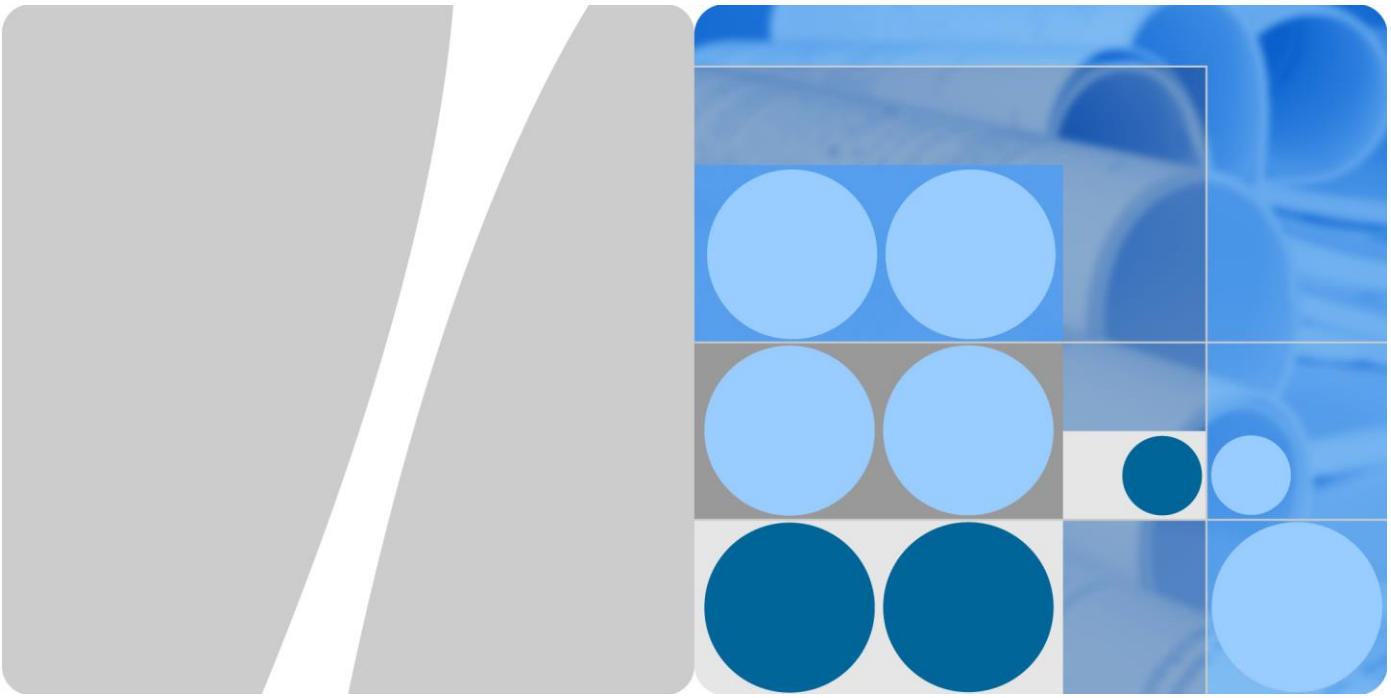

HiAI DDK V320

OMG 工具使用说明

文档版本 02

发布日期 2019-12-31

华为技术有限公司

华为 HiAI 申请方式

发送申请邮件到邮箱：developer@huawei.com

邮件名称：HUAWEI HiAI+公司名称+产品名称

邮件正文：合作公司+联系人+联系方式+联系邮箱

我们将在收到邮件的 5 个工作日内邮件给您反馈结果，请您注意查收。

官网地址 https://developer.huawei.com/consumer/cn/

前 言

概述

本文提供对 OMG 模型转换工具的使用说明。

修改记录

修改记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

日期	修订版本	修改描述
2019-12-31	02	新增 V320 版本内容
2019-09-04	01	新增 V310 版本内容

目 录

表格目录

1 概述

使用 HiAI DDK 时，可以预先使用 OMG 工具将 Caffe 和 TensorFlow 模型转换为 OM 离线模型，移动端 AI 程序直接读取离线模型进行推理。OMG 工具位于 DDK 包中的 tools 目录下，可运行在 64 位 Linux 平台上。

2 模型转换示例

2.1 Caffe 模型转换

```
./omg --model xxx.prototxt --weight yyy.caffemodel --framework 0 --
output ./modelname
```

转换示例:

```
./omg --model deploy.prototxt --weight squeezenet_v1.1.caffemodel --
framework 0 --output ./squeezenet
```

当看到 OMG generate offline model success.时，则说明转换成功,会在当前目录下生成 squeezenet.om。

2.2 TensorFlow 模型转换

```
./omg --model xxx.pb --framework 3 --output ./modelname --input_shape
"xxx:n,h,w,c" --out_nodes "node_name1:0"
```

转换示例:

```
./omg --model mobilenet_v2_1.0_224_frozen.pb --framework 3 --
output ./mobilenet_v2 --input_shape "input:1,224,224,3" --out_nodes
"MobilenetV2/Predictions/Reshape_1:0"
```

当看到 OMG generate offline model success 时，则说明转换成功，会在当前目录下生成 mobilenet_v2.om。

2.3 量化模型转换（以 Caffe 模型为例）

目前大部分模型都是使用 32bit float 类型进行计算的，使用量化既可以减少模型的体积，也可以加快模型推理速度。

量化模型转换依赖轻量化工具，利用轻量化工具生成模型及轻量化配置，通过compress_conf 参数传递给 OMG 并生成量化模型，更多说明请参考《华为HiAI_DDK_V320_轻量化工具使用说明书》。

