



K210

模型转换工具使用指南



KENDRYTE

勘智

嘉楠科技 版权©2020
KENDRYTE.COM



关于本手册

本文介绍了[Kendryte K210](#)的模型转换工具[nncase](#)的使用方法。

发布说明

日期	版本	发布说明
2020.1.20	V0.1	首次发布

免责声明

本文中的信息，包括参考的 URL 地址，如有变更，恕不另行通知。文档“按现状”提供，不负任何担保责任，包括对适销性、适用于特定用途或非侵权性的任何担保和任何提案、规格或样品在他处提到的任何担保。

本文档不负任何责任，包括使用本文档内信息产生的侵犯任何专利权行为的责任。本文档在此未以禁止反言或其他方式授予任何知识产权使用许可，不管是明示许可还是暗示许可。

文中提到的所有商标名称、商标和注册商标均属其各自所有者的财产，特此声明。

版权公告

版权归 © 2020 嘉楠科技所有。保留所有权利。

1. 概述

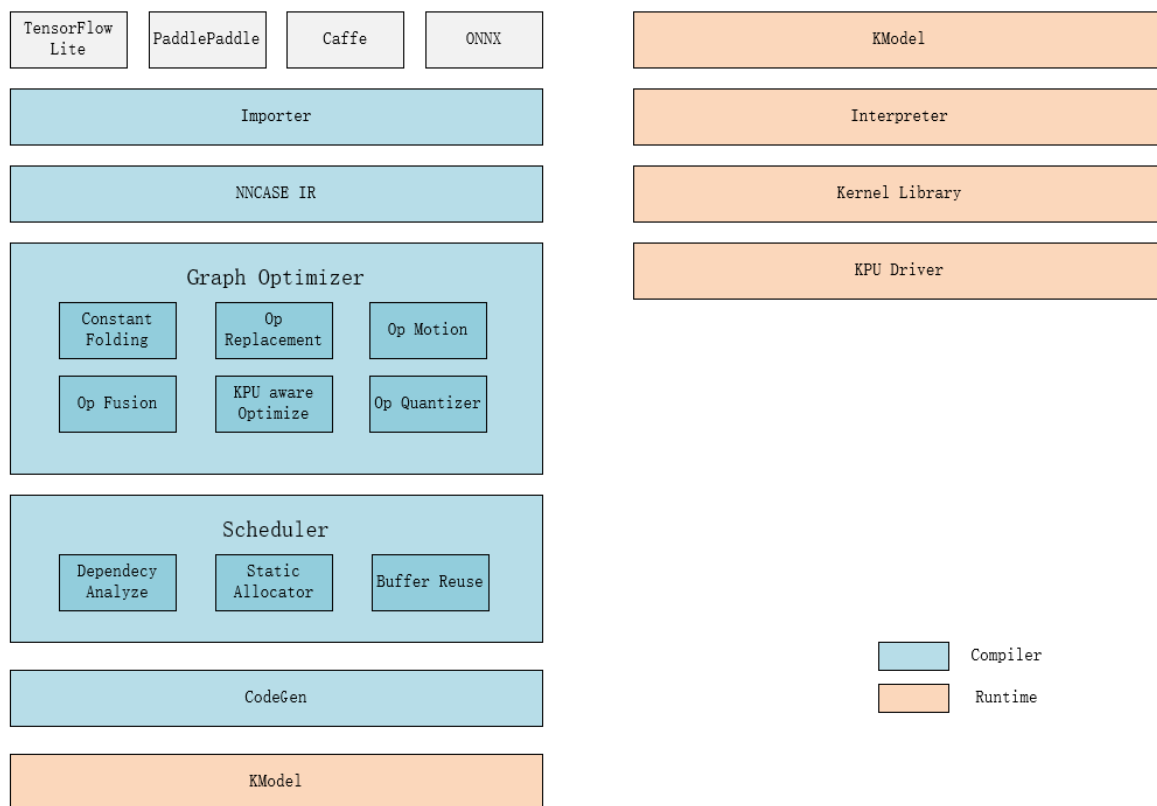
[Kendryte K210](#)是嘉楠科技自主研发设计的边缘侧AI SoC，内置自主知识产权的KPU单元，对卷积神经网络（CNN）的推理进行硬件加速。

