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

RKNN Toolkit Lite2 User Guide

(Technology Department, Graphic Computing Platform Center)

Mark: [ ] Editing [√] Released	Version	V1.5.2
	Author	Rao Hong
	Completed Date	2023-08-21
	Reviewer	Vincent
	Reviewed Date	2023-08-21

Revision History

Version	Modifier	Date	Modify description	Reviewer
V1.2.0	Rao Hong	2022-01-13	Initial version.	Vincent
V1.3.0	Rao Hong	2022-04-27	Update version.	Vincent
V1.4.0	Rao Hong	2022-08-30	Update the instruction for core_mask parameter of the init_runtime interface.	Vincent
V1.5.0	Rao Hong	2023-05-18	1. Update applicable chip, add RK3562; 2. Support new Python version: 3.8, 3.10 on ARM64 platform.	Vincent
V1.5.2	Rao Hong	2023-08-21	Update version	Vincent

Table of Contents

1 Overview

RKNN Toolkit Lite2 provides Python programming interfaces for Rockchip NPU platform to help users deploy RKNN models and accelerate the implementation of AI applications.

1.1 Applicable chip

- RK3562
- RK3566
- RK3568
- RK3588 / RK3588S

Note: In the document, RK3588 is used to refer to RK3588 and RK3588S

1.2 Applicable system

- Debian: 10 (aarch64)
- Debian: 11 (aarch64)

2 Develop environment build

2.1 Requirements/Dependencies

This software development kit supports running on the Debian operating system. It is recommended to meet the following requirements in the operating system environment:

Table 1 Operating system environment

Operating system version	Debian 10 or 11
Python version	3.7 / 3.8 / 3.9 / 3.10
Python library dependencies	'numpy' 'ruamel.yaml' 'psutils'

2.2 Installation

RKNN Toolkit Lite2 can currently be installed via the pip3 install command.

1. If python3 and pip3 are not installed in the system, please install them through apt-get first. The command is as follows

```
sudo apt-get update
sudo apt-get install -y python3 python3-dev python3-pip gcc
```

Note: When installing some dependent modules, the source code needs to be compiled, so the two packages python3-dev and gcc are also installed together to avoid compilation failure when installing dependent modules later.

2. Install dependent libraries: opencv-python and numpy

```
sudo apt-get install -y python3-opencv
sudo apt-get install -y python3-numpy
```

Note:

1） RKNN Toolkit Lite2 itself does not rely on opencv-python, but the example will use this library to load image, so the library is also installed here.

2） It may fail to install numpy directly through pip3 on Debian10 firmware. It is recommended to use the above method to install numpy.

3. Install RKNN Toolkit Lite2

The installation packages of each platform are placed in the *SDK/packages* folder. Enter *SDK/packages* folder and execute command below to install RKNN Toolkit Lite2:

```
# Python 3.7
pip3 install rknn_toolkit_lite2-1.x.y-cp37-cp37m-linux_aarch64.whl
# Python 3.8
pip3 install rknn_toolkit_lite2-1.x.y-cp38-cp38-linux_aarch64.whl
# Python 3.9
pip3 install rknn_toolkit_lite2-1.x.y-cp39-cp39-linux_aarch64.whl
# Python 3.10
pip3 install rknn_toolkit_lite2-1.x.y-cp310-cp310-linux_aarch64.whl
```

3 Usage

RKNN Toolkit Lite2 is mainly used for the deployment of RKNN models on Rockchip NPU platforms.

Before using RKNN Toolkit Lite2, users need to use RKNN Toolkit2 to convert the models of various deep learning frameworks to RKNN models.