转换命令：

```
./omg --model xxx.prototxt --weight xxx.caffemodel --framework 0 --
output ./modelname  --compress_conf=param.json
```

转换示例：

```
./omg --model deploy.prototxt --weight squeezenet_v1.1.caffemodel --
framework 0 --output ./squeezenet --compress_conf=param.json
```

当看到 OMG generate offline model success 时，说明模型量化成功，会在当前目录下生成量化模型 suqeezenet.om。

2.4 AIPP 模型转换（以 Caffe 模型为例）

如果模型推理需要对图像或其他输入数据进行变换（如图像尺寸变换、色域转换、减均值/乘系数等），可使用 AIPP 模型转换功能。转换后的模型增加算子替换此类操作，并提升了效率。

说明

本转换功能为 HiAI DDK V310 新增功能。

转换命令：

```
./omg --model xxx.prototxt --weight xxx.caffemodel --framework 0 --
insert_op_conf aipp_conf_static.cfg --output ./modelname
```

转换示例:

```
./omg --model deploy.prototxt --weight squeezenet_v1.1.caffemodel --
framework 0 --insert_op_conf aipp_conf_static.cfg --output ./squeezenet
```

当看到 OMG generate offline model success 的时候，说明 AIPP 模型转换成功，会在当前目录下生成 AIPP squeezenet.om 模型。

说明

aipp_conf_static.cfg 是 AIPP 的配置文件，位置存放在 tools/tools_omg/sample 文件夹目录中，具体说明参考“3.2.4 AIPP 配置文件说明”。

3 OMG 参数说明

说明

- 路径部分：支持大小写字母、数字，下划线。
- 文件名部分：支持大小写字母、数字，下划线和点(.)

3.1 整体参数

参数名称	参数描述	是否必选	默认值
--hiai_version	指定使用 OMG 的版本，当前支持：v300 | v310 | v320 | IR	否	IR
--model	原始框架模型文件路径。	是	不涉及
--weight	权值文件路径。当原始模型是 Caffe 时需要指定。当原始模型是 TensorFlow 时不需要指定.	否	不涉及
--framework	原始框架类型 0：Caffe 3：TensorFlow 说明 • 当 mode 为 1 时，该参数可选，可以指定 Caffe 或 TensorFlow，不指定时默认为离线模型转 json。 • 当 mode 为 0 或 3 时，该参数必选，可以指定 Caffe 或 TensorFlow。	是	不涉及
--output	存放转换后的离线模型文件的路径（包含文件名），例如“out/caffe_resnet18”。转换后的模型文件，会自动以“.om”的后缀结尾。	是	不涉及
--stream_num	模型使用的 stream 数量，当前仅支持 1。	否	1
--input_shape	输入数据的 shape。 例如：“input_name1: n1, c1,h1, w1; input_name2: n2, c2, h2,w2”。input_name 必须是转换前的网络模型中的节点名称。	否	不涉及

参数名称	参数描述	是否必选	默认值
	注：当原始模型是 TensorFlow 时，此参数必须指定。		
--h/help	显示帮助信息。	否	不涉及
--compress_conf	轻量化配置文件路径。该参数需与轻量化工具搭配使用，由轻量化工具自动生成。具体参见《华为 HiAI_DDK_V320_轻量化工具使用说明书》。	否	不涉及
--insert_op_conf	AIPP 配置文件路径。详情参考 3.2 AIPP 参数。 **说明** 本参数为 HiAI DDK V310 新增参数。	否	不涉及
--op_name_map	算子映射配置文件路径。包含 DetectionOutput 网络时需要指定。 例如：不同的网络中 DetectionOutput 算子的功能不同，指定 DetectionOutput 到 FSRDetectionOutput 或者 SSDDetectionOutput 的映射。 **说明** 算子映射配置文件的内容示例如下： DetectionOutput：SSDDetectionOutput。	否	不涉及
--om	模型文件路径。当 mode 为 1 时必填。	否	不涉及
--json	模型文件转换为 json 格式文件的路径。	否	不涉及
--mode	运行模式 • 0：生成 DaVinci 模型 • 1：模型转 json 注：json 是可查看模型结构的文本格式。 • 3：仅做预检，检查模型文件的内容是否合法。 注：预检即检查算子是否支持，会生成一个检查结果的报告。	否	0
--target	当前仅支持设置为“Lite”。	否	Lite
--out_nodes	指定输出节点。 例如：“node_name1:0;node_name1:1;node_name2:0”。 node_name 必须是模型转换前的网络模型中的节点名称。 同一个节点的输出从 0 开始，如果该节点有多个输出的，依次往后累加。 注：当原始模型是 TensorFlow 时，此参数必须指定。	否	不涉及
--input_format	输入数据格式：NCHW 和 NHWC。 注：1.当原始框架是 Caffe 时，此参数不生效。 2.当原始框架是 TensorFlow 时，绝大多数场景不需要指定，如果转换时提示需要指定，根据实际情况指定。	否	不涉及
--check_report	预检结果保存文件路径。若不指定该路径，在模型转换失败或	否	不涉及

