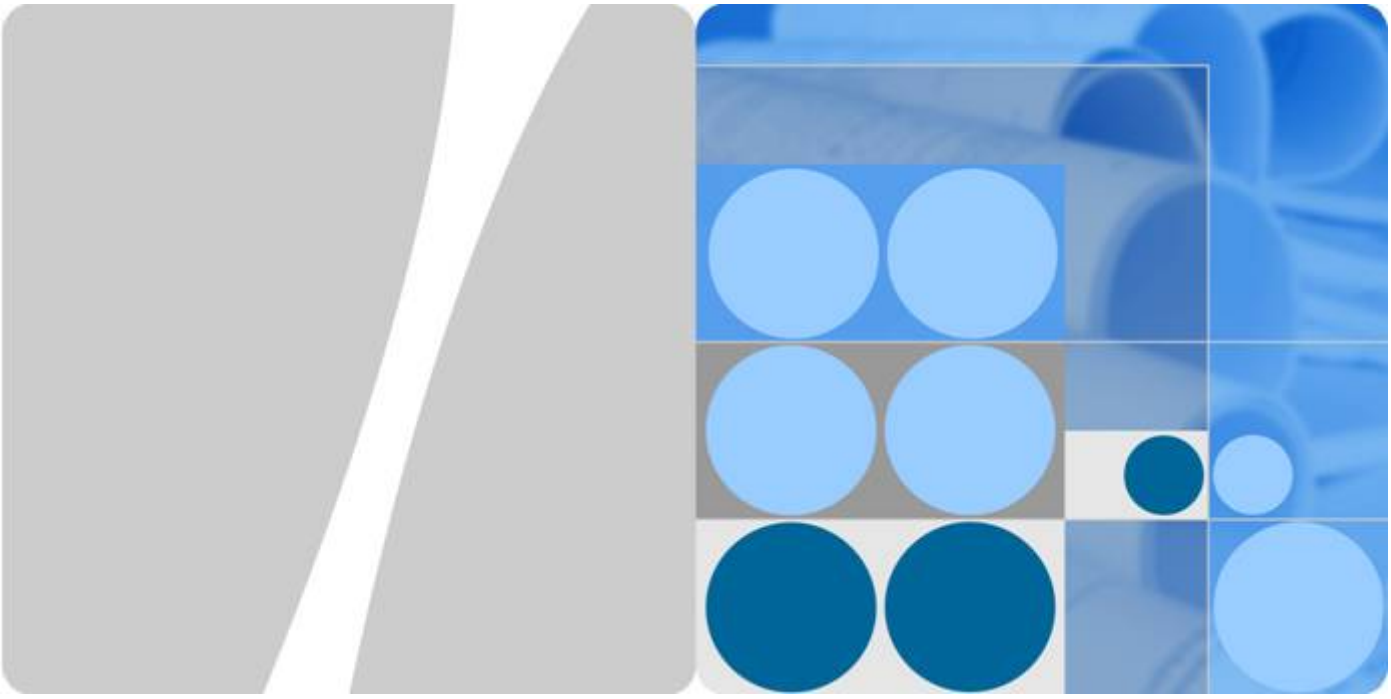

Ascend 310

V100R001

网络模型配置参考

文档版本　01

发布日期　2019-03-12

华为技术有限公司

HUAWEI

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： http://www.huawei.com

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

目 录

1 简介

网络模型指的是利用神经网络进行深度学习的算法合集，通过数据训练可以实现如图像分类，物体检测等功能。

表1-1为目前支持的网络模型。

表 1-1 网络模型视图

网络模型		说明	运行环境
分类网络 (classify_net)	ResNet-18	请参考 **2.1 ResNet-18**	支持EVB环境，PCIe，开发板等环境下运行。
	ResNet-50	请参考 **2.2 ResNet-50**	支持EVB环境，PCIe，开发板等环境下运行。
	ResNet-101	请参考 **2.3 ResNet-101**	支持EVB环境，PCIe，开发板等环境下运行。
	ResNet-152	请参考 **2.4 ResNet-152**	支持EVB环境，PCIe，开发板等环境下运行。
	ResNext-50	请参考 **2.5 ResNext-50**	支持EVB环境，PCIe，开发板等环境下运行。
	ResNext-101	请参考 **2.6 ResNext-101**	支持EVB环境，PCIe，开发板等环境下运行。
	VGG16	请参考 **2.7 VGG16**	支持EVB环境，PCIe，开发板等环境下运行。
	VGG19	请参考 **2.8 VGG19**	支持EVB环境，PCIe，开发板等环境下运行。
检测网络 (detection_net)	Faster-RCNN(VGG-16)	请参考 **2.10 Faster-RCNN**	支持EVB环境，PCIe，开发板等环境下运行。
	SSD	请参考 **2.9 SSD**	支持EVB环境，PCIe，开发板等环境下运行。

