknn.api.rknn_base.RKNNBase.load_py
torch
E File "rknn/base/RKNNlib/app/importer/import_pytorch.py", line 95, in rknn.base
.RKNNlib.app.importer.import_pytorch.ImportPytorch.run
E File "rknn/base/RKNNlib/converter/convert_pytorch.py", line 517, in rknn.base.
RKNNlib.converter.convert_pytorch.convert_pytorch.__init__
E File "rknn/base/RKNNlib/converter/convert_pytorch.py", line 601, in rknn.base.
RKNNlib.converter.convert_pytorch.convert_pytorch.model_simplify
E File "rknn/base/RKNNlib/converter/convert_pytorch.py", line 104, in rknn.base.
RKNNlib.converter.convert_pytorch.torch_inference_engine.shape_pick
E File "rknn/base/RKNNlib/converter/convert_pytorch.py", line 139, in rknn.base.
RKNNlib.converter.convert_pytorch.torch_inference_engine.__ir_shape_inference
E KeyError: 'aten::softmax'
```

The reason is that this op did not match. We will fix these bugs with every version update. Please use our latest rknn_toolkit.

5.6 “Syntax error in input! LexToken(xxx)”error occurs during conversion

The detailed error is as follows:



```
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 '"'
D import clients finished
2020-02-20 20:40:13.391282: I tensorflow/core/platform/cpu_feature_guard.cc:142]
 Your CPU supports instructions that this TensorFlow binary was not compiled to
use: AVX2
```

There are many reasons for this error, please troubleshoot in the following order:

1. Use pytorch 1.2.0. We recommend using pytorch 1.2.0 because our tests are based on 1.2.0. There may be some unknown errors if using higher versions. If it is RKNN Toolkit 1.6.0 or later, you can use Pytorch version 1.5.0 or 1.6.0.

2. Torch.nn.module is not inherited when create a network. Please inherit torch.nn.module to create a network, and then use torch.jit.trace to generate a pt file.

3. Try to upgrade to RKNN Toolkit 1.6.0 version, and use torch as convert_engine, and the torch version should also be upgraded to 1.5.0 or above.

6 Common issues of ONNX model conversion

6.1 Encountered the error "Your model ir_version is higher than the chacker`s" during conversion

RKNN Toolkit 1.1.0 and earlier versions only support models exported by ONNX from version 1.3.2 and below. RKNN Toolkit 1.4.0 and earlier versions support models exported by ONNX 1.4.1 and below; RKNN Toolkit 1.6.0 supports models exported by ONNX version 1.6.0.

7 RKNN convolution acceleration tips

7.1 How to design a convolutional neural network to achieve optimal performance on RKNN

Here are some suggestions from us：

1. Optimal Kernel Size is 3x3

Convolution cores can support a large range of kernel sizes. The minimum supported kernel size is [1] and maximum is [11 * stride - 1].

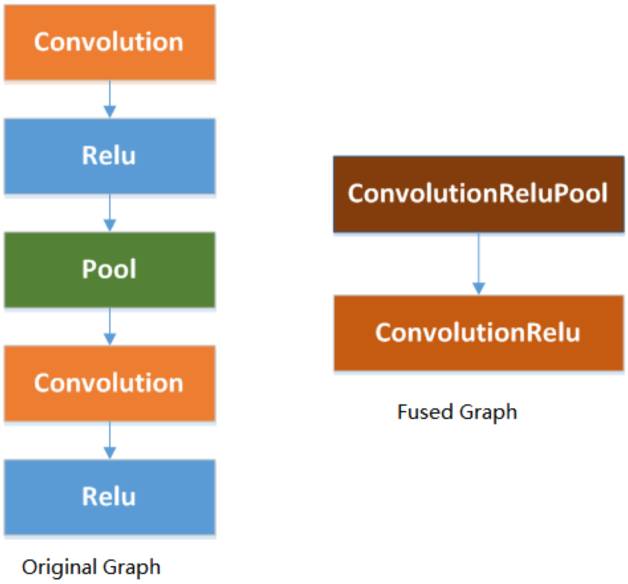
The NN Engine performs most optimally when the Convolution kernel size is 3x3, under which the highest MAC utilization can be achieved.