不同于云端AI芯片，边缘侧AI芯片通常只支持神经网络的定点运算，并在支持的操作类型等方面进行定制和优化，以取得功耗和性能的平衡，K210也是如此。使用通用的神经网络框架，如TensorFlow、Caffe，训练的模型要在K210上运行，需要将浮点数据量化为定点数据，并根据K210的特性将其中的算子或操作进行转换和适配，模型转换工具[nncase](#)正是为了这个目的而设计的。

[nncase](#)将基于通用神经网络框架训练出的模型转换成K210能支持的格式，调用[K210 SDK](#)中的API即可完成模型在芯片上的推理过程。

2. 功能特性

2.1 架构



注：目前仅支持 TensorFlow Lite 和 Caffe 的模型，而且对 TensorFlow Lite 算子的支持更全面，如果使用其他训练框架的模型，建议先借助工具将其转为 TensorFlow Lite 的格式。

2.2 特性

- 支持多输入输出网络，支持多分支结构
- 静态内存分配，不需要堆内存
- 算子合并和优化
- 支持float和量化uint8推理
- 支持训练后量化，使用浮点模型和量化校准集
- 平坦模型，支持零拷贝加载

2.3 支持的算子

2.3.1 TensorFlow Lite

算子	是否支持	算子	是否支持	算子	是否支持
Add	☑	Neg	☑	TransposeConv	☑
ArgMax	✗	NotEqual	✗	LogicalOr	✗
ArgMin	✗	Pack	✗	OneHot	✗
AveragePool2D	☑	Pad	☑	LogicalAnd	✗
BatchToSpaceND	✗	Pow	✗	LogicalNot	✗
Cast	✗	PRelu	✗	UnPack	✗
Concatenation	☑	ReduceMax	☑	ReduceMin	☑
Conv2D	☑	ReduceProd	✗	FloorDiv	✗
DepthwiseConv2D	☑	Reshape	☑	ReduceAny	✗
Div	☑	ResizeBilinear	☑	ZerosLike	✗
Equal	✗	Rsqrt	☑	Fill	✗
Exp	☑	Select	✗	FloorMod	✗
ExpandDims	✗	Shape	✗	Range	✗
Floor	☑	Sin	☑	ResizeNearestNeighbor	☑
FullyConnected	☑	Slice	☑	LeakyRelu	☑
Gather	✗	Softmax	☑	MirrorPad	✗
Greater	✗	SpaceToDepth	✗	Abs	☑
GreaterEqual	✗	SpaceToBatchND	✗	SplitV	✗
MaxPool2D	☑	SparseToDense	✗	Unique	✗
Mean	☑	Split	✗	Ceil	☑
Mul	☑	Sqrt	☑	Reverse	✗
L2Normalization	☑	Square	☑	AddN	✗
L2Pool2D	✗	Squeeze	✗	GatherND	✗
LessEqual	✗	StridedSlice	☑	Cos	☑
Log	☑	Sub	☑	Where	✗
Logistic	☑	Sum	☑	Rank	✗
LogSoftmax	✗	Tile	✗	Elu	✗
Maximum	☑	TopK	✗	ReverseSequence	✗
Minimum	☑	Transpose	☑		

2.3.2 Caffe

算子	是否支持
Concat	✓
Convolution	✓
Eltwise	✓
Permute	✓
ReLU	✓
Reshape	✓
Slice	✓
Softmax	✓
Split	✓

3. 使用方法

[nncase](#)是一款命令行工具，需要在命令行终端中使用，支持Ubuntu、Windows和Mac OS三个平台。

注：目前[nncase](#)分为[稳定版](#)和测试版，相比[稳定版](#)，测试版会包括一些新特性，可以从[nncase仓库](#)的README的Azure DevOps入口进入下载。



3.1 命令行介绍

3.1.1 命令格式

DESCRIPTION

NNCASE model compiler and inference tool.