- ResNet

  ResNet（Residual Network）是2015年ImageNet图像分类、图像物体定位和图像物体检测比赛的冠军。针对训练卷积神经网络时加深网络导致准确度下降的问题，ResNet提出了采用残差学习的方法。在已有设计思路（BN，小卷积核，全卷积网络）的基础上，引入了残差模块。每个残差模块包含两条路径，其中一条路径是输入特征的直连通路，另一条路径对该特征做两到三次卷积操作得到该特征的残差，最后再将两条路径上的特征相加。

  表1-1中的ResNet-18、ResNet-50、ResNet-101、ResNet-152分别指的是残差网络的层数，层数越高，训练误差越小。

- ResNext

  ResNext网络是ResNet的升级版， ResNext结构可以在不增加参数复杂度的前提下提高准确率，同时还减少了超参数的数量。ResNext同时采用VGG堆叠的思想和Inception的split-transform-merge 思想，但是可扩展性比较强，可以认为是在增加准确率的同时基本不改变或降低模型的复杂度。这里提到一个名词cardinality，原文的解释是the size of the set of transformations，如图1-1右边是 cardinality=32 的例子。

  上表中提到的ResNext-50，ResNext-101，分别指的是ResNext的层数，层数越高，训练误差越小。

图 1-1 ResNext 结构原理图

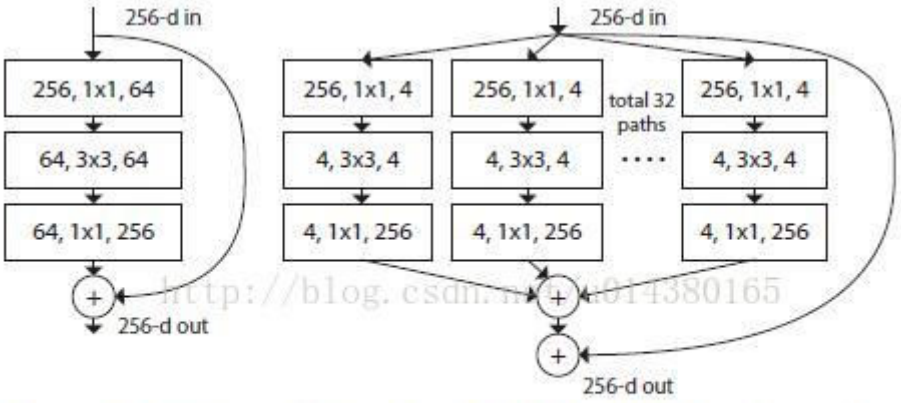

Figure 1. **Left**: A block of ResNet [14]. **Right**: A block of ResNeXt with cardinality = 32, with roughly the same complexity. A layer is shown as (# in channels, filter size, # out channels).

说明

图1-1中每个被聚合的拓扑结构都是一样的。

- VGG

  牛津大学VGG（Visual Geometry Group）组在2014年ILSVRC提出的模型被称作VGG模型。该模型相比以往模型进一步加宽和加深了网络结构，它的核心是五组卷积操作，每两组之间做Max-Pooling空间降维。同一组内采用多次连续的3X3卷积，卷积核的数目由较浅组的64增多到最深组的512，同一组内的卷积核数目是一样的。卷积之后接两层全连接层，之后是分类层。由于每组内卷积层的不同，有11、13、16、19层这几种模型，图1-2展示一个16层的网络结构。VGG模型结构相对简洁，提出之后也有很多文章基于此模型进行研究，如在ImageNet上首次公开超过人眼识别的模型就是借鉴VGG模型的结构。

上表中提到的VGG16，VGG19指的是VGG模型里使用的层数是16层或19层，层数越高，准确度越高，内存消耗越大。

图 1-2 16 层网络模型结构

2 配置参考

2.1 ResNet-18

ResNet-18即18-layers的残差网络。

使用场景

通过ResNet-18分类网络模型对图像进行分类。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
ResNet-18 model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。 **说明** Graph配置文件样例请参见《Ascend 310 HiAI Engine样例（Emulator）》中的2.4章节。