The complete installation package and usage documents of RKNN Toolkit2 can be obtained from the following link:

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

3.1 Basic usage process

The basic process of deploying the RKNN model using RKNN Toolkit Lite2 is shown in the following figure：

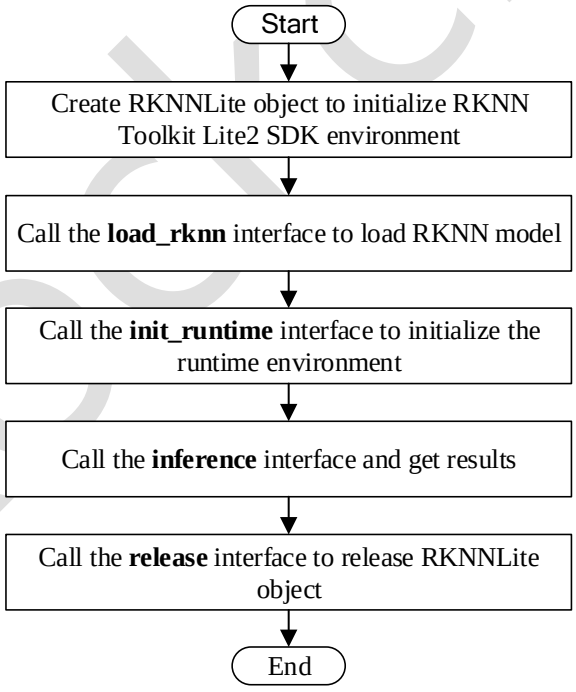

Figure 3-1-1 Usage flow of RKNN Toolkit Lite2

Note:

1. Before calling the inference interface for inference, it is necessary to obtain the input data, do the corresponding preprocessing, and then set the data_type, data_format and other parameters in the inference interface according to the input information;

2. After calling the inference interface, the inference results are usually processed accordingly to complete the upper-layer application related functions.

3.2 Example

An example inference_with_lite for model inference using RKNN Toolkit Lite2 is provided in the SDK/examples directory. Executing this example will load the Resnet18 model and perform inference to get the top5 classification result of the example image.

How to run the example:

1. Prepare a development board with RKNN Toolkit Lite2 installed;

2. Push the example to the development board;

3. Go to the examples/inference_with_lite directory on the development board and execute the following command to run the example:

```
python3 test.py
```

When performing model inference, the result of this demo is as follows:

```
-----TOP 5-----
[812]: 0.9996383190155029
[404]: 0.00028062614728696644
[657]: 1.6321087969117798e-05
[833 895]: 1.015903580992017e-05
[833 895]: 1.015903580992017e-05
```

4 RKNN Toolkit Lite2 API description

4.1 RKNNLite object initialization and release

The initialization/release function group consists of API interfaces to initialize and release the RKNNLite object as needed. The **RKNNLite()** must be called before using all the API interfaces of RKNN Toolkit Lite, and call the **release()** method to release the object when task finished.

When we init RKNNLite object, we can set ***verbose*** and ***verbose_file*** parameters, used to show detailed log information of model loading, inferencing and so on. The data type of verbose parameter is bool. If we set the value of this parameter to True, the RKNN Toolkit Lite will show detailed log information on screen. The data type of verbose_file is string. If we set the value of this parameter to a file path, the detailed log information will be written to this file (**the verbose also need be set to True**).

The sample code is as follows:

```
# Show the detailed log information on screen, and saved to
# mobilenet_build.log
rknn_lite = RKNNLite(verbose=True, verbose_file='./inference.log')
# Only show the detailed log information on screen.
rknn_lite = RKNNLite(verbose=True)
…
rknn_lite.release()
```

4.2 Loading RKNN model

API	**load_rknn**
Description	Load RKNN model
Parameter	**path**: The path of RKNN model file.
Return Value	0: Load successfully
	-1: Load failed

The sample code is as follows:

```
# Load the resnet_18 RKNN model in the current path
ret = rknn_lite.load_rknn('./resnet_18.rknn')
```

4.3 Initialize the runtime environment

Before inference or performance evaluation, the runtime environment must be initialized. This interface determines which type of runtime hardware is specified to run model.