Non-square kernels are also supported, but with some computation overhead.

2. Fused Operations Reduce Overhead

The Convolution core can fuse ReLU and MAX Pooling operations on the fly to further reduce computation and bandwidth overhead. A ReLU layer following a Convolution layer will always be fused, while MAX pooling layer fusion has the following restrictions, Max pooling must

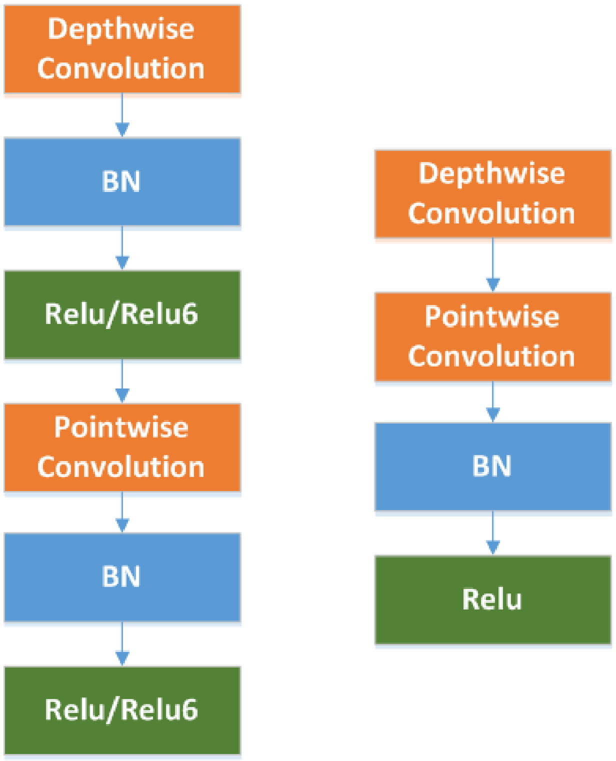
- have a pool size of 3x3 or 2x2, and stride of 2
- 2x2 pooling must have an even input size and no padding
- 3x3 pooling must have odd input size which is not one and no padding
- Horizontal input size must be less than 64 (8-bit mode) or 32 (16-bit mode) if pool size is 3x3



3. Depthwise Convolutions

Both regular 2D and Depthwise convolutions are supported, while 2D convolutions perform more optimally. Since Depthiwise Convolution-specific structure makes it less friendly to quantized model. It`s recommend to use 2D convolution whenever possible when designing your network.

If you must use a Depthwise convolution, it`s recommend to follow the rules below that can improve the accuracy of the quantized model:



- Change the activation function RELU6 to RELU.
- Remove the BN layer and activation layer of the Depthwise convolution layer.
- In training, for the Depthwise convolutional layer, L2 regularization of its weight.

4. Output channel number setting

It`s recommend to set the number of convolution output channels to be a multiple of the number of convolution kernels in the NPU to ensure that all convolution kernels are better utilized for higher hardware utilization.

5. Take advantage of Hardware's Sparse Matrix Support

Modern Neural-Networks are known to be over parameterized and have much redundancy in their design. Pruning a network to be sparse has been proven to reduce computation overhead while maintaining accuracy.

RKNN hardware is designed to support sparse matrix operations efficiently by skipping computations and memory fetches on zero values. The sparsity level can be fine grain down to individual weights. Designing a sparse network to take advantage of this technology could further improve performance on RKNN.

6. Dilation Convolution

When using TensorFlow to create a convolution with dilations parameter, use tf.nn.atrous_conv2d to create it. Currently the rknn-toolkit don’t support the direct use of tf.nn.conv2d to create convolution with dilations parameter, otherwise the inference results will be wrong.

In Addition, the version of TensorFlow need to be higher than 1.14.0, and the version of rknn-toolkit need to be higher than v.1.4.0.