出参

参数	说明
分类结果置信度	无

调用示例

结果解析请参见**2.11.1 分类网络结果解析**。

2.2 ResNet-50

ResNet-50即50-layers的残差网络。

使用场景

通过ResNet-50 分类网络模型对图像进行分类。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
ResNet-50 model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。

出参

参数	说明
分类结果置信度	无。

调用示例

结果解析请参见2.11.1 分类网络结果解析。

2.3 ResNet-101

ResNet-101即101-layers的残差网络。

使用场景

通过ResNet-101分类网络模型对图像进行分类。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
ResNet-101 model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。

出参

参数	说明
分类结果置信度	无。

调用示例

结果解析请参见**2.11.1 分类网络结果解析**。

2.4 ResNet-152

ResNet-152即152-layers的残差网络。

使用场景

通过ResNet-152 分类网络模型对图像进行分类。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
ResNet-152 model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。

出参

参数	说明
分类结果置信度	无。

调用示例

结果解析请参见**2.11.1 分类网络结果解析**。

2.5 ResNext-50

ResNext-50即50-layers的ResNext网络。

使用场景

通过ResNext-50模型对图像进行分类。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
ResNext-50-model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。

出参

参数	说明
分类结果置信度	无。

调用示例

结果解析请参见2.11.1 分类网络结果解析。

2.6 ResNext-101

ResNext-101即101-layers的ResNext网络。

使用场景

通过ResNext-101模型对图像进行分类。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
ResNext-101-model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。

出参

参数	说明
分类结果置信度	无

调用示例

结果解析请参见2.11.1 分类网络结果解析。

2.7 VGG16

VGG16即16层的VGG模型。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
VGG16 model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。

出参

参数	说明
分类结果置信度	无。

调用示例

结果解析请参见**2.11.1 分类网络结果解析**。

2.8 VGG19

VGG19即19层的VGG模型。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
VGG19 model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。

出参

参数	说明
分类结果置信度	无。

调用示例

结果解析请参见**2.11.1 分类网络结果解析**。

2.9 SSD

SSD，即 Single-Shot Detector，它的速度比 Faster-RCNN 要快很多，但其工作方式却和 R-FCN （Region Full Convolutional Network，全卷积网络，特点是在分类时用卷积层代替全连接层，用于分类）存在显著不同。

给定一个输入图像以及一系列真值标签，SSD 就会进行如下操作：

1. 在一系列卷积层中传递这个图像，产生一系列大小不同的特征图（比如 10x10、6x6、3x3 等等。）
2. 对个这些特征图中的每个位置而言，都使用一个3x3 的卷积滤波器（convolutional filter）来评估一小部分默认的边界框。这些默认的边界框本质上等价于 Faster-RCNN 的 anchor box。
3. 对每个边界框都同时执行预测： a）边界框的偏移；b）分类的概率。
4. 在训练期间，用这些基于 IoU（Intersection over Union，也被称为 Jaccard 相似系数，值为0-1，0为不重合，1为完全重合）系数的预测边界框来匹配正确的边界框。被最佳预测的边界框将被标签为「正」。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
ssd model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。

出参

参数	说明
物体置信度， 框的坐标	无

调用示例

结果解析请参见**2.11.2 检测网络结果解析（SSD）**。

2.10 Faster-RCNN

早期，使用窗口扫描进行物体识别，计算量大。 RCNN去掉窗口扫描，用聚类方式，对图像进行分割分组，得到多个侯选框的层次组。RCNN中有CNN重复计算，Fast RCNN则去掉重复计算，并微调选框位置。

经过RCNN和Fast RCNN 的积淀， Ross B. Girshick等人在2015年提出了新的Faster-RCNN。在结构上，Faster-RCNN将特征提取、proposal提取、bounding box

regression(rect refine)、classification都整合在了一个网络中，使得综合性能有较大提高，在检测速度方面尤为明显。

入参

参数	说明
待分类的JPEG图片	待分类的JPEG图片。
faster-rcnn model	模型名称。
Graph配置文件	串联整个运行流程的配置文件。

出参

参数	说明
物体置信度， 框的坐标	无
物体类别	无

调用示例

结果解析请参见**2.11.3 检测网络结果解析（Faster-RCNN）**。

2.11 网络结果解析 Sample

2.11.1 分类网络结果解析

通过模型管家加载模型

```
std::vector<hiai::AIModelDescription> model_desc_vec;
hiai::AIModelDescription model_desc_;
model_desc_.set_path(model_path);
model_desc_.set_key("");
model_desc_vec.push_back(model_desc_);
ret = ai_model_manager_->Init(config, model_desc_vec);
```

