密级状态：绝密(　　)　　秘密(　　)　　内部(　)　　公开(√)

RKNN Toolkit 快速上手指南

（技术部，图形显示平台中心）

文件状态： [　] 正在修改 [√] 正式发布	当前版本：	V1.3.0
	作　　者：	饶洪
	完成日期：	2019-12-23
	审　　核：	卓鸿添
	完成日期：	2019-12-23

福州瑞芯微电子股份有限公司

Fuzhou　Rockchips　Semiconductor　Co . , Ltd

更新记录

版本	修改人	修改日期	修改说明	核定人
V0.9.9	饶洪	2019-03-25	初始版本	卓鸿添
V1.0.0	饶洪	2019-05-08	同步 RKNN-Toolkit-V1.0.0 修改内容	卓鸿添
V1.1.0	饶洪	2019-06-28	1. 同步 RKNN-Toolkit-V1.1.0 修改内容 2. 新增 Windows/MacOS/ARM64 等平台的快速上手指南	卓鸿添
V1.2.0	饶洪	2019-08-21	同步 RKNN-Toolkit-V1.2.0 修改内容	卓鸿添
V1.2.1	饶洪	2019-09-26	同步 RKNN-Toolkit-V1.2.1 修改内容	卓鸿添
V1.3.0	饶洪	2019-12-23	同步 RKNN-Toolkit-V1.3.0 修改内容	卓鸿添

目 录

1 主要功能说明

RKNN-Toolkit 是为用户提供在 PC、RK3399Pro、RK1808、TB-RK1808 AI 计算棒或 RK3399Pro Linux 开发板上进行模型转换、推理和性能评估的开发套件，用户通过提供的 python 接口可以便捷地完成以下功能：

1) 模型转换：支持 Caffe、TensorFlow、TensorFlow Lite、ONNX、Darknet、Pytorch、MXNet 模型转成 RKNN 模型，支持 RKNN 模型导入导出，后续能够在硬件平台上加载使用。从 1.2.0 版本开始支持多输入模型。从 1.3.0 版本开始支持 Pytorch 和 MXNet，这两个功能目前还是实验性功能。
2) 量化功能：支持将浮点模型转成量化模型，目前支持的量化方法有非对称量化（asymmetric_quantized-u8），动态定点量化（dynamic_fixed_point-8 和 dynamic_fixed_point-16）。从 1.0.0 版本开始，RKNN-Toolkit 开始支持混合量化功能。
3) 模型推理：能够在 PC 上模拟运行模型并获取推理结果；也可以在指定硬件平台 RK3399Pro（或 RK3399Pro Linux 开发板）、RK1808、TB-RK1808 AI 计算棒上运行模型并获取推理结果。
4) 性能评估：能够在 PC 上模拟运行并获取模型总耗时及每一层的耗时信息；也可以通过联机调试的方式在指定硬件平台 RK3399Pro、RK1808、TB-RK1808 AI 计算棒上运行模型，或者直接在 RK3399Pro Linux 开发板上运行，以获取模型在硬件上完整运行一次所需的总时间和每一层的耗时情况。
5) 内存评估：评估模型运行时对系统和 NPU 内存的消耗情况。通过联机调试的方式获取模型在指定硬件平台 RK3399Pro、RK1808、TB-RK1808 AI 计算棒或 RK3399Pro Linux 开发板上运行时的内存使用情况。从 0.9.9 版本开始支持该功能。
6) 模型预编译：通过预编译技术生成的 RKNN 模型可以减少在硬件平台上的加载时间。对于部分模型，还可以减少模型尺寸。但是预编译后的 RKNN 模型只能在带有 NPU 的硬件平台上运行。目前只有 x86_64 Ubuntu 平台支持根据原始模型直接生成 RKNN 模型。RKNN-Toolkit 从 0.9.5 版本开始支持模型预编译功能，并在 1.0.0 版本中对预编译方法进行

