

A child wearing a pilot's cap and goggles sits on the shoulder of a large, white, humanoid robot. The child is pointing their right index finger towards a large, glowing globe in the background. The globe features a stylized world map with a grid pattern. The background is a light blue sky with several streaks of light, suggesting a futuristic or space-themed environment. The robot's head is turned slightly towards the child, and its right arm is visible, holding the child's hand.

昇腾CANN系列教程

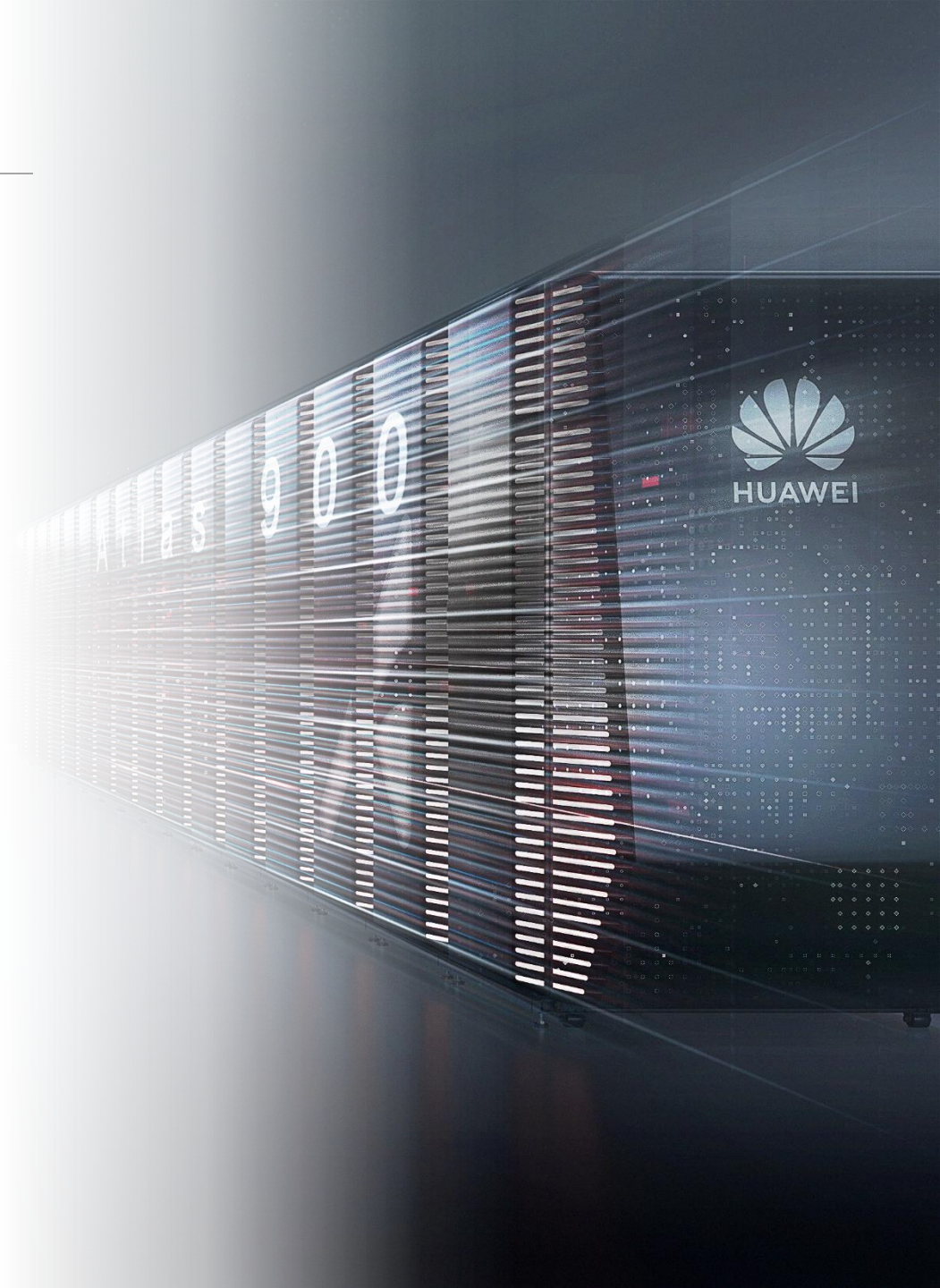
——**TBE**算子开发基本概念

关于本课程

- 本课程是介绍TBE（Tensor Boost Engine）算子开发工具的入门课程，涉及的主要内容：
 - 介绍TBE背景和基础知识
 - 介绍学习TBE工具前需要了解的基本概念
 - 介绍TBE算子多种开发方式

1 TBE基本概念

2 TBE开发方式



什么是TBE?

TBE (Tensor Boost Engine) 自定义算子开发工具:

- 一款华为自研的NPU算子开发工具
- 在TVM (Tensor Virtual Machine) 框架基础上扩展
- 提供了一套Python API来实施开发活动

TBE背景知识