参数名称	参数描述	是否必选	默认值
	mode 为 3（仅做预检）时，将预检结果保存在当前路径下。		
--input_fp16_nodes	不支持与--input_type 同时设置 指定数据类型为“fp16nchw”的输入节点名称。 例如：“node_name1;node_name2”。 注：该参数不推荐使用，尽量使用--input_type	否	不涉及
--is_output_fp16	不支持与--output_type 同时设置。 标注输出的数据类型是否为“fp16 nchw”。 例如：false, true, false, true。 注：该参数不推荐使用，尽量使用--output_type	否	False
--net_format	指定网络算子优先选用的数据格式 注：该参数现已不推荐使用，可通过网络推导得出	否	不涉及
--output_type	支持设定模型输出格式，支持指定为 FP32, FP16, INT32，UINT8 等，包括多输出和单输出。具体是否能成功生成离线模型，取决于原始模型是否支持指定为该格式的输出。模型输出支持场景详见表 3-3。 例如："output_name1：FP16；output_name2：UINT8" **说明** 本参数为 HiAI DDK V310 新增参数。	否	FP32
--input_type	支持设定模型输入格式，支持指定为 FP32, FP16, INT32， UINT8 等，包括多输入和单输入。具体是否能成功生成离线模型，取决于原始模型是否支持指定为该格式的输入。模型输入支持场景详见表 3-1 和表 3-2。 例如："input_name1:FP16;input_name2:UINT8" **说明** 本参数为 HiAI DDK V310 新增参数。	否	FP32
--weight_data_type	支持设定模型权值数据类型，支持指定为 FP32 或 FP16。该参数仅针对模型中权值数据类型为 FP32 时生效，根据设定将权值数据类型由 FP32 转为 FP16 存储或维持 FP32 不变。该参数默认值为 FP16。 例如："weight_data_type:FP16" **说明** 本参数为 HiAI DDK V320 新增参数。	否	FP16

须知

表 3-1、表 3-2、表 3-3 中未列出的情况一律不支持。

表3-1 非 AIPP 场景设定输入类型支持情况

原始模型实际输入	离线模型期望输入（用户设定）	针对 OMG 参数
FP32	FP16	--input_type
FP32	FP32	--input_type
FP16	FP16	--input_type
UINT8	UINT8	--input_type
INT32	INT32	--input_type
INT8	INT8	--input_type
INT16	INT16	--input_type
INT32	INT32	--input_type
BOOL	BOOL	--input_type
INT64	INT64	--input_type
DOUBLE	DOUBLE	--input_type

含 AIPP 场景：客户设定了输入输出数据类型，是否支持的场景如下：

表3-2 AIPP 场景设定输入类型支持情况

原始模型实际输入	离线模型期望输入	是否有 AIPP	针对 OMG 参数
FP32	UINT8	有	--input_type
FP16	UINT8	有	--input_type
UINT8	UINT8	有	--input_type

模型输出支持场景如下：

表3-3 设定输出类型支持情况

原始模型	离线模型输出	针对 OMG 参数
FP32	UINT8	--output_type
FP16	UINT8	--output_type
UINT8	UINT8	--output_type
FP16	FP16	--output_type
FP32	FP32	--output_type

原始模型	离线模型输出	针对 OMG 参数
FP32	FP16	--output_type
FP16	FP32	--output_type
INT32	INT32	--output_type
INT8	INT8	--output_type
INT16	INT16	--output_type
BOOL	BOOL	--output_type
INT64	INT64	--output_type
UINT32	UINT32	--output_type
DOUBLE	DOUBLE	--output_type

3.2 AIPP 参数

说明

本参数为 HiAI DDK V310 新增参数。

3.2.1 AIPP 简介

AIPP（AI Pre-Process）用于在硬件上完成图像预处理，包括改变图像尺寸、色域转换（转换图像格式）、减均值/乘系数（改变图像像素）。

AIPP 分为静态 AIPP 和动态 AIPP，两者使用严格区分，静态 AIPP 模型不能接收模型推理时传入的 AIPP 参数，不兼容动态 AIPP 场景，两者区别详见"表 3-4"。

表3-4 静态/动态 AIPP 区别

	设置 AIPP 参数方式	优点
静态 AIPP	在模型生成时通过配置文件或者 IR 定义预置	更高效率，模型加载阶段即可完成 AIPP 初始化
动态 AIPP	仅标记该模型具备 AIPP 处理功能，推理时通过接口外部传入，具体接口请参考《华为 HiAI_DDK_V320_模型推理集成指导》的 AIPP 对外接口类	更灵活，每次推理可传入不同 AIPP 参数

3.2.2 AIPP 支持的输入格式

AIPP 的可配置的图片格式如下：

- YUV420SP_U8
- XRGB8888_U8
- ARGB8888_U8
- YUYV_U8
- YUV422SP_U8
- AYUV444_U8
- YUV400_U8

格式后缀 U8 表示图片像素点为 Uint8 类型，范围为 0 到 255。当图片的输入为 YUV 类型时，无论是 YUV420 还是 YUV422 或者 YUYV，AIPP 自动将图片数据补齐为 YUV444 格式。

除以上列举的图片类型，AIPP 还可以通过开启 Channl Swap 通道交换功能，支持更加丰富的图片输入格式。

3.2.3 AIPP 支持的功能

AIPP 按照芯片的处理顺序，支持的功能如下：

- Crop 图片裁剪
- Channel Swap 通道交换
- Color Space Conversion 色域转换
- Resize 图片缩放
- Data Type Conversion 数据类型转换
- Padding 图片补边