了升级，升级后的预编译模型无法与旧驱动兼容。

7) 模型分段：该功能用于多模型同时跑的场景下，可以将单个模型分成多段在 NPU 上执行，借此来调节多个模型占用 NPU 的执行时间，避免因为一个模型占用太多执行时间，而使其他模型得不到及时执行。RKNN-Toolkit 从 1.2.0 版本开始支持该功能。该功能必须在带有 NPU 的硬件上使用，且 NPU 驱动版本要大于 0.9.8。

8) 自定义算子功能：如果模型含有 RKNN-Toolkit 不支持的算子（operator），那么在模型转换阶段就会失败。这时候可以使用自定义算子功能来添加不支持的算子，从而使模型能正常转换和运行。RKNN-Toolkit 从 1.2.0 版本开始支持该功能。自定义算子的使用和开发请参考《Rockchip_Developer_Guide_RKNN_Toolkit_Custom_OP_CN》文档。

9) 量化精度分析功能：该功能将给出模型量化前后每一层推理结果的欧氏距离或余弦距离，以分析量化误差是如何出现的，为提高量化模型的精度提供思路。该功能从 1.3.0 版本开始支持。

10) 可视化功能：该功能以图形界面的形式呈现 RKNN-Toolkit 的各项功能，简化用户操作步骤。用户可以通过填写表单、点击功能按钮的形式完成模型的转换和推理等功能，而不需要再去手动编写脚本。有关可视化功能的具体使用方法请参考《Rockchip_User_Guide_RKNN_Toolkit_Visualization_CN》文档。

11) 模型优化等级功能：RKNN-Toolkit 在模型转换过程中会对模型进行优化，默认的优化选型可能会对模型精度产生一些影响。通过设置优化等级，可以关闭部分或全部优化选型，以分析 RKNN-Toolkit 模型优化选项对精度的影响。有关优化等级的具体使用方法请参考 config 接口中 optimization_level 选项的说明。该功能从 1.3.0 版本开始支持。

2 系统依赖说明

本开发套件支持运行于 Ubuntu、Windows、MacOS、Debian 等操作系统。需要满足以下运行环境要求：

表 1 运行环境

操作系统版本	Ubuntu16.04（x64）及以上 Windows 7（x64）及以上 Mac OS X 10.13.5（x64）及以上 Debian 9.8（x64）及以上
Python 版本	3.5/3.6
Python 库依赖	'numpy == 1.16.3' 'scipy == 1.3.0' 'Pillow == 5.3.0' 'h5py == 2.8.0' 'lmdb == 0.93' 'networkx == 1.11' 'flatbuffers == 1.10', 'protobuf == 3.6.1' 'onnx == 1.4.1' 'onnx-tf == 1.2.1' 'flask == 1.0.2' 'tensorflow == 1.11.0' or 'tensorflow-gpu' 'dill == 0.2.8.2' 'ruamel.yaml == 0.15.81' 'psutils == 5.6.2' 'ply == 3.11' 'requests == 3.11' 'pytorch == 1.2.0' 'mxnet == 1.5.0'

注：Windows 及 MacOS 只提供 Python3.6 的安装包。

3 Ubuntu 平台快速上手

本章节以 Ubuntu 16.04、Python3.5 为例说明如何快速上手使用 RKNN-Toolkit。

3.1 环境准备

- 一台安装有 ubuntu16.04 操作系统的 x86_64 位计算机。
- RK1808 EVB 板。
- 将 RK1808 EVB 板通过 USB 连接到 PC 上，使用 adb devices 命令查看，结果如下：

```
rk@rk:~$ adb devices
List of devices attached
0123456789ABCDEF        device
```

