君正®

venus 推理框架

使用指南

Date: Jan. 2022

北京君正集成电路股份有限公司
Ingenic Semiconductor Co., Ltd.

君正®

venus推理框架

使用指南

Release history

Date	Author	Revision	Change
Jan.2022	zhliu	1.0.1	First release
Feb.2022	zfni	1.0.2	1、Add image preprocessing class sample

Disclaimer

This documentation is provided for use with Ingenic products. No license to Ingenic property rightsis granted. Ingenic assumes no liability, provides no warranty either expressed or implied relatingto the usage, or intellectual property right infringement except as provided for by Ingenic Terms and Conditions of Sale.

Ingenic products are not designed for and should not be used in any medical or life sustaining orsupporting equipment.

All information in this document should be treated as preliminary. Ingenic may make changes tothis document without notice. Anyone relying on this documentation should contact Ingenic forthe current documentation and errata.

Ingenic Semiconductor Co., Ltd.

Ingenic Headquarters, East Bldg. 14, Courtyard #10
Xibeiwang East Road, Haidian District, Beijing, China,
Tel: 86-10-56345000
Fax:86-10-56345001
Http: //www.ingenic.com

目 录

1 简介 2

2 API 介绍 2

2.1 数据结构 2
2.2 前向推理 API 4
2.3 工具类 API 7
2.4 Inference 流程 22
2.4.1 算子集合 24
2.4.1.1 Convolution 功能说明 25
2.4.1.2 DepthWiseConvolution 功能说明 25
2.4.1.3 FullConnected 功能说明 25
2.4.1.4 Add 功能说明 26
2.4.1.5 Mul 功能说明 26
2.4.1.6 Pooling 功能说明 26
2.4.1.7 Upsample 功能说明 27
2.4.1.8 Concat 功能说明 27
2.4.1.9 Normalize 功能说明 27
2.4.1.10 LSTM 功能说明 27
2.4.1.11 Eltwise 功能说明 27
2.4.1.12 Sum 功能说明 28
2.4.1.13 Softmax 功能说明 28
2.4.1.14 Embedding 功能说明 28
2.4.1.15 Padding 功能说明 28
2.4.1.16 Slice 功能说明 29
2.4.1.17 Permute 功能说明 29
2.4.2 Profiler 工具 29
2.4.2.1 开启方法 29
2.4.2.2 使用示例 29
2.4.3 完整示例 30

1 简介

Venus 是一个轻量高效、易于拓展的高性能的深度学习预测框架。支持 8bit/4bit/2bit 量化推理，支持 8bit 后量化推理和混合精度推理，支持常用卷积、Depthwise 卷积、常用池化、常用激活、全连接、LSTM、Concat 等算子，支持多输入和具有分支结构的网络，基于 AIE 平台的高性能 NPU 加速器和 SIMD512 加速引擎实现，针对 AIE 平台的硬件特点，对底层算子进行了深度优化，能够充分发挥 AIE 平台的计算性能。通过高效精巧的内存管理结构设计，实现无计算依赖的节点间内存高效复用，大幅度降低内存资源消耗，同时支持跨模型内存复用，进一步降低内存消耗，提高内存利用率。另外还支持基于硬件的图像抠图、缩放、旋转、仿射变换、透视变换和色彩空间转换等高速图像处理操作，目前主要应用于 T40、T02 和 X2500 芯片平台。

2 API 介绍

2.1 数据结构

- **Tensor**
  - 定义

    表示多维的 tensor 数据，主要的数据排放格式为 NHWC 格式
  - 接口

    shape　获取 tensor 的形状信息[batch, height, width, channel]

    reshape　根据输入的形状，改变 tensor 的形状信息

    mudata　获取指定类型的 tensor 数据的可读可写指针

    data　获取指定类型的 tensor 数据的只读指针
- **shape_t**
  - 定义

    表示 Tensor 的形状信息
  - 属性

    batch　四维 tensor 的 batch

    height　tensor 的高

    width　tensor 的宽

    channel　tensor 的通道
- **Bbox_t**
  - 定义

    表示目标的坐标框
  - 属性

    x0　目标框左上角坐标

    y0　目标框左上角坐标

    x1　目标框右下角坐标

    y1　目标框右下角坐标
- **ObjBbox_t**
  - 定义

    表示目标检测网络输出的坐标框、对应的类别和置信度

- 属性

  box 目标的坐标框
  score 目标框对应的置信度
  class_id 目标的类别

- **PaddingType**
  - 定义

    图像边界填充的类型
  - 属性

    NONE 不填充
    BOTTOM_RIGHT 右下填充
    SYMMETRY 四周对称填充
- **AddressLocate**
  - 定义

    输入指针的地址来源
  - 属性

    NMEM_VIRTUAL 指针是来自 nmem 的虚拟地址
    RMEM_PHYSICAL 指针是来自 rmem 的物理地址
- **ChannelLayout**
  - 定义

    图像格式
  - 属性

    NONE 默认为空
    NV12 NV12 图像格式
    BGRA BGRA 图像格式
    RGBA RGBA 图像格式
    ARGB ARGB 图像格式
    RGB RGB 图像格式
- **BsPadType**
  - 定义

    图像边界填充的类型
  - 属性

    NONE 不填充
    BOTTOM_RIGHT 右下填充
    SYMMETRY 四周对称填充
- **BsBaseParam**
  - 定义

    图像处理基本参数
  - 属性

    in_layout 输入图像格式
    out_layout 输出图像格式
    coef_off_enable 是否使用默认转换参数
    coef nv12 转 BGRA 参数
    offset nv12 转 BGRA 像素偏移

- **BsExtendParam**
  - 定义
    图像处理参数
  - 属性

pad_type	图像边界填充的类型(BsPadType)
pad_val	边界像素填充值(BGR)

- **BsCommonParam**
  - 定义
    图像处理调用 common 接口参数
  - 属性

input_height	输入图像高
input_width	输入图像宽
input_line_stride	输入图像行间距

2.2 前向推理 API

- **venus_init**
  - 功能描述
    初始化运行环境，返回状态码
  - 接口定义
    int venus_init(void);
  - 参数列表
    无
  - 返回值
    等于 0 成功，否则失败
- **venus_deinit**
  - 功能描述
    释放运行环境，返回状态码
  - 接口定义
    int venus_deinit();
  - 参数列表
    无
  - 返回值
    等于 0 成功，否则失败
- **BaseNet**
  - 定义
    Venus 框架的推理引擎基础类
  - **load_model**
    - 功能描述
      从内存或者文件中加载模型参数
    - 接口定义
      int load_model(const char model_path, bool memory_model=false, int start_off=0);*
    - 参数列表

model_path　当 memory_model 为 true 时， 表示指向模型参数内存区的指针；当 memory_model 为 false 时，表示模型文件的文件名。

memory_model 表示是否从内存中加载模型，设置为 true 时表示从内存中加载模型，设置为 false 时表示从文件中加载模型，默认为 false。目前只支持 false。