3.2.3.1 Crop 图片裁剪

AIPP 的 Crop 功能用于对输入图片进行裁剪，涉及参数如下：

名称	描述	取值范围
switch	裁剪使能开关	false/true
load_start_pos_w	裁剪起始位置水平方向坐标	load_start_pos_w < src_image_size_w
load_start_pos_h	裁剪起始位置垂直方向坐标	load_start_pos_h < src_image_size_h
crop_size_w	裁剪出的图像宽度	load_start_pos_w + crop_size_w <= src_image_size_w
crop_size_h	裁剪出的图像高度	load_start_pos_h + crop_size_h <= src_image_size_h

YUV 类型的图片受图片自身类型的限制，当输入图片类型为 YUV420SP、YUYV、YUV422SP 和 AYUV444 时，裁剪的起始坐标和裁剪的宽高都应该是偶数，系统会进行校验。

3.2.3.2 Channel Swap 通道交换

AIPP 支持两种类型的通道交换：RB/UV 通道交换和 AX 通道交换。

RB/UV 通道交换丰富了输入图片的格式，开启 RB/UV 通道交换后，AIPP 支持的图片输入格式比可配置的输入类型丰富了一倍。

配置类型	可接受图片类型
YUV420SP_U8	YUV420，YVU420 + rbuv_swap_switch
XRGB8888_U8	XRGB，XBGR + rbuv_swap_switch
ARGB8888_U8	ARGB，ABGR + rbuv_swap_switch
YUYV_U8	YUYV，YVYU + rbuv_swap_switch
YUV422SP_U8	YUV422，YVU422 + rbuv_swap_switch
AYUV444_U8	AYUV + rbuv_swap_switch

当配置的图片输入格式为 XRGB、ARGB 或 AYUV 时，支持开启 AX 通道交换。开启通道交换后，图片第一个通道的输入被搬移到第四个通道上；即当 XRGB、ARGB 和 AYUV 开启 AX 通道交换后，转变为 RGBX、RGBA 和 YUVA。

当模型训练集为 RGB 格式的图片，而推理时的图片输入为 XRGB 或者 ARGB 时，可以通过使能 AX 通道交换，将 RGB 通道前移，实现兼容。

3.2.3.3 Color Space Conversion 色域转换

Color Space Conversion 色域转换（以下简称 CSC），特指在 YUV444 和 RGB888 两种图片格式之间进行转换。涉及如下配置参数：

名称	描述	类型	取值范围
csc_switch	色域转换开关	bool	true/false
matrix_r0c0 matrix_r0c1 matrix_r0c2 matrix_r1c0 matrix_r1c1 matrix_r1c2 matrix_r2c0 matrix_r2c1 matrix_r2c2	CSC 矩阵元素	int16	[-32677,32676]
output_bias_0 output_bias_1 output_bias_2	RGB 转 YUV 时的输出偏移	uint8	[0, 255]

名称	描述	类型	取值范围
input_bias_0 input_bias_1 input_bias_2	YUV 转 RGB 时的输入偏移	uint8	[0, 255]

参考 1. YUV 和 BGR 的转换公式：

YUV 转 BGR：

```
 | B |   | matrix_r0c0 matrix_r0c1 matrix_r0c2 | | Y - input_bias_0 |
 | G | = | matrix_r1c0 matrix_r1c1 matrix_r1c2 | | U - input_bias_1 | >>
8
 | R |   | matrix_r2c0 matrix_r2c1 matrix_r2c2 | | V - input_bias_2 |
```

BGR 转 YUV：

```
 | Y |   | matrix_r0c0 matrix_r0c1 matrix_r0c2 | | B |       |
output_bias_0 |
 | U | = | matrix_r1c0 matrix_r1c1 matrix_r1c2 | | G | >> 8 + |
output_bias_1 |
 | V |   | matrix_r2c0 matrix_r2c1 matrix_r2c2 | | R |       |
output_bias_2 |
```

参考 2，BT-601 narrow，JPEG 和 BT-709 narrow 三种类型图片的转换公式

BT-601 narrow:

```
Y = (66×R + 129×G + 25×B+128)/256+ 2^4
Cb = ((-38)×R + (-74)×G + 112×B+128)/256+ 2^7
Cr = (112×R + (-94)×G + (-18)×B+128)/256+ 2^7
R = (298×(Y-2^4) + 0×(Cb-2^7) + 409×(Cr-2^7)+128)/256
G = (298×(Y-2^4 )+(-100)×(Cb-2^7) +(-208)×(Cr-2^7)+128)/256
B = (298×(Y-2^4) + 516×(Cb-2^7) + 0×(Cr-2^7)+128)/256
```

JPEG:

```
Y = (77×R + 150×G + 29×B+128)/256
Cb = ((-43)×R + (-85)×G + 128×B+128)/256+ 2^7
Cr = (128×R + (-107)×G + (-21)×B+128)/256+ 2^7
R = (256×Y + 0×(Cb-2^7) + 359×(Cr-2^7)+128)/256
G = (256×Y+(-88)×(Cb-2^7) +(-183)×(Cr-2^7)+128)/256
B = (256×Y + 454× (Cb-2^7) + 0×(Cr-2^7)+128)/256
```

BT-709 narrow:

```
Y = (47×R + 157×G + 16×B+128)/256+ 2^4
```



```
Cb = ((-26)×R +(-87)×G + 112×B+128)/256+ 2^7
Cr = (112×R +(-102)×G +(-10)×B+128)/256+ 2^7
R = (298×(Y-2^4) + 0×(Cb-2^7) + 459×(Cr-2^7)+128)/256
G = (298×(Y-2^4) +(-55)×(Cb-2^7) +(136)×(Cr-2^7)+128)/256
B = (298×(Y-2^4) + 541×(Cb-2^7) + 0×(Cr-2^7)+128)/256
```