其中标红的为设备 ID。

3.2 安装 RKNN-Toolkit（以 Python3.5 为例）

1. 安装 Python3.5

```
sudo apt-get install python3.5
```

2. 安装 pip3

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

3. 获取 RKNN-Toolkit 安装包，然后执行以下步骤：

   a) 进入 package 目录：

```
cd package/
```

   b) 安装 Python 依赖

```
pip3 install tensorflow==1.11.0
pip3 install mxnet==1.5.0
pip3 install torch==1.2.0 torchvision==0.4.0
```

```
pip3 install opencv-python
pip3 install gluoncv
```

c) 安装 RKNN-Toolkit

```
sudo pip3 install rknn_toolkit-1.3.0-cp35-cp35m-linux_x86_64.whl
```

d) 检查 RKNN-Toolkit 是否安装成功

```
rk@rk:~/rknn-toolkit-v1.3.0/package$ python3
>>> from rknn.api import RKNN
>>>
```

如果导入 RKNN 模块没有失败，说明安装成功。

3.3 运行安装包中附带的示例

3.3.1 在 PC 上仿真运行示例

RKNN-Toolkit 自带了一个 RK1808 的模拟器，可以用来仿真模型在 RK1808 上运行时的行为。

这里以 mobilenet_v1 为例。示例中的 mobilenet_v1 是一个 Tensorflow Lite 模型，用于图片分类，它是在模拟器上运行的。

运行该示例的步骤如下：

1. 进入 examples/tflite/mobilenet_v1 目录

```
rk@rk:~/rknn-toolkit-v1.3.0/package$ cd ../examples/tflite/mobilenet_v1
rk@rk:~/rknn-toolkit-v1.3.0/examples/tflite/mobilenet_v1$
```

2. 执行 test.py 脚本

```
rk@rk:~/rknn-toolkit-v1.3.0/examples/tflite/mobilenet_v1$ python3 test.py
```

3. 脚本执行完后得到如下结果：

```
--> config model
done
--> Loading model
done
--> Building model
```

```
done
--> Export RKNN model
done
--> Init runtime environment
W [RK_nn_softmax_compute:45]Softmax's beta is 0. Set beta to 1
done
--> Running model
mobilenet_v1
-----TOP 5-----
[156]: 0.85107421875
[155]: 0.09173583984375
[205]: 0.01358795166015625
[284]: 0.006465911865234375
[194]: 0.002239227294921875

done
--> Begin evaluate model performance
W [RK_nn_softmax_compute:45]Softmax's beta is 0. Set beta to 1
========================================================================
                            Performance
========================================================================
Layer ID    Name                                        Time(us)
0           tensor.transpose_3                            72
44          convolution.relu.pooling.layer2_2             363
59          convolution.relu.pooling.layer2_2             201
45          convolution.relu.pooling.layer2_2             185
60          convolution.relu.pooling.layer2_2             243
46          convolution.relu.pooling.layer2_2             98
61          convolution.relu.pooling.layer2_2             149
47          convolution.relu.pooling.layer2_2             104
62          convolution.relu.pooling.layer2_2             120
48          convolution.relu.pooling.layer2_2             72
63          convolution.relu.pooling.layer2_2             101
49          convolution.relu.pooling.layer2_2             92
64          convolution.relu.pooling.layer2_2             99
50          convolution.relu.pooling.layer2_2             110
65          convolution.relu.pooling.layer2_2             107
51          convolution.relu.pooling.layer2_2             212
66          convolution.relu.pooling.layer2_2             107
52          convolution.relu.pooling.layer2_2             212
67          convolution.relu.pooling.layer2_2             107
53          convolution.relu.pooling.layer2_2             212
68          convolution.relu.pooling.layer2_2             107
54          convolution.relu.pooling.layer2_2             212
69          convolution.relu.pooling.layer2_2             107
55          convolution.relu.pooling.layer2_2             212
70          convolution.relu.pooling.layer2_2             107
56          convolution.relu.pooling.layer2_2             174
71          convolution.relu.pooling.layer2_2             220
57          convolution.relu.pooling.layer2_2             353
```

```
28          pooling.layer2_1                            36
58          fullyconnected.relu.layer_3                 110
30          softmaxlayer2.layer_1                       90
Total Time(us): 4694
FPS(800MHz): 213.04
============================================================================

done
```