start_off 表示从 model_path 指定的文件首地址或内存首地址跳过的字节数。用于多模型打包加载。

- 返回值

  0 表示成功，否则失败

- **get_forward_memory_size**
  - 功能描述

    获取模型前向推理阶段所需的内存空间大小
  - 接口定义

    int get_forward_memory_size(size_t &memory_size);
  - 参数列表

    memory_size　　存放内存大小的变量引用
  - 返回值

    0 表示成功，否则失败
- **int set_forward_memory**
  - 功能描述

    当内存共享模式为外部设置内存时，通过该接口设置前向推理内存的首地址
  - 接口定义

    *int set_forward_memory(void *memory);*
  - 参数列表

    memory　　内存的首地址
  - 返回值

    0 表示成功，否则失败
- **get_input**
  - 功能描述

    通过网络输入节点的序号获取网络输入节点的 Tensor，如果网络有多个输入，可多次调用该接口获取。
  - 接口定义

    std::unique_ptr<Tensor> get_input(int index);
  - 参数列表

    index　网络输入节点的序号
  - 返回值

    返回指向网络输入节点 Tensor 的智能指针
- **get_input_by_name**
  - 功能描述

    通过网络输入节点的名字获取网络输入节点的 Tensor，如果网络有多个输入，可多次调用该接口获取。
  - 接口定义

    std::unique_ptr<Tensor> get_input_by_name(std::string &name);
  - 参数列表

name　网络输入节点的名字

- 返回值

  返回指向网络输入节点 Tensor 的智能指针

- **get_output**
  - 功能描述

    通过网络输出节点的序号获取网络输出节点的 Tensor，如果网络有多个输出，可多次调用该接口获取。
  - 接口定义

    std::unique_ptr<const Tensor> get_output(int index);
  - 参数列表

    index　网络输出节点的序号
  - 返回值

    返回指向网络输出节点 Tensor 的智能指针
- **get_output_by_name**
  - 功能描述

    通过网络输出节点的名字获取网络输出节点的 Tensor，如果网络有多个输出，可多次调用该接口获取。
  - 接口定义

    std::unique_ptr<const Tensor> get_output_by_name(std::string &name);
  - 参数列表

    name　网络输出节点的名字
  - 返回值

    返回指向网络输出节点 Tensor 的智能指针
- **run**
  - 功能描述

    执行模型前向推理，返回状态码
  - 接口定义

    int run();
  - 参数列表

    无
  - 返回值

    0 表示成功，否则失败

- **net_create**
- 功能描述

  创建推理引擎句柄，返回引擎句柄
- 接口定义

  std::unique_ptr<BaseNet> net_create(
  TensorFormat input_data_fmt=TensorFormat::NHWC,
  ShareMemoryMode smem_mode=ShareMemoryMode::DEFAULT);
- 参数列表

  input_data_fmt 表示输入图像的类型，有 NHWC、NV12 两种选项。

  smem_mode 表示内存共享的方式， 有 DEAFAULT、SHARE_ONE_THREAD、SET_FROM_EXTERNAL 三种选项，分别表示模型的内存共享方式、同一个线程中

的多个模型共享内存和用户从外部开辟推理内存等三种内存共享模式。

- 返回值

  返回推理引擎句柄，非 NULL 表示成功，否则失败

2.3 工具类 API

- 图像预处理类
  - **warp_resize**
    - 功能描述

      对输入图像执行缩放操作，返回缩放后的图像
    - 接口定义

      *int warp_resize(const Tensor &input, Tensor &output, BsExtendParam *param)*
    - 参数列表

      input　　输入图像

      output　　输出图像

      param　　图像缩放参数
    - 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048
output	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048

    - 返回值

      0 表示成功，否则失败
    - 注意事项
      - 当输入输出图像的宽高比不一致时，可以通过 param->padtype 选择输出直接缩放图像、四周填充的等比例缩放图像、右下填充的等比例缩放图像。
      - 输入为 NV12 格式时，输入宽高要求为 2 的倍数；输入为 BGRA、RGBA、ARGB 格式时，输入宽要求为 2 的倍数，输出格式与输入格式保持一致。
      - 输出为 NV12 格式时，输出宽高要求为 2 的倍数；输出为 BGRA、RGBA、ARGB 格式时，输出宽高无对齐要求。
  - **crop_resize**
    - 功能描述

      根据输入的 boxes 坐标集合对输入图像执行抠图缩放操作，返回抠图缩放后的图像集合。
    - 接口定义

      *int crop_resize(const Tensor &input, std::vector<Tensor> &output, std::vector<Bbox_t>&boxes,BsExtendParam *param)*
    - 参数列表

      input　　输入图像

      output　　输出图像集合

      boxes　　需要缩放的 box 的坐标及宽高的集合

      param　　图像缩放参数
    - 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048
output	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048
boxes	-	见注意事项	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项
  - 当抠的 ROI 图像和输出图像的宽高比不一致时，可以通过 param->padtype 选择输出直接缩放图像、四周填充的等比例缩放图像、右下填充的等比例缩放图像。
  - 输入为 NV12 格式时，即 Bbox_t 数据结构中的 x0,y0,x1,y1 均必须为偶数；输入为 BGRA、RGBA、ARGB 格式时，输入宽（x1-x0）要求为 2 的倍数，输出格式与输入格式保持一致。
  - 输出为 NV12 格式时，输出宽高要求为 2 的倍数；输出为 BGRA、RGBA、ARGB 格式时，输出宽高无对齐要求。

- **warp_affine**
  - 功能描述

    根据仿射变换矩阵对输入图像执行仿射变换操作
  - 接口定义

    *int warp_affine(const Tensor &input, Tensor &output,Tensor &matrix, BsExtendParam *param)*
  - 参数列表

    input 输入图像
    output 输出图像
    matrix 仿射变换矩阵的集合
    param 图像仿射变换参数
  - 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048
output	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048

  - 返回值

    0 表示成功，否则失败
  - 注意事项
    - 输入为 NV12 格式时，输入宽高要求为 2 的倍数；输入为 BGRA、RGBA、ARGB 格式时，输入宽要求为 2 的倍数，输出格式与输入格式保持一致。
    - 输出为 NV12 格式时，输出宽高要求为 2 的倍数；输出为 BGRA、RGBA、ARGB 格式时，输出宽高无对齐要求。

- **crop_affine**
  - 功能描述

    根据输入的 boxes 在原图上执行抠图操作，然后对抠完的图根据各自的仿射变换矩阵执行仿射变换
  - 接口定义

*int crop_affine(const Tensor &input,std::vector<Tensor> &output, std::vector<Tensor> &matrix, std::vector<Bbox_t>&boxes,BsExtendParam *param)*

◆ 参数列表

input 输入图像
output 输出图像集合
matrix 仿射变换矩阵的集合
boxes 需要仿射变换的 box 的坐标及宽高的集合
param 图像仿射变换参数

◆ 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048
output	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048
boxes	-	见注意事项	2x2 ~ 1024x2048

◆ 返回值

0 表示成功，否则失败

◆ 注意事项

- 输入为 NV12 格式时，即 Bbox_t 数据结构中的 x0,y0,x1,y1 均必须为偶数；输入为 BGRA、RGBA、ARGB 格式时，输入宽（x1-x0）要求为 2 的倍数，输出格式与输入格式保持一致。
- 输出为 NV12 格式时，输出宽高要求为 2 的倍数；输出为 BGRA、RGBA、ARGB 格式时，输出宽高无对齐要求。