SYNOPSIS

```
ncc compile <input file> <output file> -i <input format> [-o <output
format>] [-t <target>] [--dataset <dataset path>] [--dataset-format
<dataset format>] [--inference-type <inference type>] [--input-mean
<input mean>] [--input-std <input std>] [--dump-ir] [--input-type <input
type>] [--max-allocator-solve-secs <max allocator solve secs>]
[--calibrate-method <calibrate method>] [-v]
```

```
ncc infer <input file> <output path> --dataset <dataset path>
[--dataset-format <dataset format>] [--input-mean <input mean>]
[--input-std <input std>] [-v]
```

OPTIONS

compile

<input file>	input file
<output file>	output file
-i, --input-format	input file format: e.g. tflite, caffe
-o, --output-format	output file format: e.g. kmodel, default is kmodel
-t, --target	target arch: e.g. cpu, k210, default is k210
--dataset	calibration dataset, used in post quantization
--dataset-format	dataset format: e.g. image, raw default is image
--inference-type	inference type: e.g. float, uint8 default is uint8
--input-mean	input mean, default is 0.000000
--input-std	input std, default is 1.000000
--dump-ir	dump nncase ir to .dot files
--input-type	input type: e.g. default, float, uint8, default means equal to inference type

--max-allocator-solve-secs

max optimal layout solve time in secs used by allocators, 0 means don't use solver, default is 60

--calibrate-method calibrate method: e.g. no_clip, l2, default is

```
no_clip

infer
  <input file>          input kmodel
  <output path>         inference result output directory
  --dataset             input dataset to inference
  --dataset-format      dataset format: e.g. image, raw default is image
  --input-mean          input mean, default is 0.000000
  --input-std           input std, default is 1.000000

-v, --version          show version
```

3.1.2 参数描述

ncc 是 [nncase](#) 的命令行工具名称，它包括两个命令： `compile` 和 `infer`。

1. `compile` 命令用于将训练好的模型（`.tflite`，`.caffemodel`）转换为K210支持的格式（`.kmodel`）。

 - o `<input file>`：输入模型的路径。
 - o `<output file>`：输出模型的路径。
 - o `-i, --input-format`：输入模型的格式，目前支持 `.tflite` 和 `.caffemodel` 格式。
 - o `-o, --output-format`：输出模型的格式，目前仅支持 `.kmodel` 格式。
 - o `-t, --target`：指定模型在哪种目标设备上运行。k210 指 [Kendryte K210](#) SoC平台，cpu 指几乎所有平台都支持的通用目标。如果需要模型在K210上运行或者在PC上模拟K210运行，则需要指定 `-t k210`。
 - o `--inference-type`：指定推理类型。如果为 `uint8`，则会利用K210的KPU能力进行加速，得到较快的执行速度，此时需要 `--dataset` 指定量化校准集；如果为 `float`，会达到更高的精度，但是会占用更多内存，且无法利用K210的KPU能力。
 - o `--dataset`：指定用于模型量化的量化校准集。需要从训练集中选择尽量覆盖各个类别的几百到上千个数据放到该目录中。只有指定 `--inference-type uint8` 时才需要指定该参数。
 - o `--dataset-format`：指定量化校准集的格式。默认是 `image`，[nncase](#) 会使用 `opencv` 读取 `--dataset` 指定目录中的图片，并自动缩放到输入模型的尺寸。如果有3个通道，[nncase](#) 会将图片转换为值域是[0, 1]、布局是 `NCHW` 的张量；如果有1个通道，[nncase](#) 会将图片灰度化。如果数据集不是图片（例如音频或者矩阵），则需要指定为 `raw`，这种场景下需要事先把数据集转换为 `float` 张量的二进制文件。只有指定 `--inference-type uint8` 时才需要指定该参数。
 - o `--input-std`，`--input-mean`：指定量化校准集的预处理方法。如上所述[nncase](#) 会将图片转换为值域是[0, 1]、布局是 `NCHW` 的张量，之后[nncase](#) 会使用 $y = (x - \text{mean}) / \text{std}$ 公式对数据进行归一化。

输入值域	--input-std	--input-mean
[0, 1]（默认）	1	0
[-1, 1]	0.5	0.5

- o `--calibrate-method`：指定量化校准方法，用来选择最优的激活函数值域。默认值是 `no_clip`，[nncase](#) 会使用整个激活函数值域；如果需要更好的量化结果，可以指定 `12`，但是会需要更长时间来寻找最优值域。