这个例子涉及到的主要操作有：创建 RKNN 对象；模型配置；加载 TensorFlow Lite 模型；构建 RKNN 模型；导出 RKNN 模型；加载图片并推理，得到 TOP5 结果；评估模型性能；释放 RKNN 对象。

examples 目录中的其他示例的执行方式与 mobilenet_v1 相同，这些模型主要用于分类、目标检测。

3.3.2 在 RK1808 上运行示例

这里以 mobilenet_v1 为例。工具包中带的 mobilenet_v1 示例是在 PC 模拟器上运行的，如果要在 RK1808 EVB 板上运行这个示例，可以参考以下步骤：

1. 进入 examples/tflite/mobilenet_v1 目录

```
rk@rk:~/rknn-toolkit-v1.3.0/examples/tflite/mobilenet_v1$
```

2. 修改 test.py 脚本里的初始化环境变量时带的参数

```
rk@rk:~/rknn-toolkit-v1.3.0/examples/tflite/mobilenet_v1$ vim test.py
# 找到脚本里初始化环境变量的方法 init_runtime，如下
ret = rknn.init_runtime()
# 修改该方法的参数
ret = rknn.init_runtime(target='rk1808', device_id='0123456789ABCDEF')
# 保存修改并退出
```

3. 执行 test.py 脚本，得到如下结果：

```
rk@rk:~/rknn-toolkit-v1.3.0/examples/tflite/mobilenet_v1$ python test.py
--> config model
done
--> Loading model
done
--> Building model
```

```
done
--> Export RKNN model
done
--> Init runtime environment
done
--> Running model
mobilenet_v1
-----TOP 5-----
[156]: 0.85107421875
[155]: 0.09173583984375
[205]: 0.01358795166015625
[284]: 0.006465911865234375
[194]: 0.002239227294921875

done
--> Begin evaluate model performance
==========================================================================
                            Performance
==========================================================================
Total Time(us): 5805
FPS: 172.27
==========================================================================

done
```