API	**init_runtime**
Description	Initialize the runtime environment.
	core_mask: set NPU working core mode. The optional values are as follows: **RKNNLite.NPU_CORE_AUTO**: Automatic scheduling mode, automatically running on the currently idle NPU core. **RKNNLite.NPU_CORE_0**: Running on NPU core 0. **RKNNLite.NPU_CORE_1**: Running on NPU core 1. **RKNNLite.NPU_CORE_2**: Running on NPU core 2. **RKNNLite.NPU_CORE_0_1**：Runing on NPU core 0 and core 1. **RKNNLite.NPU_CORE_0_1_2**：Runing on NPU core 0 and core 1 and core 2. Default value is RKNNLite.NPU_CORE_AUTO. Note: this parameter is valid for RK3588.
Return Value	0: Initialize the runtime environment successfully
	-1: Initialize the runtime environment failed

The sample code is as follows:

```
# Initialize the runtime environment
ret = rknn_lite.init_runtime(core_mask=RKNNLite.NPU_CORE_AUTO)
if ret != 0:
    print('Init runtime environment failed')
    exit(ret)
```

4.4 Inference with RKNN model

Before using the model for inference, you must load a RKNN model first.

API	**inference**
Description	Use the model to perform inference with specified input and get the inference result.
Parameter	**inputs**: Inputs to be inferred, such as images processed by cv2. The object type is list. List members are ndarray.
	data_format: The shape format of input data. Optional values are "nhwc", only valid for 4-dims input. The default value is None, indicates that all input layouts are NHWC.
Return Value	results: The result of inference, the object type is list. List members are ndarray.

The sample code is as follows:

```
# Preform inference for a picture with a model and get a top-5 result
……
outputs = rknn_lite.inference(inputs=[img])
show_top5(outputs)
……
```

The result of top-5 is as follows:

```
-----TOP 5-----
[812]: 0.999442994594574
[404]: 0.0004096269840374589
[657]: 3.284541890025139e-05
[833]: 2.6112385967280716e-05
[895]: 1.8509887013351545e-05
```

4.5 Get SDK version

API	**get_sdk_version**
Description	Get API version and driver version of referenced SDK. Note: Before we use this interface, we must load model and initialize runtime first.
Parameter	None
Return Value	sdk_version：API and driver version. Data type is string.

The sample code is as follows：

```
# Get SDK version
……
sdk_version = rknn_lite.get_sdk_version()
……
```

The SDK version looks like below：

```
I RKNN: [10:47:23.097] RKNN Runtime Information: librknnrt version: 1.2.0b1 (a47985372@2021-12-06T10:20:14)
I RKNN: [10:47:23.097] RKNN Driver Information: version: 0.6.2
I RKNN: [10:47:23.098] RKNN Model Information: version: 1, toolkit version: 1.1.2(compiler version: 1.1.2b13 (1e5726f57@2021-11-29T14:13:55)), target: RKNPU v2, target platform: rk3588, framework name: PyTorch, framework layout: NCHW
```

4.6 Query RKNN model runnable platform

API	**list_support_target_platform**
Description	Query the chip platform that a given RKNN model can run on.
Parameter	**rknn_model**: RKNN model path. If the model path is not specified, the chip platforms currently supported by the RKNN Toolkit Lite2 are printed by category
Return Value	support_target_platform (dict): Return the chip platform where the model can run. If the RKNN model path is empty or does not exist, return None

The sample code is as follows:

```
rknn_lite.list_support_target_platform(rknn_model='resnet18.rknn')
```

The runnable chip platforms look like below：

```
**************************************************
Target platforms filled in RKNN model:              ['RK3588']
Target platforms supported by this RKNN model: ['RK3588']
**************************************************
```

5 Appendix

5.1 Reference documents

For RKNN Toolkit2 model conversion related documents, please refer to the following link:

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

5.2 Issue feedback

All the issue can be feedback via the follow ways:

- RKNN QQ group: 1025468710
- Github link: https://github.com/rockchip-linux/rknn-toolkit2/issues
- Rockchip Redmine: https://redmine.rock-chips.com/

Note: Redmine account can only be registered by an authorized salesperson. If your development board is from the third-party manufacturer, please contact them to report the issue.