■ **warp_perspective**

◆ 功能描述

根据透视变换矩阵对输入图像(NV12)执行透视变换操作

◆ 接口定义

*int warp_perspective(const Tensor &input, Tensor &output,Tensor &matrix, BsExtendParam *param)*

◆ 参数列表

input 输入图像
output 输出图像
matrix 仿射变换矩阵的集合
param 图像透视变换参数

◆ 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048
output	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048

◆ 返回值

0 表示成功，否则失败

◆ 注意事项

- 输入为 NV12 格式时，输入宽高要求为 2 的倍数；输入为 BGRA、RGBA、ARGB 格式时，输入宽要求为 2 的倍数，输出格式与输入格式保持一致。

- 输出为 NV12 格式时，输出宽高要求为 2 的倍数；输出为 BGRA、RGBA、ARGB 格式时，输出宽高无对齐要求。

■ **crop_perspective**

◆ 功能描述

根据输入的 boxes 在原图上执行抠图操作，然后对抠完的图根据各自的透视变换矩阵执行透视变换

◆ 接口定义

*int crop_perspective(const Tensor &input,std::vector<Tensor> &output, std::vector<Tensor> &matrix, std::vector<Bbox_t>&boxes,BsExtendParam *param)*

◆ 参数列表

input	输入图像
output	输出图像集合
matrix	仿射变换矩阵的集合
boxes	需要仿射变换的 box 的坐标及宽高的集合
param	图像透视变换参数

◆ 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048
output	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048
boxes	-	见注意事项	2x2 ~ 1024x2048

◆ 返回值

0 表示成功，否则失败

◆ 注意事项

- 输入为 NV12 格式时，即 Bbox_t 数据结构中的 x0,y0,x1,y1 均必须为偶数；输入为 BGRA、RGBA、ARGB 格式时，输入宽（x1-x0）要求为 2 的倍数，输出格式与输入格式保持一致。
- 输出为 NV12 格式时，输出宽高要求为 2 的倍数；输出为 BGRA、RGBA、ARGB 格式时，输出宽高无对齐要求。

■ **common_resize**

◆ 功能描述

对输入图像（NV12）执行缩放处理，返回缩放后的图像数据(BGRA/RGBA/ARGB/NV12)。

◆ 接口定义

*common_resize(const void *input, Tensor &output, AddressLocate input_locate,BsCommonParam *param)*

◆ 参数列表

input	输入数据指针
output	输出图像
input_locate	输入图像地址属性
param	图像缩放参数

◆ 参数规格

参数名称	支持图像类型	对齐要求	分辨率

input	NV12	宽高偶对齐	2x2 ~ 1024x2048
output	BGRA、RGBA、ARGB、NV12	见注意事项	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项
  - 当输入输出图像的宽高比不一致时，可以通过 padtype 选择输出直接缩放图像、四周填充的等比例缩放图像、右下填充的等比例缩放图像。
  - 输出为 NV12 格式时，输出宽高要求为 2 的倍数；输出为 BGRA、RGBA、ARGB 格式时，输出宽高无对齐要求。
  - 输入图像的内存必须来自物理地址连续的内存空间（rmem/nmem）。当 input_locate 为 NMEM_VIRTUAL 时，input 必须为 nmem 上的虚拟地址（Tensor 的 data/mudata 接口获取的指针，或者通过 nmem_memalign 申请的内存指针）；当 input_locate 为 RMEM_PHYSICAL 时，input 必须为 rmen 上的物理地址（通过 isvp 平台的相关接口获取的视频流物理地址）。

- **similar_transform**
  - 功能描述

    根据输入输出坐标，计算出相似变换矩阵。
  - 接口定义

    similar_transform(Tensor &input_src, Tensor &input_dst,Tesnor &output)
  - 参数列表

    input_src　　输入矩阵坐标

    Input_dst　　输出矩阵坐标

    output　　相似变换矩阵参数
  - 参数规格

参数名称	数据格式
input_src	1*3*2*1
input_dst	1*3*2*1
output	1*3*3*1

  - 返回值

    0 表示成功，否则失败
  - 注意事项
    - 调用函数生成一个 3*3 的矩阵参数，用于后续相似变换计算

- **get_affine_transform**
  - 功能描述

    根据输入输出坐标，计算出仿射变换矩阵。
  - 接口定义

    get_affine_transform(Tensor &input_src, Tensor &input_dst,Tesnor &output)
  - 参数列表

    input_src　　输入矩阵坐标

    Input_dst　　输出矩阵坐标

    output　　相似变换矩阵参数
  - 参数规格

参数名称	数据格式

input_src	1*3*2*1
input_dst	1*3*2*1
output	1*3*2*1

- 返回值

  0 表示成功，否则失败

- 注意事项
  - 调用函数生成一个 3*2 的矩阵参数，用于后续仿射变换计算

（以下为旧版本 API，之后将不再更新，建议使用上文描述 API）

warp_covert_nv2bgr

- 功能描述

  对输入的图像(NV12)执行色彩空间转换功能，返回转换之后的图像(BGRA)

- 接口定义

  int warp_covert_nv2bgr(const Tensor &input, Tensor &output)

- 参数列表

  input　　输入图像

  output　　输出图像

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	NV12	宽高为 2 的倍数	2x2 ~ 1024x2048
output	BGRA	宽高为 2 的倍数	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项
  - YUV 转 BGR 的公式（BT601）如下：

    B = 1.164(Y - 16) + 2.018(U - 128)

    G = 1.164(Y - 16) - 0.391(U - 128) - 0.813(V - 128)

    R = 1.164(Y - 16) + 1.596(V - 128)
  - 其他格式（BT709/BT656 等）的转换参数可配置。

warp_resize_bgra

- 功能描述

对输入图像(BGRA)执行缩放操作，返回缩放后的图像(BGRA)

- 接口定义

  int warp_resize_bgra(const Tensor &input，Tensor &output,
  PaddingType padtype,uint8_t padval)

- 参数列表

  input　　输入图像

  output　　输出图像

  padtype　　边界像素填充方式

  padval　　边界像素填充值(BGR)

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率

input	BGRA	宽为 2 的倍数	2x2 ~ 1024x2048
output	BGRA	-	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项
  - 当输入输出图像的宽高比不一致时，可以通过 padtype 选择输出直接缩放图像、四周填充的等比例缩放图像、右下填充的等比例缩放图像。

■ **warp_resize_nv12**

- 功能描述

  对输入图像(NV12)执行缩放操作，返回缩放后的图像(BGRA/NV12)

- 接口定义

  int warp_resize_nv12(const Tensor &input, Tensor &output,
  bool cvtbgra,PaddingType padtype, uin8_t padval)

- 参数列表

  input　　输入图像
  output　　输出图像
  cvtbgra　　是否将输出图像转成 BGRA
  padtype　　边界像素填充方式
  padval　　边界像素填充值(BGR)

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	NV12	宽高为 2 的倍数	2x2 ~ 1024x2048
output	NV12、BGRA	见注意事项	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项
  - 当输入输出图像的宽高比不一致时，可以通过 padtype 选择输出直接缩放图像、四周填充的等比例缩放图像、右下填充的等比例缩放图像。
  - 输出为 NV12 格式时，输出宽高要求为 2 的倍数；输出为 BGRA 格式时，输出宽高无对齐要求。