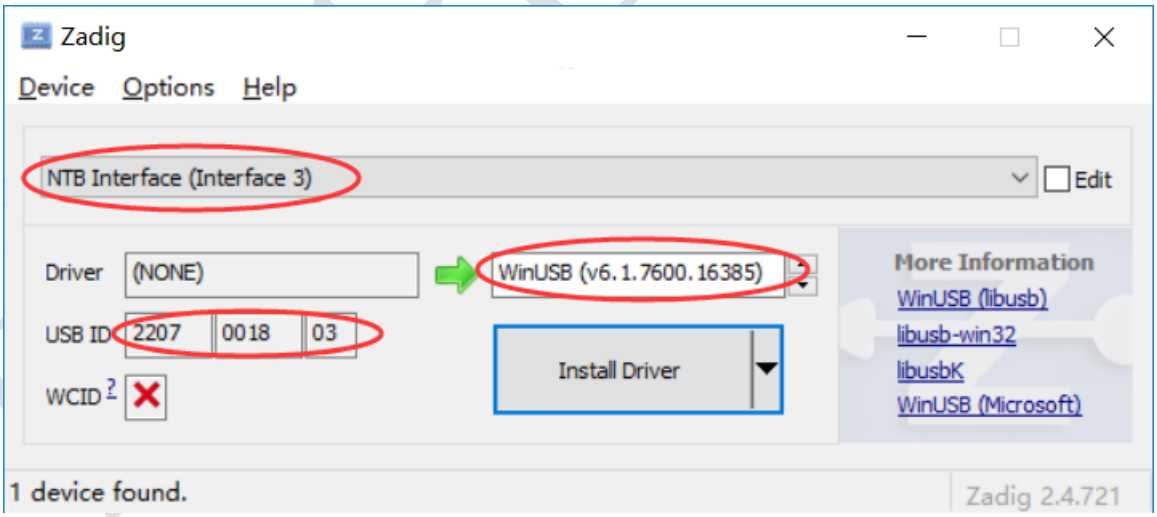
4 Windows 平台（Python3.6）快速上手指南

本章节说明如何在 Windows 系统、Python3.6 环境中使用 RKNN-Toolkit。

4.1 环境准备

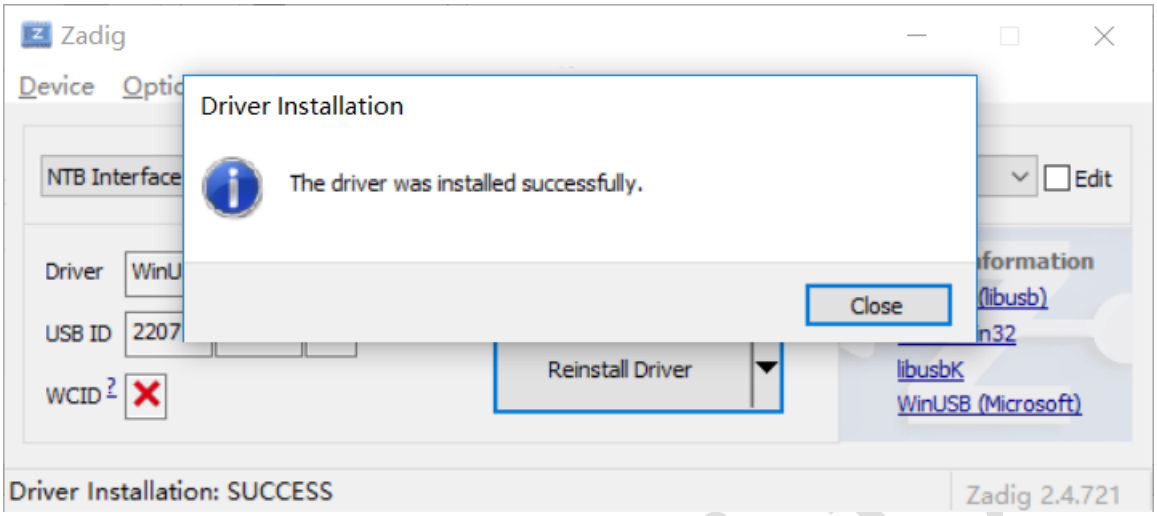
- 一台安装有 Windows 7（或 Windows10）操作系统的 PC。
- 一个 TB-RK1808 AI 计算棒（Windows 平台目前只支持计算棒）。
- 将 TB-RK1808 AI 计算棒通过 USB 连接到 PC 上。第一次使用计算棒时需要安装相应的驱动，安装方式如下：
  - 进入 SDK 包 platform-tools/drivers_installer/windows-x86_64 目录，以管理员身份运行 zadig-2.4.exe 程序安装计算棒的驱动，如下图所示：

    1. 确认待安装的设备及需要安装的驱动



注：待安装的设备其 USB ID 应该是 **2207:0018**；安装的驱动选择默认的 WinUSB 确认完后点 Install Driver 开始安装驱动。

    2. 安装成功后会出现如下界面：



■ 安装完后如果windows设备管理器的中的TB-RK1808 AI计算棒没有感叹号，且如下所示，说明安装成功：

注：安装完驱动后需要重启计算机。

4.2 安装 RKNN-Toolkit

安装 RKNN-Toolkit 前需要确保系统里已经安装有 Python3.6。这可以通过 cmd 里执行 python –version 确定，如下说明系统已经安装有 Python3.6：

```
C:\Users\momen.raul>python --version
Python 3.6.8
```

获取 RKNN-Toolkit SDK 包，然后执行以下步骤：

1. 在 sdk 根目录以管理员权限执行 cmd，然后进入 package 目录：

```
D:\workspace\rknn-toolkit-v1.3.0>cd packages
```

2. 安装依赖（tensorflow 和 opencv-python）：

```
pip install tensorflow==1.11.0
pip install torch==1.2.0+cpu torchvision==0.4.0+cpu -f
https://download.pytorch.org/whl/torch_stable.html --user
pip install mxnet==1.5.0
pip install opencv-python
```

```
pip install gluoncv
```