- `--input-type`: 指定推理时输入的数据类型, 默认和 `--inference-type` 相同。如果指定为 `uint8`, 推理时需要输入为RGB888 `uint8`张量; 如果指定为 `float`, 推理时需要输入为RGB `float`张量。
 - `--max-allocator-solve-secs`: 指定 `nncase` 做最优分配时的最大搜索时间。如果搜索超过了这个时间, `nncase` 会退而使用first fit算法。默认是60秒, 如果要禁用搜索则设置为0。
 - `--dump-ir`: 调试选项。打开时 `nncase` 会在工作目录产生一些 `.dot` 文件, 可以使用 [Graphviz Online](#) 来查看这些文件中的结构图。
2. `infer` 命令用于在PC上模拟 `.kmodel` 的运行, 通常用来调试。 `nncase` 会将模型的输出张量按照NCHW布局保存到 `.bin` 文件中。
- `<input file>`: `.kmodel` 的路径。
 - `<output path>`: 输出结果保存目录, 即 `.bin` 文件所有目录。
 - `--dataset`: 指定测试数据集路径。
 - `--dataset-format`、`--input-std` 和 `--input-mean` 含义与 `compile` 命令中相同。

3.2 转换示例

这里以 [mobilenetv1_1.0.pb](#) 为例, 说明Ubuntu下 `nncase` 的使用方法。假设 `nncase` 目录层次如下:

```
nncase/
  ncc                                # nncase转换工具
  model/                             # 模型文件目录
    mobilenetv1_1.0.pb
  dataset/                           # 量化校准集目录
  testset/                           # 测试数据集目录
  result/                            # 模型输出目录
```

1. `.pb` 转为 `.tflite`

```
toco --graph_def_file=./model/mobilenetv1_1.0.pb --
output_file=./model/mobilenetv1_1.0.tflite --output_format=TFLITE --
input_shape=1,224,224,3 --input_arrays=inputs --
output_arrays=MobileNetV1/Bottleneck2/BatchNorm/Reshape_1 --inference_type=FLOAT
```

注: TensorFlow 的 `.pb` 模型可以由其自带的 `toco` 工具转换成 `.tflite`, 其他训练框架的模型需要其他转换工具转成 `.tflite`。

2. `.tflite` 转为 `.kmodel`

```
./ncc compile ./model/mobilenetv1_1.0.tflite ./model/mobilenetv1_1.0.kmodel -i
tflite -o kmodel --dataset ./dataset/
```

注: 此处其他参数根据模型参数而使用了默认值。

3. 模拟运行 `.kmodel`

```
./ncc infer ./model/mobilenetv1_1.0.kmodel ./result --dataset ./testset
```

注: 此处其他参数根据模型参数而使用了默认值。

执行完上述命令后, 对于测试数据集中的每张图片, 都会在 `./result` 目录下生成一个结果 `.bin` 文件。这些 `.bin` 文件可以通过如下python脚本转换成浮点数:

```
import numpy as np

results = np.fromfile(bin_file, dtype='float32')
np.set_printoptions(threshold=np.inf)
print(results)
```

3.3 上板测试

将模型转换成 `.kmodel` 之后，就可以使用[K210 SDK](#)中的 `kpu_load_kmodel`、`kpu_run_kmodel` 和 `kpu_get_output` 三个API来编写上板代码了。完整代码参考[K210code](#)。

将[K210code](#)拷贝到[K210 SDK](#)的 `src` 目录下，根据《K210命令行开发环境搭建指南》完成编译下载后，就可以通过串口查看模型的输出结果了。

4. 注意事项

K210 KPU硬件支持的神经网络算子类型及要求（以TensorFlow为例）如下，模型设计时如果能尽可能满足这些约束，则可以达到尽可能优化的性能。

1. conv2d

kernel size	stride	要求的 TensorFlow padding方式
1*1	1	same或valid
3*3	1	same
3*3	2	先手动padding一圈，然后valid

2. depthwise conv2d

kernel size	stride	要求的 TensorFlow padding方式
3*3	1	same
3*3	2	先手动padding一圈，然后valid

3. pooling

kernel size	stride	要求的 TensorFlow padding方式
2*2	1	same
2*2	2	如果feature map size是kernel size的整数倍，same或valid；否则valid
4*4	4	如果feature map size是kernel size的整数倍，same或valid；否则valid