■ **crop_resize_bgra**

- 功能描述

  根据输入的 boxes 坐标集合对输入图像（BGRA）执行抠图缩放操作，返回抠图缩放后的图像(BGRA)集合。

- 接口定义

  int crop_resize_bgra(const Tensor &input, std::vector<Tensor> &output,
  std::vector<Bbox_t>&boxes, PaddingType padtype, uint8_t padval)

- 参数列表

  input　　输入图像

output 输出图像集合
boxes 需要缩放的 box 的坐标及宽高的集合
padtype 边界像素填充方式
padval 边界像素填充值(BGR)

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA	宽为 2 的倍数	2x2 ~ 1024x2048
output	BGRA	-	2x2 ~ 1024x2048
boxes	-	见注意事项	2x2 ~ 1024x2048

- 返回值

0 表示成功，否则失败

- 注意事项
  - 当抠的 ROI 图像和输出图像的宽高比不一致时，可以通过 padtype 选择输出直接缩放图像、四周填充的等比例缩放图像、右下填充的等比例缩放图像。
  - boxes 中每个 box 的宽必须为 2 的倍数，即 Bbox_t 的(x1-x0)为 2 的倍数。

■ **crop_resize_nv12**

- 功能描述

根据输入的 boxes 坐标集合对输入图像（NV12）执行抠图缩放操作，返回抠图缩放后的图像(NV12)集合。

- 接口定义

int crop_resize_nv12(const Tensor &input, std::vector<Tensor> &output, std::vector<Bboxt>&boxes, PaddingType padtype, uint8_t padval)

- 参数列表

input 输入图像
output 输出图像集合
boxes 需要缩放的 box 的坐标及宽高的集合
padtype 边界像素填充方式
padval 边界像素填充值(BGR)

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	NV12	宽高为 2 的倍数	2x2 ~ 1024x2048
output	NV12、BGRA	宽高为 2 的倍数	2x2 ~ 1024x2048
boxes	-	见注意事项	2x2 ~ 1024x2048

- 返回值

0 表示成功，否则失败

- 注意事项
  - 当抠的 ROI 图像和输出图像的宽高比不一致时，可以通过 padtype 选择输出直接缩放图像、四周填充的等比例缩放图像、右下填充的等比例缩放图像。
  - boxes 中每个 box 的左上右下坐标必须为偶数（能被 2 整除），即 Bbox_t 数据结构中的 x0,y0,x1,y1 均必须为偶数。

- **common_resize_nv12**
  - 功能描述

    对输入图像（NV12）执行缩放处理，返回缩放后的图像数据(BGRA/NV12)。
  - 接口定义

    *common_resize_nv12(const void *input, Tensor &output, int img_h, int img_w, int line_stride, bool cvtbgra, PaddingType padtype, uint8_t padval, AddressLocate input_locate)*
  - 参数列表

    input　　输入数据指针
    output　　输出图像
    img_h　　输入图像高
    img_w　　输入图像宽
    line_stride　　输入图像行间距
    cvtbgra　　是否将输出图像转成 BGRA
    padtype　　边界像素填充方式
    input_locate　输入图像地址属性
    padval　　边界像素填充值(BGR)
  - 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	NV12	宽高为 2 的倍数	2x2 ~ 1024x2048
output	NV12、BGRA	宽高为 2 的倍数	2x2 ~ 1024x2048

  - 返回值

    0 表示成功，否则失败
  - 注意事项
    - 当输入输出图像的宽高比不一致时，可以通过 padtype 选择输出直接缩放图像、四周填充的等比例缩放图像、右下填充的等比例缩放图像。
    - 输出为 NV12 格式时，输出宽高要求为 2 的倍数；输出为 BGRA 格式时，输出宽高无对齐要求。
    - 输入图像的内存必须来自物理地址连续的内存空间（rmem/nmem）。当 input_locate 为 NMEM_VIRTUAL 时，input 必须为 nmem 上的虚拟地址（Tensor 的 data/mudata 接口获取的指针，或者通过 nmem_memalign 申请的内存指针）；当 input_locate 为 RMEM_PHYSICAL 时，input 必须为 rmen 上的物理地址（通过 isvp 平台的相关接口获取的视频流物理地址）。

- **wrap_affine_bgra**
  - 功能描述

    根据仿射变换矩阵对输入图像(BGRA)执行仿射变换操作
  - 接口定义

    int wrap_affine_bgra(const Tensor &input, Tensor &output, Tensor &matrix)
  - 参数列表

    input　　输入图像

output 输出图像
matrix 仿射变换矩阵

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA	宽为 2 的倍数	2x2 ~ 1024x2048
output	BGRA	-	2x2 ~ 1024x2048

- 返回值

0 表示成功，否则失败

- 注意事项
  - 超出边界的默认填充 0

■ **wrap_affine_nv12**

- 功能描述

根据仿射变换矩阵对输入图像(NV12)执行仿射变换操作

- 接口定义

int wrap_affine_nv12(const Tensor &input, Tensor &output, Tensor &matrix, bool cvtbgra)

- 参数列表

input 输入图像
output 输出图像
matrix 仿射变换矩阵
cvtbgra 是否输出 BGRA 图像

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	NV12	宽高为 2 的倍数	2x2 ~ 1024x2048
output	NV12、BGRA	宽高为 2 的倍数	2x2 ~ 1024x2048

- 返回值

0 表示成功，否则失败

- 注意事项
  - 超出边界的默认填充成全黑

■ **crop_affine_bgra**

- 功能描述

根据输入的 boxes 在原图（BGRA）上执行抠图操作，然后对抠完的图根据各自的仿射变换矩阵执行仿射变换

- 接口定义

int crop_affine_bgra(const Tensor &input, std::vector<Tensor> &output, std::vector<Tensor> &matrixes, std::vector<Bbox_t>&boxes)

- 参数列表

input 输入图像
output 输出图像的集合

matrixes　　仿射变换矩阵的集合
boxes　　需要仿射变换的 box 的坐标及宽高的集合

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA	宽为 2 的倍数	2x2 ~ 1024x2048
output	BGRA	-	2x2 ~ 1024x2048
boxes	-	见注意事项	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项
  - boxes 中每个 box 的宽必须为 2 的倍数，即 Bbox_t 的(x1-x0)为 2 的倍数。

■ **crop_affine_nv12**

- 功能描述

  根据输入的 boxes 在原图（NV12）上执行抠图操作，然后对抠完的图根据各自的仿射变换矩阵执行仿射变换

- 接口定义

  int crop_affine_nv12(const Tensor &input,std::vector<Tensor> &output, std::vector<Tensor> &matrixes, bool cvtbgra, std::vector<Bbox_t>&boxes)

- 参数列表

  input　　输入图像
  output　　输出图像集合
  matrixes　　仿射变换矩阵的集合
  boxes　　需要仿射变换的 box 的坐标及宽高的集合
  cvtbgra　　是否输出 BGRA 格式的图像

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	NV12	宽高为 2 的倍数	2x2 ~ 1024x2048
output	NV12、BGRA	宽高为 2 的倍数	2x2 ~ 1024x2048
boxes	-	见注意事项	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项
  - boxes 中每个 box 的左上右下坐标必须为偶数（能被 2 整除），即 Bbox_t 数据结构中的 x0,y0,x1,y1 均必须为偶数。