注：opencv-python 和 gluoncv 在运行 example 中的例子时会用到。

3. 手动安装 lmdb，该 wheel 包放在 packages\required-packages-for-win-python36 目录下：

```
D:\workspace\rknn-toolkit-v1.3.0\packages\required-packages-for-win-python36>pip install lmdb-0.95-cp36-cp36m-win_amd64.whl
```

4. 安装 RKNN-Toolkit

```
pip install rknn_toolkit-1.3.0-cp36-cp36m-win_amd64.whl
```

5. 检查 RKNN-Toolkit 是否安装成功

```
D:\workspace\rknn-toolkit-v1.3.0\packages>python
Python 3.6.8 (tags/v3.6.8:3c6b436a57, Dec 24 2018, 00:16:47) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from rknn.api import RKNN
>>>
```

如果导入 RKNN 模块没有失败，说明安装成功。

4.3 运行安装包中附带的示例

这里以 mobilenet_v1 为例。示例中的 mobilenet_v1 是一个 Tensorflow Lite 模型，用于图片分类。

运行该示例的步骤如下：

1. 进入 examples/tflite/mobilenet_v1 目录

```
D:\workspace\rknn-toolkit-v1.3.0\packages>cd ..\

D:\workspace\rknn-toolkit-v1.3.0>cd examples\tflite\mobilenet_v1
```

2. 修改脚本 test.py 脚本，找到调用 init_runtime 接口的地方，添加参数 target='rk1808'：

```
#修改前：
ret = rknn.init_runtime()
#修改后：
```

```
    ret = rknn.init_runtime(target='rk1808')
```

3. 执行 test.py 的脚本：

```
    D:\workspace\rknn-toolkit-v1.3.0\examples\tflite\mobilenet_v1>python
test.py
```

4. 脚本执行完后得到概率最高的前 5 类结果及模型运行的参考性能：

```
    --> config model
    done
    --> Loading model
    done
    --> Building model
    done
    --> Export RKNN model
    done
    --> Init runtime environment
    done
    --> Running model
    mobilenet_v1
    -----TOP 5-----
    [156]: 0.8828125
    [155]: 0.06768798828125
    [188 205]: 0.0086669921875
    [188 205]: 0.0086669921875
    [263]: 0.006366729736328125

    done
    --> Begin evaluate model performance
    ========================================================
                                Performance
    ========================================================
    Total Time(us): 6032
    FPS: 165.78
    ========================================================

    done
```

这个例子涉及到的主要操作有：创建 RKNN 对象；模型配置；加载 TensorFlow Lite 模型；构建 RKNN 模型；导出 RKNN 模型；加载图片并推理，得到 TOP5 结果；评估模型性能；释放 RKNN 对象。

examples 目录中的其他示例的执行方式与 mobilenet_v1 相同，这些模型主要用于图像分类、目标检测。

注：

1. Windows 平台并不提供 NPU 模拟器功能，所以在 Windows 平台上必须接 TB-RK1808 计算棒进行使用。
2. 更多有关 TB-RK1808 AI 计算棒的资料，请参考链接：

   http://t.rock-chips.com/wiki.php?mod=view&pid=28

5 Mac OS X 平台（Python3.6）快速上手指南

本章节说明如何在 Mac OS X 系统、Python3.6 环境中使用 RKNN-Toolkit。

5.1 环境准备

- 一台安装有 MacOS High Sierra（或更高版本）操作系统的 Mac PC。
- 一个 TB-RK1808 AI 计算棒。
- 将 TB-RK1808 AI 计算棒通过 USB 连接到 PC 上。在 pc 上进入 SDK 包 platform-tools/ntp/mac-osx-x86_64 目录，运行 npu_transfer_proxy 程序查看是否存在可用的 RK1808 计算棒，命令如下：

```
macmini:ntp rk$ ./npu_transfer_proxy devices
List of ntb devices attached
TS018080000000013       2bed0cc1        USB_DEVICE
```