使用配置文件生成静态 AIPP 模型时，需要根据以上的公式配置 CSC 矩阵以及 input_bias 或者 output_bias 的值。使用 IR 定义的方式定义 AIPP CSC 功能算子时，以及使用 HiAI 接口配置 CSC 参数时，支持传入目标类型，由系统来填充 CSC 配置参数。

以下给出 JPEG 和 BT-601NARROW 两种图片类型下的 CSC 配置参考：

1） 输入为 YUV 格式图片，模型训练集为 RGB

JPEG	BT-601NARROW
matrix_r0c0 : 256 matrix_r0c1 : 0 matrix_r0c2 : 359 matrix_r1c0 : 256 matrix_r1c1 : -88 matrix_r1c2 : -183 matrix_r2c0 : 256 matrix_r2c1 : 454 matrix_r2c2 : 0 input_bias_0 : 0 input_bias_1 : 128 input_bias_2 : 128	matrix_r0c0 : 298 matrix_r0c1 : 0 matrix_r0c2 : 409 matrix_r1c0 : 298 matrix_r1c1 : -100 matrix_r1c2 : -208 matrix_r2c0 : 298 matrix_r2c1 : 516 matrix_r2c2 : 0 input_bias_0 : 16 input_bias_1 : 128 input_bias_2 : 128

2） 输入为 YUV 格式图片，模型训练集为 BGR

JPEG	BT-601NARROW
matrix_r0c0 : 256 matrix_r0c1 : 454 matrix_r0c2 : 0 matrix_r1c0 : 256 matrix_r1c1 : -88 matrix_r1c2 : -183 matrix_r2c0 : 256 matrix_r2c1 : 0 matrix_r2c2 : 359 input_bias_0 : 0	matrix_r0c0 : 298 matrix_r0c1 : 0 matrix_r0c2 : 409 matrix_r1c0 : 298 matrix_r1c1 : -100 matrix_r1c2 : -208 matrix_r2c0 : 298 matrix_r2c1 : 516 matrix_r2c2 : 0 input_bias_0 : 16

JPEG	BT-601NARROW
input_bias_1 : 128 input_bias_2 : 128	input_bias_1 : 128 input_bias_2 : 128

3） 输入为 YUV 格式图片，模型训练集为灰度图（YUV400_U8）

matrix_r0c0 : 256 matrix_r0c1 : 0 matrix_r0c2 : 0 matrix_r1c0 : 0 matrix_r1c1 : 0 matrix_r1c2 : 0 matrix_r2c0 : 0 matrix_r2c1 : 0 matrix_r2c2 : 0

4） 输入为 RGB 格式图片，模型训练集为灰度图（YUV400_U8）

matrix_r0c0 : 76 matrix_r0c1 : 150 matrix_r0c2 : 30 matrix_r1c0 : 0 matrix_r1c1 : 0 matrix_r1c2 : 0 matrix_r2c0 : 0 matrix_r2c1 : 0 matrix_r2c2 : 0

5） 输入为 RGB 类型图片，模型训练集为 YUV

JPEG	BT-601NARROW
matrix_r0c0 : 77 matrix_r0c1 : 150 matrix_r0c2 : 29 matrix_r1c0 : -43 matrix_r1c1 : -85 matrix_r1c2 : 128 matrix_r2c0 : 128 matrix_r2c1 : -107	matrix_r0c0 : 66 matrix_r0c1 : 129 matrix_r0c2 : 25 matrix_r1c0 : -38 matrix_r1c1 : -74 matrix_r1c2 : 112 matrix_r2c0 : 112 matrix_r2c1 : -94

JPEG	BT-601NARROW
matrix_r2c2 : -21 output_bias_0 : 0 output_bias_1 : 128 output_bias_2 : 128	matrix_r2c2 : -18 output_bias_0 : 16 output_bias_1 : 128 output_bias_2 : 128

6） 输入为 RGB 格式图片，模型训练集为 YVU

JPEG	BT-601NARROW
matrix_r0c0 : 77 matrix_r0c1 : 150 matrix_r0c2 : 29 matrix_r1c0 : 128 matrix_r1c1 : -107 matrix_r1c2 : -21 matrix_r2c0 : -43 matrix_r2c1 : -85 matrix_r2c2 : 128 output_bias_0 : 0 output_bias_1 : 128 output_bias_2 : 128	matrix_r0c0 : 66 matrix_r0c1 : 129 matrix_r0c2 : 25 matrix_r1c0 : 112 matrix_r1c1 : -94 matrix_r1c2 : -18 matrix_r2c0 : -38 matrix_r2c1 : -74 matrix_r2c2 : 112 output_bias_0 : 16 output_bias_1 : 128 output_bias_2 : 128

从使用的角度，将灰度图转成 RGB 没有意义，系统约束当输入格式配置为 YUV400_U8 时，不支持 CSC。

3.2.3.4 Resize 图片缩放

图片缩放涉及参数如下：

名称	描述	取值范围
switch	缩放使能开关	false/true
resize_input_w	缩放前图像宽度	resize_input_w <=8192 resize_output_w / resize_input_w ∈[1/16,16]
resize_input_h	缩放前图像高度	resize_output_h /resize_input_h∈[1/16,16]
resize_output_w	缩放后图像宽度	[16,8192]
resize_output_h	缩放后图像高度	>=16