■ **wrap_perspective_bgra**

- 功能描述

  根据透视变换矩阵对输入图像(BGRA)执行透视变换操作

- 接口定义

  int wrap_perspective_bgra(const Tensor &input, Tensor &output, Tensor

&matrix)

- 参数列表

  input　　输入图像
  output　　输出图像
  matrix　　透视变换矩阵

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA	宽为 2 的倍数	2x2 ~ 1024x2048
output	BGRA	-	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项
  - 超出边界的默认填充 0

■ **wrap_perspective_nv12**

- 功能描述

  根据透视变换矩阵对输入图像(NV12)执行透视变换操作

- 接口定义

  int wrap_perspective_bgra(const Tensor &input, Tensor &output, Tensor &matrix, bool cvtbgra)

- 参数列表

  input　　输入图像
  output　　输出图像
  matrix　　透视变换矩阵
  cvtbgra　　是否输出 BGRA 格式的图像

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	NV12	宽高为 2 的倍数	2x2 ~ 1024x2048
output	NV12、BGRA	宽高为 2 的倍数	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项
  - 超出边界的默认填充成全黑

■ **crop_perspective_bgra**

- 功能描述

  根据输入的 boxes 在原图（BGRA）上执行抠图操作，然后对抠完的图根据各自的透视变换矩阵执行透视变换

- 接口定义

  int crop_perspective_bgra(const Tensor &input, std::vector<Tensor> &output, std::vector<Tensor> &matrixes, std::vector<Bbox_t>&boxes)

- 参数列表

  input　　输入图像
  output　　输出图像集合
  matrixes　　透视变换矩阵集合
  boxes　　需要透视变换的 box 的坐标及宽高的集合

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	BGRA	宽为 2 的倍数	2x2 ~ 1024x2048
output	BGRA	-	2x2 ~ 1024x2048
boxes	-	见注意事项	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项

  boxes 中每个 box 的宽必须为 2 的倍数，即 Bbox_t 的(x1-x0)为 2 的倍数。

■ **crop_perspective_nv12**

- 功能描述

  根据输入的 boxes 在原图（NV12）上执行抠图操作，然后对抠完的图根据各自的透视变换矩阵执行透视变换

- 接口定义

  int crop_perspective_nv12(const Tensor &input, std::vector<Tensor> &output, std::vector<Tensor> &matrixes, bool cvtbgra, std::vector<Bbox_t>&boxes)

- 参数列表

  input　　输入图像
  output　　输出图像集合
  matrixes　　透视变换矩阵集合
  boxes　　需要透视变换的 box 的坐标及宽高的集合
  cvtbgra　　是否输出 BGRA 格式的图像

- 参数规格

参数名称	支持图像类型	对齐要求	分辨率
input	NV12	宽高为 2 的倍数	2x2 ~ 1024x2048
output	NV12、BGRA	宽高为 2 的倍数	2x2 ~ 1024x2048
boxes	-	见注意事项	2x2 ~ 1024x2048

- 返回值

  0 表示成功，否则失败

- 注意事项

  - boxes 中每个 box 的左上右下坐标必须为偶数（能被 2 整除），即 Bbox_t 数据结构中的 x0,y0,x1,y1 均必须为偶数。

● 网络后处理类

- **nms**
  - 功能描述
    对目标检测网络输出的 Bbox 做非极大值抑制。
  - 接口定义
    void nms(std::vector<ObjBbox_t> &input, std::vector<ObjBbox_t> &output, float nms_threshold = 0.3, NmsType type = NmsType::HARD_NMS)
  - 参数列表

input	输入坐标框数组
output	输出坐标框数组
nms_threshold	NMS 的阈值
type	NMS 的类型（HARD_NMS、SOFT_NMS）

  - 返回值
    无
- **generate_box**
  - 功能描述
    根据 anchor 生成候选框
  - 接口定义
    void generate_box(std::vector<Tensor> &features, std::vector<float> &strides, std::vector<float> &anchor, std::vector<ObjBbox_t> &candidate_boxes, int img_w, int img_h, int classes, int box_num,float box_score_threshold, DetectorType detector_type = DetectorType::YOLOV3)
  - 参数列表

features	输入的特征图
strides	输入特征图对应的下采样倍数
anchor	预先设定的 anchor
candidate_boxes	生成的候选框
img_w	网络输入图像的宽
img_h	网络输入图像的高
classes	目标检测的类别个数
box_num	每个锚点对应的 box 数目
box_score_threshold	过滤多余框的得分阈值
detector_type	检测器的类型（YOLOV3 系列或者 YOLOV5 系列）

  - 返回值
    无

- 内存管理类
- **nmem_malloc**
  - 功能描述
    根据输入的字节数在 nmem 上申请一块内存空间，返回对应的首地址指针
  - 接口定义
    *void *nmem_malloc(unsigned int size)*
  - 参数列表