上图标红的这一行即为我们插入的 TB-RK1808 AI 计算棒。设备 ID 为“**TS018080000000013**”。

5.2 安装 RKNN-Toolkit

获取 RKNN-Toolkit SDK 包，然后执行以下步骤：

1. 进入 rknn-toolkit-v1.3.0/packages 目录：

```
cd packages/
```

2. 安装 tensorflow、opencv-python 依赖

```
pip3 install tensorflow==1.11.0
pip3 install mxnet==1.5.0
pip3 install torch==1.2.0 torchvision==0.4.0
pip3 install opencv-python
pip3 install gluoncv
```

注：opencv-python 和 gluoncv 在运行 example 中的例子时会用到。

3. 安装 RKNN-Toolkit

```
pip3 install rknn_toolkit-1.3.0-cp36-cp36m-macosx_10_9_x86_64.whl
```

4. 检查 RKNN-Toolkit 是否安装成功

```
(rknn-venv)macmini:rknn-toolkit-v1.3.0 rk$ python3
>>> from rknn.api import RKNN
>>>
```

如果导入 RKNN 模块没有失败，说明安装成功。

5.3 运行安装包中附带的示例

这里以 mobilenet_v1 为例。示例中的 mobilenet_v1 是一个 Tensorflow Lite 模型，用于图片分类。

运行该示例的步骤如下：

1. 进入 examples/tflite/mobilenet_v1 目录

```
(rknn-venv)macmini:rknn-toolkit-v1.3.0 rk$ cd
examples/tflite/mobilenet_v1
```

2. 修改脚本 test.py 脚本，找到调用 init_runtime 接口的地方，添加参数 target='rk1808'：

```
#修改前：
ret = rknn.init_runtime()
#修改后：
ret = rknn.init_runtime(target='rk1808')
```

3. 执行 test.py 脚本

```
(rknn-venv)macmini:mobilenet_v1 rk$ python3 test.py
```

4. 脚本执行完后得到 Top5 结果：

```
--> config model
done
--> Loading model
done
```

```
--> Building model
done
--> Export RKNN model
done
--> Init runtime environment
done
--> Running model
mobilenet_v1
-----TOP 5-----
[156]: 0.85107421875
[155]: 0.09173583984375
[205]: 0.01358795166015625
[284]: 0.006465911865234375
[194]: 0.002239227294921875

done
--> Begin evaluate model performance
============================================
                Performance
============================================
Total Time(us): 6046
FPS: 165.40
============================================

done
```

这个例子涉及到的主要操作有：创建 RKNN 对象；模型配置；加载 TensorFlow Lite 模型；构建 RKNN 模型；导出 RKNN 模型；加载图片并推理，得到 TOP5 结果；评估模型性能；释放 RKNN 对象。

examples 目录中的其他示例的执行方式与 mobilenet_v1 相同，这些模型主要用于图像分类、目标检测。

注：

1. Mac OS X 平台并不提供 NPU 模拟器功能，所以在 Mac OS X 平台上必须接 TB-RK1808 计算棒进行使用。
2. 更多有关 TB-RK1808 AI 计算棒的资料，请参考链接：

   http://t.rock-chips.com/wiki.php?mod=view&pid=28

6 ARM64 平台（Python3.5）快速上手指南

本章节说明如何在 ARM64 平台（Debian 9.8 系统）、Python3.5 环境中使用 RKNN-Toolkit。

6.1 环境准备

- 一台安装有 Debian 9.8 操作系统的 RK3399Pro，并且确保 root 分区剩余空间大于 5GB。
- 确保 NPU 驱动版本大于 0.9.6。
- 如果/usr/bin 目录下没有 npu_transfer_proxy 或 npu_transfer_proxy.proxy 程序，则将 rknn-toolkit-v1.3.0\platform-tools\ntp\linux_aarch64 目录下的 npu_transfer_proxy 拷贝到/usr/bin/目录下，并进到该目录执行以下命令（每次重启后都要启动该程序，可以将它加到开机脚本中）：

```
sudo ./npu_transfer_proxy &
```