说明

- Resize 子功能的 resize_input_w 和 resize_input_h 两个参数对用户不可见。当 Crop 功能关闭时，图片缩放前的大小取输入图片的大小，当 Crop 功能打开时，因为 AIPP 的 Resize 处理总是在 Crop 之后，图片缩放前的大小取图片裁剪后的大小。配置时，只需要关心缩放后的大小即可。
- 通过配置文件转换静态 AIPP 模型时，Crop 之后的大小 crop_size_w 和 crop_size_h，以及 Resize 之后的大小 resize_output_w 和 resize_output_h 可以省去不配置，前提是这两个参数可以通过计算获取。省略 resize_output_w 和 resize_output_h 时，Resize 功能这两个值取模型训练集的图片尺寸减去 AIPP Padding 之后的结果；当 Resize 不使用时，同理可省略 Crop 功能 crop_size_w 和 crop_size_h。

3.2.3.5 Data Type Conversion 数据类型转换

数据类型转换（Data Type Conversion）以下简称 DTC，DTC 用于将输入图片中像素值转换为模型训练时的数据类型。AIPP 允许用户设置 DTC 参数，使得转换之后的数据落到一个预期的范围，避免强制转换。

将 Uint8 类型的数据转换为 Int8 类型的数据，计算规则如下：

pixel_out_chx(i) = pixel_in_chx(i) - mean_chn_i

将 Uint8 类型的数据转换为 Float16 类型的数据，计算规则如下：

pixel_out_chx(i) = [pixel_in_chx(i) - mean_chn_i - min_chn_i] * var_reci_chn

DTC 涉及的配置参数如下表：

名称	描述	取值范围
switch	DTC 使能开关	false/true
mean_chn_0	通道 0 均值	[0,255]
mean_chn_1	通道 1 均值	[0,255]
mean_chn_2	通道 2 均值	[0,255]
mean_chn_3	通道 3 均值	[0,255]
min_chn_0	通道 0 最小值	[-65504, 65504]
min_chn_1	通道 1 最小值	[-65504, 65504]
min_chn_2	通道 2 最小值	[-65504, 65504]
min_chn_3	通道 3 最小值	[-65504, 65504]
var_reci_chn_0	通道 0 方差	[-65504, 65504]
var_reci_chn_1	通道 1 方差	[-65504, 65504]
var_reci_chn_2	通道 2 方差	[-65504, 65504]
var_reci_chn_3	通道 3 方差	[-65504, 65504]

说明

当 DTC 开关为 false 时，或者用户调用 HiAI 接口未传入 DTC 参数时，系统默认对图片输入数据进行类型强转，效果同通道均值和最小值均为 0，通道方差为 1。

3.2.3.6 Padding 图片补边

AIPP 的 Padding 功能用于对输入图片进行补边，涉及的参数如下：

名称	*描述*	*取值范围*
switch	Padding 使能开关	false/true
left_padding_size	图像左侧 padding 像素数	-
right_padding_size	图像右侧 padding 像素数	-
top_padding_size	图像上侧 padding 像素数	-
bottom_padding_size	图像下侧 padding 像素数	-

建议：上下左右的 Padding 值不要超过 32，如果 Padding 值过大，AIPP 将使用软件代码进行处理，效率低于硬件实现。

3.2.4 AIPP 配置文件说明

说明

1、以下示例为 DDK V320 优化配置文件，V310 及以前版本配置文件可参考"4 附录：旧版 AIPP 配置文件"。

2、当前使用 DDK V320 的优化配置文件生成的 DaVinci 模型无法在 HiAI 320 以前的 ROM 版本中使用。

AIPP 配置文件，用于通过 OMG 转换得到 DaVinci 模型时，为转换得到的 DaVinci 模型使能 AIPP，一份功能完整的 AIPP 配置文件示例如下：

```
# AIPP的配置以aipp_op开始，标识这是一个AIPP算子的配置，aipp_op支持配置多个
aipp_op {
# input_name参数为可选，标识对模型的哪个输入做AIPP处理
# 类型：string
input_name: "data"

# related_input_rank参数为可选，与input_name对应，推荐使用input_name，当模型输入名称未知时，可使用related_input_rank标识对模型的第几个输入做AIPP处理
# 类型：uint32
related_input_rank:0
```



```
# node_after_aipp参数为可选，用于当一个输入之后有多个分支，需要对分支进行不同的AIPP处理的场景，且需满足均配置或者均不配置，不能只配置一部分分支
# 类型：string
node_after_aipp: "conv01"

# input_edge_idx参数为可选，与node_after_aipp对应，当输入Data算子之后的若干算子名称重复时，可以使用input_edge_idx标识对第几个分支进行AIPP处理
# 类型：uint32
input_edge_idx: 0

input_para {
# 输入图片的类型
# 类型：enum
# 取值范围：[YUV420SP_U8, XRGB8888_U8, ARGB8888_U8, YUYV_U8, YUV422SP_U8, AYUV444_U8, YUV400_U8]
format: AYUV444_U8

shape {
# 输入图片的宽度、高度
# 类型：uint32
# 取值范围 & 约束： [0,4096]、对于除了YUV400之外的YUV类型的图片，要求取值是偶数
src_image_size_w: 800
src_image_size_h: 600
}

# max_src_image_size用于动态AIPP的场景，当图片的长宽或者输入类型不确定时，设置输入图片最大的size
# 类型：uint32
max_src_image_size: 102400
}

# == Crop参数设置，相关参数的含义及取值范围请参考3.2.3.1节 == #
crop_func {
switch: true
load_start_pos_w: 50
load_start_pos_h: 50
crop_size_w: 400
crop_size_h: 400
}
```



```
# == Channel Swap参数设置，相关参数的含义及取值范围请参考3.2.3.2节 == #
swap_func {
rbuv_swap_switch: true
ax_swap_switch: true
}

# == Resize参数设置，相关参数的含义及取值范围请参考第3.2.3.4节 == #
resize_func {
switch: true
resize_output_w: 200
resize_output_h: 200
}

# == Color Space Conversion参数设置，相关参数的含义及取值范围请参考第3.2.3.3节
== #
csc_func {
switch: true
matrix_r0c0: 256
matrix_r0c1: 0
matrix_r0c2: 259
matrix_r1c0: 256
matrix_r1c1: -88
matrix_r1c2: -183
matrix_r2c0: 256
matrix_r2c1: 454
matrix_r2c2: 0
output_bias_0: 0
output_bias_1: 0
output_bias_2: 0
input_bias_0: 16
input_bias_1: 128
input_bias_2: 128
}

# == Data Type Conversion参数设置，相关参数的含义及取值范围请参考第3.2.3.5节 ==
#
dtc_func {
switch: true
mean_chn_0: 0
mean_chn_1: 0
mean_chn_2: 0
mean_chn_3: 0
min_chn_0: 0
```



```
min_chn_1: 0
min_chn_2: 0
min_chn_3: 0
var_reci_chn_0: 1.0
var_reci_chn_1: 1.0
var_reci_chn_2: 1.0
var_reci_chn_3: 1.0
}

# == Padding参数设置，相关参数的含义及取值范围请参考第3.2.3.6节 == #
padding_func {
switch: true
left_padding_size: 12
right_padding_size: 12
top_padding_size: 12
bottom_padding_size: 12
}
}
```