基础知识自检:

知识点	知识库
领域特定语言(DSL)	http://wikipedia.moesalih.com/Domain_specific_language
Python编程语言	https://www.python.org/
编译器基础知识	编译原理（龙书）
Davinci基本架构	TBE高级课程有专门课时介绍
神经网络基础知识	http://wikipedia.moesalih.com/Neural_network

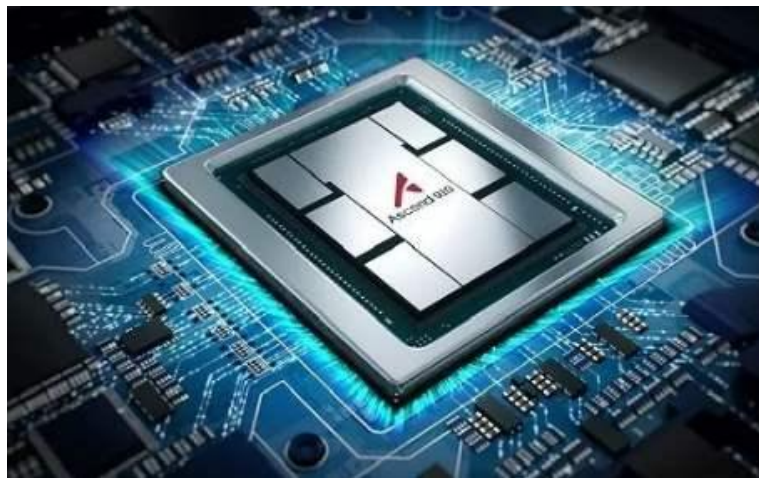
TBE基本概念——NPU

➤ NPU (Neural-network Processing Unit) , 神经网络处理器。

➤ 在维基百科中, NPU这个词条被直接指向了“人工智能加速器”, 释义是这样的:

“**人工智能加速器** (英语: **AI accelerator**) 是一类专用于人工智能 (特别是人工神经网络、机器视觉、机器学习等) 硬件加速的微处理器或计算系统。典型的应用包括机器人学、物联网等数据密集型应用或传感器驱动的任务。”

➤ 本系列课程中, NPU可以特指为昇腾系列AI处理器。



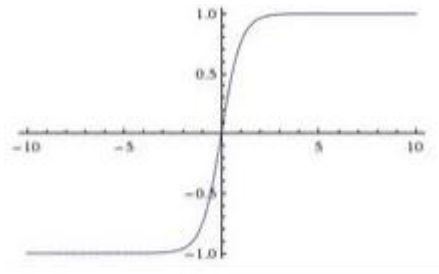
TBE基本概念——算子

- 算子基本概念

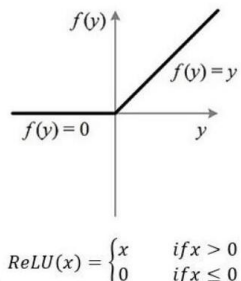
算子是一个函数空间到函数空间上的映射 $O: X \rightarrow X$ ；广义的讲，对任何函数进行某一项操作都可以认为是一个算子。于我们而言，我们所开发的算子是网络模型中涉及到的计算函数。在Caffe中，算子对应层中的计算逻辑，例如：卷积层（Convolution Layer）中的卷积算法，是一个算子；全连接层（Fully-connected Layer, FC layer）中的权值求和过程，是一个算子。