6.2 安装 RKNN-Toolkit

1. 执行以下命令更新系统包，这些包在后面安装 Python 依赖包时会用到。

```
sudo apt-get update
sudo apt-get install cmake gcc g++ libprotobuf-dev protobuf-compiler
sudo apt-get install liblapack-dev libjpeg-dev zlib1g-dev
sudo apt-get install python3-dev python3-pip python3-scipy
```

2. 执行以下命令更新 pip

```
pip3 install --upgrade pip
```

3. 安装 Python 打包工具

```
pip3 install wheel setuptools
```

4. 安装依赖包 h5py/gluoncv

```
sudo apt-get build-dep python3-h5py && \
pip3 install h5py
pip3 install gluoncv
```

5. 安装 TensorFlow，相应的 whl 包在

rknn-toolkit-v1.3.0\packages\required-packages-for-arm64-debian9-python35 目录下：

```
pip3 install tensorflow-1.11.0-cp35-none-linux_aarch64.whl --user
```

注：由于 TensorFlow 依赖的一些库在 ARM64 平台上需要下载后源码后进行编译安装，所以这一步会耗费较长时间。

6. 安装 opencv-python，相应的 whl 包在

rknn-toolkit-v1.3.0\packages\required-packages-for-arm64-debian9-python35 目录下：

```
pip3 install \
opencv_python_headless-4.0.1.23-cp35-cp35m-linux_aarch64.whl
```

7. 安装 RKNN-Toolkit，相应的 whl 包在 rknn-toolkit-v1.3.0\packages\目录下：

```
pip3 install rknn_toolkit-1.3.0-cp35-cp35m-linux_aarch64.whl --user
```

注：由于 RKNN-Toolkit 依赖的一些库在 ARM64 平台上需要下载源码后编译安装，所以这一步会耗费较长时间。

6.3 运行安装包中附带的示例

这里以 mobilenet_v1 为例。示例中的 mobilenet_v1 是一个 Tensorflow Lite 模型，用于图片分类。

运行该示例的步骤如下：

1. 进入 examples/tflite/mobilenet_v1 目录

```
linaro@linaro-alip:~/rknn-toolkit-v1.3.0/ $ cd examples/tflite/mobilenet_v1
```

2. 执行 test.py 脚本

```
```

```
    linaro@linaro-alip:
~/rknn-toolkit-v1.3.0/examples/tflite/mobilenet_v1$ python3 test.py
```

3. 脚本执行完后得到如下结果：

```
    --> config model
    done
    --> Loading model
    done
    --> Building model
    done
    --> Export RKNN model
    done
    --> Init runtime environment
    done
    --> Running model
    mobilenet_v1
    -----TOP 5-----
    [156]: 0.85107421875
    [155]: 0.09173583984375
    [205]: 0.01358795166015625
    [284]: 0.006465911865234375
    [194]: 0.002239227294921875

    done
    --> Begin evaluate model performance
    ========================================================
                                Performance
    ========================================================
    Total Time(us): 5761
    FPS: 173.58
    ========================================================

    done
```

这个例子涉及到的主要操作有：创建 RKNN 对象；模型配置；加载 TensorFlow Lite 模型；构建 RKNN 模型；导出 RKNN 模型；加载图片并推理，得到 TOP5 结果；评估模型性能；释放 RKNN 对象。

examples 目录中的其他示例的执行方式与 mobilenet_v1 相同，这些模型主要用于图像分类、目标检测。

注：

1. ARM64 平台不支持模拟器功能，所以这些示例都是跑在 RK3399Pro 自带的 NPU 上。
2. ARM64 平台目前只支持 RK3399 和 RK3399Pro，如果是 RK3399，需要外接一个 TB-RK1808 AI 计算棒。
3. 更多有关 TB-RK1808 AI 计算棒的更多资料，请参考链接：

   http://t.rock-chips.com/wiki.php?mod=view&pid=28

7 参考文档

有关 RKNN-Toolkit 更详细的用法和接口说明，请参考《Rockchip_User_Guide_RKNN_Toolkit_v1.3.0_CN.pdf》手册。