3.2.4.1 AIPP 配置的多输入支持

AIPP 支持对一个多输入模型的多个输入分别配置 AIPP，也支持在一个输入 Data 算子有多个输出分支的情况下，对不同的输出分支分别配置 AIPP。

AIPP 配置的多输入支持由 2 组共 4 个配置参数控制：input_name 和 related_input_rank 用于指定对哪一个输入进行 AIPP 处理，node_after_aipp 和 input_edge_idx 用于指定对 Data 算子的多个输出中的哪一个输出进行 AIPP 处理。

input_name 和 related_input_rank 两个参数推荐使用 input_name，related_input_rank 参数用于模型输入名称不确定的场景，如果同时配置了这两个参数，则两个参数互为校验；如果两个参数都没有被配置，默认对模型的第一个输入进行 AIPP 处理。

node_after_aipp 和 input_edge_idx 两个参数推荐使用 node_after_aipp，input_edge_idx 用于 Data 算子的多个输出分支衔接的算子名称重复或不确定的场景，如果同时配置这两个参数，则两个参数互为校验；如果两个参数都没有被配置，则该 Data 算子的所有输出分支使用同一个 AIPP 处理。

3.2.4.2 AIPP 配置区分动态 AIPP 与静态 AIPP

如果所有的 AIPP 子功能的配置开关均为 false，则生成的 DaVinci 模型为动态 AIPP 模型，需要在模型推理阶段传入 AIPP 配置参数；相反只要有一个 AIPP 子功能的配置开关为 true，则生成的 DaVinci 模型为静态 AIPP 模型，模型推理阶段使用配置文件中定义的 AIPP 配置参数。

对于动态 AIPP 的场景，AIPP 可以允许输入图片的长宽，以及图片类型不确定，对应即 src_image_size_w, src_image_size_h 和 input_format 三个参数不配置，此时用户需要指定动态 AIPP 处理时的最大的图片尺寸，配置 max_src_image_size。