- 算子举例：

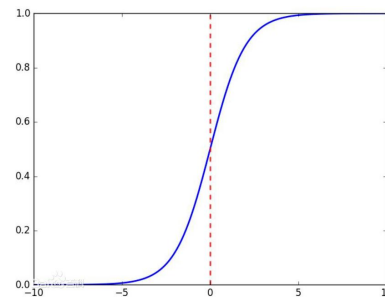
在网络模型中被用作激活函数的算子：tanh ReLU Sigmoid等



Tanh函数图像



ReLU函数图像



Sigmoid函数图像

TBE基本概念——算子（续）

- 算子类型（Type）：
 - 算子的type，代表算子的类型，例如卷积算子的类型为Convolution，在一个网络中同一类型的算子可能存在多个。
- 算子名称（Name）：
 - 算子的名称，用于标识网络中的某个算子，同一网络中算子的名称需要保持唯一。如下图所示conv1，pool1，conv2都是此网络中的算子名称，其中conv1与conv2算子的类型为Convolution，表示分别做一次卷积运算。



TBE基本概念——算子（续）

- 张量（Tensor）：
 - Tensor是TBE算子中的数据，包括输入数据与输出数据，TensorDesc（Tensor描述符）是对输入数据与输出数据的描述，TensorDesc数据结构包含如下属性：

属性	定义
名称（name）	用于对Tensor进行索引，不同Tensor的name需要保持唯一。
形状（shape）	Tensor的形状，比如（10,）或者（1024, 1024）或者（2, 3, 4）等。 默认值：无 形式：(i1, i2,...in)，其中i1, i2, in为正整数
数据类型（dtype）	功能描述：指定Tensor对象的数据类型。 默认值：无 取值范围：float16, float32, int8, int16, int32, uint8, uint16, bool。 说明 不同计算操作支持的数据类型不同，详细请参见9 接口参考。
数据排布格式（format）	多个维度的排布顺序。

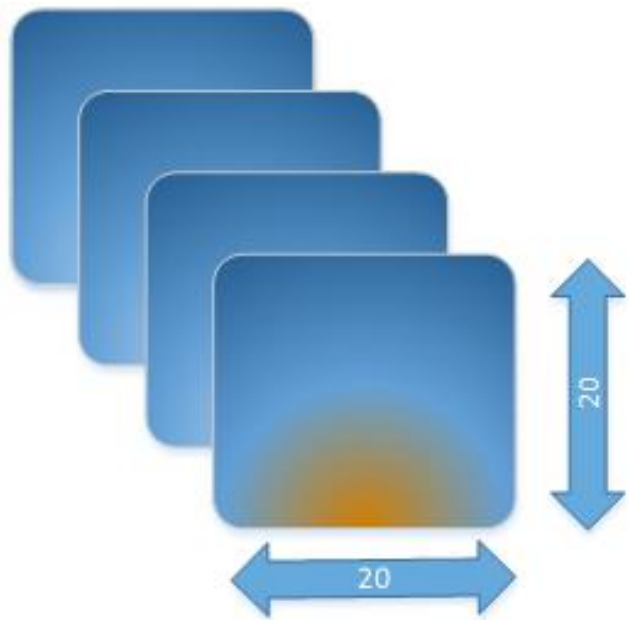
TBE基本概念——算子（续）

- 张量 (Tensor) :
 - 形状 (Shape) :
 - 张量的形状，以(D0, D1, ... ,Dn-1)的形式表示，D0到Dn是任意的正整数。
 - 如形状(3,4)表示第一维有3个元素，第二维有4个元素，(3,4)表示一个3行4列的矩阵数组。
 - 在形状的中括号中有多少个数字，就代表这个张量是多少维的张量。形状的第一个元素要看张量最外层的中括号中有几个元素，形状的第二个元素要看张量中从左边开始数第二个中括号中有几个元素，依此类推。例如：