size 需要申请的内存字节数

- 返回值

  NULL 表示申请失败，否则返回申请到的内存块对应的首地址指针

- **nmem_memalign**
  - 功能描述

    根据输入的字节数在 nmem 上申请一块内存空间，并将首地址按照指定的字节数对齐，然后返回对应的首地址指针
  - 接口定义

    *void *nmem_memalign(unsigned int align,unsigned int size)*
  - 参数列表

    align 内存块的首地址按照 align 个字节对齐

    size 需要申请的内存字节数
  - 返回值

    NULL 表示申请失败，否则返回申请到的内存块对应的首地址指针
- **nmem_free**
  - 功能描述

    释放输入指针指向的 nmem 内存
  - 接口定义

    *void nmem_free(void *ptr)*
  - 参数列表

    ptr 指向 nmem 内存块的指针
  - 返回值

    无
- **memcopy**
  - 功能描述

    将源地址中的一块数据拷贝到目标地址中
  - 接口定义

    *void memcopy(void *dst, void *src, int n)*
  - 参数列表

    dst 目标地址

    src 源地址

    n 需要拷贝的字节数
  - 返回值

    无

完整示例：

获取 Magik 软件包后，具体参见目录：

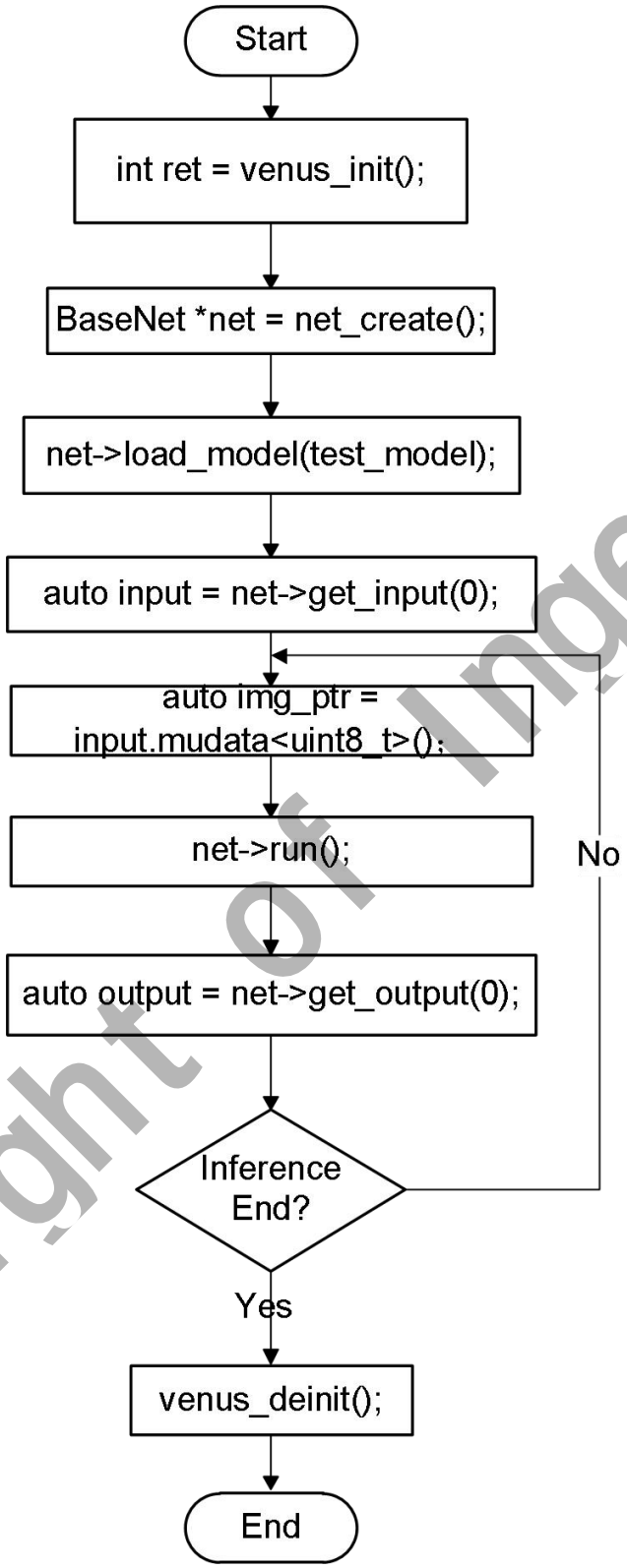
your_root/InferenceKit/nna1/mips720-glibc229/sample/imgproc_run/

2.4 Inference 流程

为了方便您的使用，Venus 进行了良好的 API 设计，隐藏了大量您不需要投入时间研究的细节。您只需要简单的七步即可使用 Venus 在 T40 和 T02 芯片平台上完成模型前向推理。

1. 系统初始化。调用 venus_init，完成对系统环境的初始化。
2. 创建 BaseNet 句柄。通过调用 net_create 接口创建推理引擎的句柄。
3. 加载模型。通过调用 BaseNet 句柄的方法 load_model 加载需要推理的模型。
4. 准备输入数据。通过调用 BaseNet 句柄的方法 get_input 获取模型的输入节点 Tensor，然后将图像数据拷贝的 tensor 中即可。
5. 执行推理。并通过调用 BaseNet 句柄的方法 run 执行模型推理。
6. 获取结果。通过调用 BaseNet 句柄的方法 get_output 获取模型的输出节点 Tensor 即可。
7. 调用 venus_deinit，释放系统资源。

具体的推理流程如图 2-1 所示。



图中**Inference**例子配置信息如下：

1、test_model：表示模型文件的文件名

2、模型只有一个输入节点和一个输出节点

图 2-1 Venus Inference 流程图

2.4.1 算子集合

算子类型	推理位宽	备注
Convolution	8bit/4bit/2bit	
DepthWiseConvolution	8bit/4bit/2bit	
BatchNorm	8bit/4bit/2bit	推理时融合到卷积中
BiasAdd	8bit/4bit/2bit	推理时融合到卷积中
FullConnected	8bit/4bit/2bit	
Pooling	12bit/10bit/8bit/4bit/2bit	MaxPool/AvgPool/GlobalAvgPool
UpSample	Float/12bit/10bit/8bit/4bit/2bit	
Relu	8bit/4bit/2bit	推理时融合到卷积中
Relu6	8bit/4bit/2bit	推理时融合到卷积中
PRelu	8bit/4bit/2bit	推理时融合到卷积中
LeakyRelu	8bit/4bit/2bit	推理时融合到卷积中
Silu	8bit/4bit/2bit	推理时融合到卷积中
Tanh	8bit/4bit/2bit	推理时融合到卷积中
Sigmoid	8bit/4bit/2bit	推理时融合到卷积中
Swish	8bit	推理时融合到卷积中
Lstm	8bit/4bit	
Concat	8bit/10bit/12bit	
Softmax	Float	
Mul	Float/12bit/10bit/8bit/4bit/2bit	
Add	12bit/10bit/8bit/4bit/2bit	
Eltwise	12bit/10bit/8bit	Min/Max/Sub
Sum	Float	
Permute	8bit	
Slice	8bit/4bit/2bit	
Padding	8bit	
Normalize	8bit/10bit/12bit	

表 2-1 Venus 算子列表

2.4.1.1 Convolution 功能说明

输入位宽 / 参数支持	2bit	4bit	8bit
权重位宽	2bit/4bit	2bit/4bit/6bit/8bit	2bit/4bit/6bit/8bit
输出位宽	2bit/4bit/8bit/float	2bit/4bit/8bit/float	4bit/8bit/float
核尺寸	任意	任意	任意
步长	<=2	<=2	<=2
空洞卷积	支持	支持	支持
BATCH	不支持	支持	支持
激活函数	relu/relu6	relu/relu6/prelu/leaky_relu/tanh/silu/sigmoid/swish	relu/relu6/prelu/leaky_relu/tanh/silu/sigmoid/swish

表 2-2 Convolution 功能说明

2.4.1.2 DepthWiseConvolution 功能说明

输入位宽 / 参数支持	2bit	4bit	8bit
权重位宽	2bit/4bit	2bit/4bit	2bit/4bit/6bit/8bit
输出位宽	2bit/4bit	2bit/4bit	8bit
核尺寸	任意	任意	任意
步长	任意	任意	<=2
BATCH	支持	支持	支持
激活函数	relu/relu6	relu/relu6	relu/relu6/prelu/leaky_relu/tanh/silu/sigmoid/swish

表 2-3 DepthWiseConvolution 功能说明

2.4.1.3 FullConnected 功能说明

输入位宽 / 参数支持	2bit	4bit	8bit
权重位宽	2bit/4bit/8bit	2bit/4bit/8bit	2bit/4bit/8bit

输出位宽	2bit/4bit/float	2bit/4bit/8bit/float	8bit/float
BATCH	不支持	不支持	支持
激活函数	输出位宽为float时支持relu/relu6/prelu/leaky_relu/tanh/silu/sigmoid/swish，其他位宽时只支持relu/relu6	输出位宽为float时支持relu/relu6/prelu/leaky_relu/tanh/silu/sigmoid/swish，其他位宽时只支持relu/relu6	relu/relu6/prelu/leaky_relu/tanh/silu/sigmoid/swish

表 2-4 FullConnected 功能说明

2.4.1.4 Add 功能说明

输入位宽 / 参数支持	2bit	4bit	8bit	10bit/12bit
BATCH	支持	支持	支持	支持
激活函数	无	RELU/NONE/LINEAR/RELU6/HARD_SIGMOID TANH/SWISH/SIGMOID/SILU/LEAKY_RELU	NONE/RELU/RELU6/SWISH/TANH/SIGMOID	NONE/RELU/RELU6/SWISH/TANH/SIGMOID