4 附录：旧版 AIPP 配置文件

```
# AIPP的配置以aipp_op开始，标识这是一个AIPP算子的配置，aipp_op支持配置多个
aipp_op {
#
# AIPP当前支持色域转换、抠图、减均值、乘系数、通道数据交换、单行模式的能力。
# 输入图片的类型仅支持UINT8格式。
# 使用此配置文件时，请将需要配置的参数去注释，并改为合适的值。
# 模板中参数值为默认值，其中input_format属性为必选属性，其余属性均为可选配置。
#========================= 全局设置 =========================
# aipp_mode指定了AIPP的模式，必须配置
# 类型：enum
# 取值范围：static，static 表示静态AIPP，当前仅支持static
# aipp_mode: static

# related_input_rank参数为可选，标识对模型的第几个输入做AIPP处理，从0开始，默认为
0。例如模型有两个输入，需要对第2个输入做AIPP，则配置related_input_rank为1。
# 类型：整型
# 配置范围 >= 0
# related_input_rank: 0

# input_edge_idx参数为可选，如果一个模型输入为多个算子共有，即Data算子后面跟着多个算
子，配置该参数，对Data算子的不同的输出边做不同的AIPP处理。
# 类型：整型
# 配置范围 >= 0
# input_edge_idx: 0

# input_format输入图像类型参数
# 类型：enum
# 取值范围：
```



```
YUV420SP_U8/XRGB8888_U8/ARGB8888_U8/YUYV_U8/YUV422SP_U8/AYUV444_U8/YUV400
_U8
# input_format :

# 图像的宽度、高度
# 类型：uint16
# 取值范围 & 约束： (0,4096]、对于YUV420SP_U8类型的图像，要求取值是偶数
# 说明：请根据实际图片的宽、高配置src_image_size_w、src_image_size_h，若不设置或
设置为0，则会取网络输入定义的w和h
# src_image_size_w :0
# src_image_size_h :0
约束条件：
（1）取值为16的倍数
（2）如果没有crop，resize或者padding，src_image_size可以不配置，或者与模型输入的宽
高保持一致；如果配置了crop，resize或者padding其中之一，src_image_size_w与
src_image_size_h就必须配置，且大于0；

#========================= crop参数设置 ====================
# AIPP处理图片时是否支持抠图
# 类型：bool
# 取值范围：true/false，true表示支持，false表示不支持
# crop :false

# 抠图起始位置水平、垂直方向坐标，抠图大小为网络输入定义的w和h
# 类型：uint16
# 取值范围 & 约束： [0,4096]、对于YUV420SP_U8类型的图像，要求取值是偶数
#说明：load_start_pos_w加上网络输入定义的w需要小于等于src_image_size_w，
load_start_pos_h加上网络输入定义的h需要小于等于src_image_size_h
# load_start_pos_w :0
# load_start_pos_h :0

# 抠图后的图像size
# 类型：uint16
# 取值范围 & 约束： [0,4096]、偶数、load_start_pos_w + crop_size_w <=
src_image_size_w、load_start_pos_h + crop_size_h <= src_image_size_h
# crop_size_w :0
# crop_size_h :0

#=================== resize参数设置 ==================
# AIPP处理图片时是否支持缩放，保留字段
```



```
# 类型：bool
# 取值范围：true/false，true表示支持，false表示不支持
resize :false
# 缩放后图像的宽度和高度，保留字段
# 类型：uint16s
# 取值范围 & 约束： [0,4096]、偶数、小于src_image_size
resize_output_w :0
resize_output_h :0

#========================= padding参数设置 =================
# AIPP处理图片时padding使能开关，保留字段
# 类型：bool
# 取值范围：true/false，true表示支持，false表示不支持
# padding :false
# C方向的填充值，静态AIPP配置，保留字段
# 类型： float16
# c_padding_value :0.0
# left_padding_size :0
# right_padding_size :0
# top_padding_size :0
# bottom_padding_size :0

#========================= 色域转换参数设置 ================
# 色域转换开关，静态AIPP配置
# 类型：bool
# 取值范围：true/false
# csc_switch :false

# 色域转换前，R通道与B通道交换开关/U通道与V通道交换开关
# 类型：bool
# 取值范围：true/false
# rbuv_swap_switch :false

# 色域转换前，RGBA->ARGB, YUVA->AYUV交换开关
# 类型：bool
# 取值范围：true/false
# ax_swap_switch :false

# 单行处理模式（只处理抠图后的第一行）开关
# 类型：bool
```



```
# 取值范围：true/false
# single_line_mode :false

# 若色域转换开关为false，则本功能旁路。
# 若输入图片通道数为4，则忽略第一通道。
# YUV转BGR：
# | B | | matrix_r0c0 matrix_r0c1 matrix_r0c2 | | Y - input_bias_0 |
# | G | = | matrix_r1c0 matrix_r1c1 matrix_r1c2 | | U - input_bias_1 | >>
8
# | R | | matrix_r2c0 matrix_r2c1 matrix_r2c2 | | V - input_bias_2 |
# BGR转YUV：
# | Y | | matrix_r0c0 matrix_r0c1 matrix_r0c2 | | B | | output_bias_0 |
# | U | = | matrix_r1c0 matrix_r1c1 matrix_r1c2 | | G | >> 8 + |
output_bias_1 |
# | V | | matrix_r2c0 matrix_r2c1 matrix_r2c2 | | R | | output_bias_2 |
# 3*3 CSC矩阵元素
# 类型：int16
# 取值范围：[-32768 ,32767]
# matrix_r0c0 :298
# matrix_r0c1 :516
# matrix_r0c2 :0
# matrix_r1c0 :298
# matrix_r1c1 :-100
# matrix_r1c2 :-208
# matrix_r2c0 :298
# matrix_r2c1 :0
# matrix_r2c2 :409

# RGB转YUV时的输出偏移
# 类型：uint8
# 取值范围：[0, 255]
# output_bias_0 :16
# output_bias_1 :128
# output_bias_2 :128

# YUV转RGB时的输入偏移
# 类型：uint8
# 取值范围：[0, 255]
# input_bias_0 :16
# input_bias_1 :128
# input_bias_2 :128
```



```
#====================== 减均值、乘系数设置 ===============
# 计算规则如下：
# 当uint8->uint8时，本功能旁路
# 当uint8->int8时，pixel_out_chx(i) = pixel_in_chx(i) - mean_chn_i
# 当uint8->fp16时，pixel_out_chx(i) = [pixel_in_chx(i) - mean_chn_i -
min_chn_i] * var_reci_chn

# 通道n均值
# 类型：uint8
# 取值范围：[0, 255]
# mean_chn_0 :0
# mean_chn_1 :0
# mean_chn_2 :0

# 通道n最小值
# 类型：float16
# 取值范围：[-65504, 65504]
# min_chn_0 :0.0
# min_chn_1 :0.0
# min_chn_2 :0.0

# 通道n方差或(max-min)的倒数
# 类型：float16
# 取值范围：[-65504, 65504]
# var_reci_chn_0 :1.0
# var_reci_chn_1 :1.0
# var_reci_chn_2 :1.0
}
```