说明

模型管家相关接口请参见《Ascend 310 HiAI Engine API参考》中的4.1章节。

通过模型管家执行模型处理

```
ret = ai_model_manager_->Process(ai_context,
input_data_vec, output_data_vec, 0);
```

分类网络结果解析

```
将output转换为AINeuralNetworkBuffer
shared_ptr<AINeuralNetworkBuffer> output_tensor =
static_pointer_cast<AINeuralNetworkBuffer>(output_data_vec[0]);
//取出结果的buffer转换为float类型
```



```
float * result =(float *)output_tensor->GetBuffer();
int label_index = 0;
float max_value = 0.0;
//遍历查找最大的分类下标和对应的置信度值
for(int i=0; i< output_tensor->GetSize()/sizeof(float) ; i++)
{
    if(*(result + i) > max_value)
    {
        max_value = *(result + i);
        label_index = i;
    }
}
//结果展示
printf("label index:%d, Confidence :%f\n", label_index, max_value);
```

2.11.2 检测网络结果解析（SSD）

```
// 生成data_num和data_bbox信息
IMAGE_HEIGHT = 300;
IMAGE_WIDTH = 300;

//box_num 结果大小为4个字节，为一个float32的数，表示网络中检测到N个框
std::shared_ptr<hiai::AINeuralNetworkBuffer> output_data_num =
std::static_pointer_cast<hiai::AINeuralNetworkBuffer>(output_data_vec[1]);

// box_data 检测框的结果信息，shape(200,7)，数据类型为float32
std::shared_ptr<hiai::AINeuralNetworkBuffer> output_data_bbox =
std::static_pointer_cast<hiai::AINeuralNetworkBuffer>(output_data_vec[0]);
/*
|-----0-------1-------2------3------3------4------5--------6----|
| image_id| Label | score | xmin | ymin | xmax | ymax | reserve |----------bbox1
| image_id| Label | score | xmin | ymin | xmax | ymax | reserve |----------bbox2
|-------------------------------------------------------------|
*/
取对应的前N个框
```

2.11.3 检测网络结果解析（Faster-RCNN）

```
// 生成data_num和data_bbox信息，32个int32类型数，表示每个目标检测的框的数目
std::shared_ptr<hiai::AINeuralNetworkBuffer> output_data_num =
std::static_pointer_cast<hiai::AINeuralNetworkBuffer>(output_data_vec[0]);
/*
|--1---2---3---4---5------------32---------|
|  0 | 0 | 1 | 2 | 0 | ..... | 0 |
|------------------------------------------|
*/
表示label3 有1个框， label4 有两个框， label不包含background

// 生成box_data,维度为(32,608,8)
std::shared_ptr<hiai::AINeuralNetworkBuffer> output_data_bbox =
std::static_pointer_cast<hiai::AINeuralNetworkBuffer>(output_data_vec[1]);
/*
---------|-------------------------------------------------------------|
         | xmin | ymin | xmax | ymax | score | reserve | reserve | reserve |-----bbox1
         | xmin | ymin | xmax | ymax | score | reserve | reserve | reserve |-----bbox2
label1   |
         |
         | xmin | ymin | xmax | ymax | score | reserve | reserve | reserve |-----bbox608
---------|-------------------------------------------------------------|
.
.
.
---------|-------------------------------------------------------------|
         | xmin | ymin | xmax | ymax | score | reserve | reserve | reserve |-----bbox1
         | xmin | ymin | xmax | ymax | score | reserve | reserve | reserve |-----bbox2
label32  |
         |
         | xmin | ymin | xmax | ymax | score | reserve | reserve | reserve |-----bbox608
```



```
---------|---------------------------------------------------------|

box[i, j, 0] 表示第i个分类的第j个框 box的 xmin
box[i, j, 1] 表示第i个分类的第j个框 box的 ymin
box[i, j, 2] 表示第i个分类的第j个框 box的 xmax
box[i, j, 3] 表示第i个分类的第j个框 box的 ymax
box[i, j, 4] 表示第i个分类的第j个框 score
*/
```