表 2-5 Add 功能说明

2.4.1.5 Mul 功能说明

输入位宽 / 参数支持	2bit	4bit	8bit/10bit/12bit	Fp32
BATCH	支持	支持	支持	支持
广播类型	SINGLE/CHANNEL/ELEMET/SPATIAL	SINGLE/CHANNEL/ELEMET/SPATIAL	SINGLE/CHANNEL/ELEMET/SPATIAL	CHANNEL

表 2-6 Mul 功能说明

2.4.1.6 Pooling 功能说明

输入位宽 / 参数支持	2bit	4bit	8bit	10bit/12bit
BATCH	支持	支持	支持	支持
PoolType	MAX_POOL/AVG_POOL/GLOBAL_AVG_POOL	MAX_POOL/AVG_POOL/GLOBAL_AVG_POOL	MAX_POOL/AVG_POOL/GLOBAL_AVG_POOL	MAX_POOL/AVG_POOL/GLOBAL_AVG_POOL

表 2-7 Pooling 功能说明

2.4.1.7 Upsample 功能说明

输入位宽 / 参数支持	2bit	4bit	8bit	10bit/12bit	Fp32
BATCH	支持	支持	支持	支持	支持
UppType	FILL_ZERO/ NEARST/ PIXEEL_SHUFFLE/ BILINEAR	FILL_ZERO/ NEARST/ PIXEEL_SHUFFLE	FILL_ZERO/ NEARST/ PIXEEL_SHUFFLE	NEARST	FILL_ZERO/ NEARST

表 2-8 Upsample 功能说明

2.4.1.8 Concat 功能说明

输入位宽 / 参数支持	8bit	10bit	12bit
BATCH	支持	支持	支持

表 2-9 Concat 功能说明

2.4.1.9 Normalize 功能说明

输入位宽 / 参数支持	8bit	10bit	12bit
BATCH	不支持	不支持	不支持

表 2-10 Normalize 功能说明

2.4.1.10 LSTM 功能说明

输入位宽 / 参数支持	4bit	8bit
权重位宽	4bit	2bit/4bit/6bit/8bit
输出位宽	4bit	8bit
BATCH	不支持	不支持

表 2-11 LSTM 功能说明

2.4.1.11 Eltwise 功能说明

输入位宽 / 参数支持	8bit	10bit	12bit
BATCH	不支持	不支持	不支持

EltwiseType	Sub/Max/Min	Sub/Max/Min	Sub/Max/Min

表 2-12 Eltwise 功能说明

2.4.1.12 Sum 功能说明

输入位宽 参数支持	float
BATCH	支持
AXIS	2（按 width 方向）

表 2-13 Sum 功能说明

2.4.1.13 Softmax 功能说明

输入位宽 参数支持	float
BATCH	不支持
AXIS	2(按 width 方向)/ 3(按 channel 方向)

表 2-14 Softmax 功能说明

2.4.1.14 Embedding 功能说明

输入位宽 参数支持	int32
feature 位宽	8bit
输出位宽	8bit
BATCH	不支持

表 2-15 Embedding 功能说明

2.4.1.15 Padding 功能说明

输入位宽 参数支持	8bit
BATCH	支持

表 2-16 Padding 功能说明

2.4.1.16 Slice 功能说明

输入位宽 参数支持	2bit	4bit	8bit
BATCH	支持	支持	支持

表 2-17 Slice 功能说明

2.4.1.17 Permute 功能说明

输入位宽 参数支持	8bit
BATCH	支持

表 2-18 Permute 功能说明

2.4.2 Profiler 工具

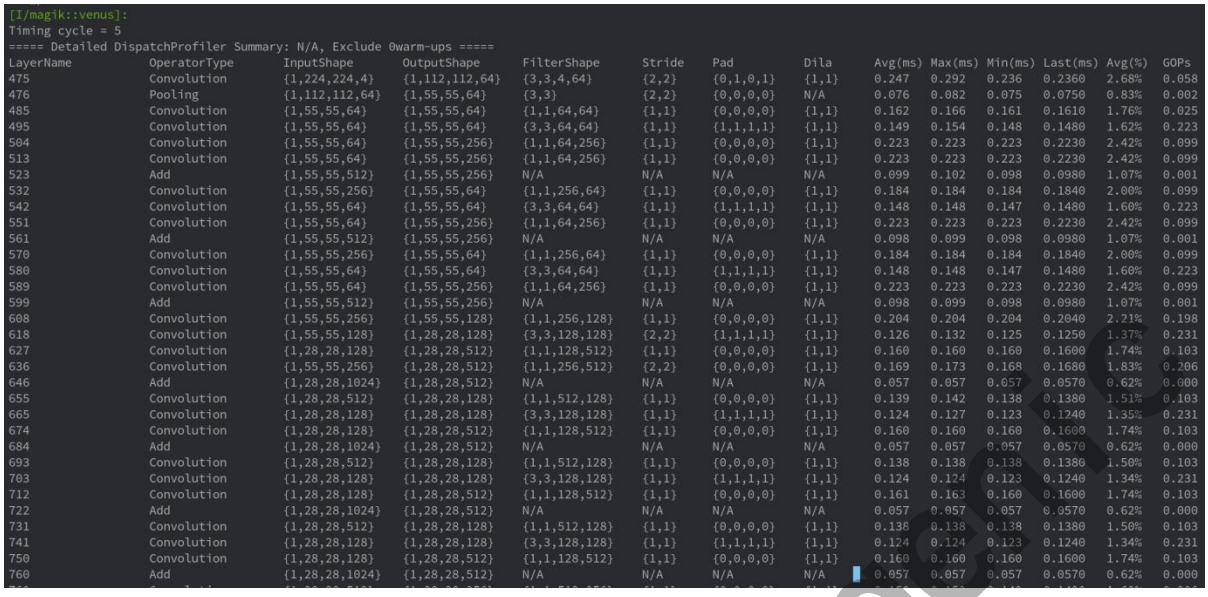
Profiler 用于统计分析网络每层的详细运行时间和每种类型算子的运行时间及调用次数等相关信息，便于用户更好的优化设计网络模型结构。其中网络每层的统计结果主要包含层名称、算子类型、输入 Feature 尺寸、输出 Feature 尺寸、权重尺寸、Stride 信息、Pad 信息、卷积膨胀系数、当前层运行时间、当前层运行时间占比、当前层 GOPs 和当前层的带宽等信息；Profiler 整体的总结结果包括算子类型、每种算子的运行时间、时间占比、GOPs、带宽和调用次数等信息。

2.4.2.1 开启方法

在编译链接基于 Venus 可执行程序时，链接 libvenus.p.so 即可开启 Profiler 工具进行网络性能分析。

2.4.2.2 使用示例

在 main 函数中实例化需要分析的模型句柄，然后循环多次调用 run 接口，在模型执行完毕后将自动打印类似下图的 profiler 日志信息，其中图 2-2 是网络每层的详细统计信息，图 2-3 是模型整体的统计总结信息。



```
[I/magik::venus]:
Timing cycle = 5
===== Detailed DispatchProfiler Summary: N/A, Exclude 0warm-ups =====
LayerName          OperatorType       InputShape       OutputShape      FilterShape      Stride    Pad          Dila     Avg(ms) Max(ms) Min(ms) Last(ms) Avg(%)  GOPs
475                Convolution        {1,224,224,4}    {1,112,112,64}   {3,3,4,64}       {2,2}     {0,1,0,1}    {1,1}    0.247   0.292   0.236   0.2360   2.68%   0.058
476                Pooling            {1,112,112,64}   {1,55,55,64}     {3,3}            {2,2}     {0,0,0,0}    N/A      0.076   0.082   0.075   0.0750   0.83%   0.002
485                Convolution        {1,55,55,64}     {1,55,55,64}     {1,1,64,64}      {1,1}     {0,0,0,0}    {1,1}    0.162   0.166   0.161   0.1610   1.76%   0.025
495                Convolution        {1,55,55,64}     {1,55,55,64}     {3,3,64,64}      {1,1}     {1,1,1,1}    {1,1}    0.149   0.154   0.148   0.1480   1.62%   0.223
504                Convolution        {1,55,55,64}     {1,55,55,256}    {1,1,64,256}     {1,1}     {0,0,0,0}    {1,1}    0.223   0.223   0.223   0.2230   2.42%   0.099
513                Convolution        {1,55,55,64}     {1,55,55,256}    {1,1,64,256}     {1,1}     {0,0,0,0}    {1,1}    0.223   0.223   0.223   0.2230   2.42%   0.099
523                Add                {1,55,55,512}    {1,55,55,256}    N/A              N/A       N/A          N/A      0.099   0.102   0.098   0.0980   1.07%   0.001
532                Convolution        {1,55,55,256}    {1,55,55,64}     {1,1,256,64}     {1,1}     {0,0,0,0}    {1,1}    0.184   0.184   0.184   0.1840   2.00%   0.099
542                Convolution        {1,55,55,64}     {1,55,55,64}     {3,3,64,64}      {1,1}     {1,1,1,1}    {1,1}    0.148   0.148   0.147   0.1480   1.60%   0.223
551                Convolution        {1,55,55,64}     {1,55,55,256}    {1,1,64,256}     {1,1}     {0,0,0,0}    {1,1}    0.223   0.223   0.223   0.2230   2.42%   0.099
561                Add                {1,55,55,512}    {1,55,55,256}    N/A              N/A       N/A          N/A      0.098   0.099   0.098   0.0980   1.07%   0.001
570                Convolution        {1,55,55,256}    {1,55,55,64}     {1,1,256,64}     {1,1}     {0,0,0,0}    {1,1}    0.184   0.184   0.184   0.1840   2.00%   0.099
580                Convolution        {1,55,55,64}     {1,55,55,64}     {3,3,64,64}      {1,1}     {1,1,1,1}    {1,1}    0.148   0.148   0.147   0.1480   1.60%   0.223
589                Convolution        {1,55,55,64}     {1,55,55,256}    {1,1,64,256}     {1,1}     {0,0,0,0}    {1,1}    0.223   0.223   0.223   0.2230   2.42%   0.099
599                Add                {1,55,55,512}    {1,55,55,256}    N/A              N/A       N/A          N/A      0.098   0.099   0.098   0.0980   1.07%   0.001
608                Convolution        {1,55,55,256}    {1,55,55,128}    {1,1,256,128}    {1,1}     {0,0,0,0}    {1,1}    0.204   0.204   0.204   0.2040   2.21%   0.198
618                Convolution        {1,55,55,128}    {1,28,28,128}    {3,3,128,128}    {2,2}     {1,1,1,1}    {1,1}    0.126   0.132   0.125   0.1250   1.37%   0.231
627                Convolution        {1,28,28,128}    {1,28,28,512}    {1,1,128,512}    {1,1}     {0,0,0,0}    {1,1}    0.160   0.160   0.160   0.1600   1.74%   0.103
636                Convolution        {1,55,55,256}    {1,28,28,512}    {1,1,256,512}    {2,2}     {0,0,0,0}    {1,1}    0.169   0.173   0.168   0.1680   1.83%   0.206
646                Add                {1,28,28,1024}   {1,28,28,512}    N/A              N/A       N/A          N/A      0.057   0.057   0.057   0.0570   0.62%   0.000
655                Convolution        {1,28,28,512}    {1,28,28,128}    {1,1,512,128}    {1,1}     {0,0,0,0}    {1,1}    0.139   0.142   0.138   0.1380   1.51%   0.103
665                Convolution        {1,28,28,128}    {1,28,28,128}    {3,3,128,128}    {1,1}     {1,1,1,1}    {1,1}    0.124   0.127   0.123   0.1240   1.35%   0.231
674                Convolution        {1,28,28,128}    {1,28,28,512}    {1,1,128,512}    {1,1}     {0,0,0,0}    {1,1}    0.160   0.160   0.160   0.1600   1.74%   0.103
684                Add                {1,28,28,1024}   {1,28,28,512}    N/A              N/A       N/A          N/A      0.057   0.057   0.057   0.0570   0.62%   0.000
693                Convolution        {1,28,28,512}    {1,28,28,128}    {1,1,512,128}    {1,1}     {0,0,0,0}    {1,1}    0.138   0.138   0.138   0.1380   1.50%   0.103
703                Convolution        {1,28,28,128}    {1,28,28,128}    {3,3,128,128}    {1,1}     {1,1,1,1}    {1,1}    0.124   0.124   0.123   0.1240   1.34%   0.231
712                Convolution        {1,28,28,128}    {1,28,28,512}    {1,1,128,512}    {1,1}     {0,0,0,0}    {1,1}    0.161   0.163   0.160   0.1600   1.74%   0.103
722                Add                {1,28,28,1024}   {1,28,28,512}    N/A              N/A       N/A          N/A      0.057   0.057   0.057   0.0570   0.62%   0.000
731                Convolution        {1,28,28,512}    {1,28,28,128}    {1,1,512,128}    {1,1}     {0,0,0,0}    {1,1}    0.138   0.138   0.138   0.1380   1.50%   0.103
741                Convolution        {1,28,28,128}    {1,28,28,128}    {3,3,128,128}    {1,1}     {1,1,1,1}    {1,1}    0.124   0.124   0.123   0.1240   1.34%   0.231
750                Convolution        {1,28,28,128}    {1,28,28,512}    {1,1,128,512}    {1,1}     {0,0,0,0}    {1,1}    0.160   0.160   0.160   0.1600   1.74%   0.103
760                Add                {1,28,28,1024}   {1,28,28,512}    N/A              N/A       N/A          N/A      0.057   0.057   0.057   0.0570   0.62%   0.000
```

图 2-2 Profiler 每层统计结果

```
[I/magik::venus]:
Timing cycle = 5
===== Concise Dispatch Profiler Summary: N/A, Exclude 0 warm-ups =====
OperatorType          Avg(ms)  Max(ms)  Min(ms)  Avg(%)   GOPs     CalledTimes
Add                   0.733    0.738    0.732    7.95%    0.005    16
Convolution           8.392    8.493    8.358    91.03%    7.941    53
Pooling               0.094    0.103    0.092    1.02%    0.002    2
```

图 2-3 Profiler 整体统计结果

2.4.3 完整示例

获取 Magik 软件包后，具体参见 your_root/Models/post/soc_sample/T40/venus_sample_ptq_yolov5s/目录。