张量	形状
1	(0,)
[1,2,3]	(3,)
[[1,2],[3,4]]	(2,2)
[[[1,2],[3,4]], [[5,6],[7,8]]]	(2,2,2)

TBE基本概念——算子（续）

- 张量形状的物理含义：
 - 假设我们有这样一个 $\text{shape}=(4, 20, 20, 3)$ 。
 - 假设有一些照片，每个像素点都由红/绿/蓝3色组成，即 shape 里面3的含义，照片的宽和高都是20，也就是 $20*20=400$ 个像素，总共有4张照片，这就是 $\text{shape}=(4, 20, 20, 3)$ 的物理含义。



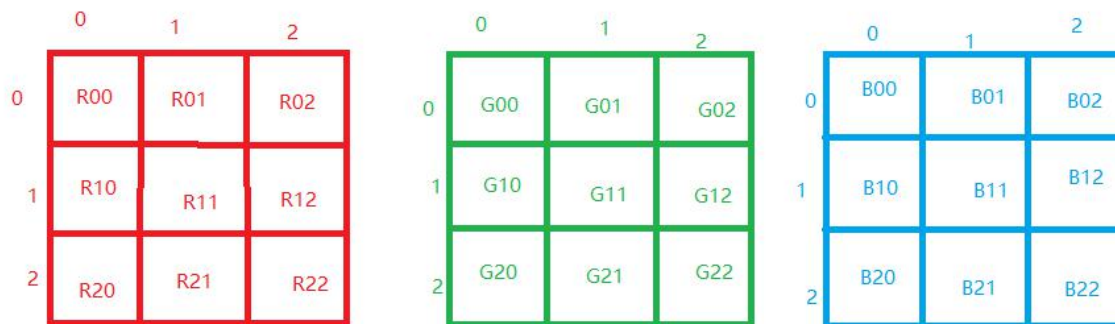
TBE基本概念——算子（续）

- 张量 (Tensor) :
 - 数据排布格式 (Format) :
 - 在深度学习框架中，多维数据通过多维数组存储，比如卷积神经网络的特征图用四维数组保存，四个维度分别为批量大小 (Batch, N)、特征图高度 (Height, H)、特征图宽度 (Width, W) 以及特征图通道 (Channels, C)。
 - 由于数据只能线性存储，因为这四个维度有对应的顺序。不同深度学习框架会按照不同的顺序存储特征图数据，比如 Caffe，排列顺序为 [Batch, Channels, Height, Width]，即 NCHW。Tensorflow 中，排列顺序为 [Batch, Height, Width, Channels]，即 NHWC。
 - 如下图所示，以一张格式为 RGB 的图片为例，NCHW 实际存储的是 “RRRGGG BBB”，同一通道的所有像素值顺序存储在一起，而 NHWC 实际存储的则是 “RGBRGBRGB”，多个通道的同一位置的像素值顺序存储在一起。



TBE基本概念——算子（续）

- 张量的数据排布格式——换个图再看一下：



TBE基本概念——算子（续）

- 算子属性

- 不同的算子包含的属性值不同，下面介绍几个常见的算子属性：

- 轴（Axis）：

- axis代表张量中维度的下标，比如张量a是一个5行6列的二维数组，即shape是(5,6)，则axis=0表示是张量中的第一维，即行。axis=1表示是张量中的第二维，即列。
 - 例如张量数据[[[1,2],[3,4]], [[5,6],[7,8]]]，Shape为(2,2,2)，则轴0代表第一个维度的数据即[[1,2],[3,4]]与[[5,6],[7,8]]这两个矩阵，轴1代表第二个维度的数据即[1,2]、[3,4]、[5,6]、[7,8]这四个数组，轴2代表第三个维度的数据即1，2，3，4，5，6，7，8这八个数。
 - 轴axis可以为负数，此时表示是倒数第axis个维度。
 - N维Tensor的轴有：0，1，2，.....，N-1。

shape: (4, 20, 20, 3)

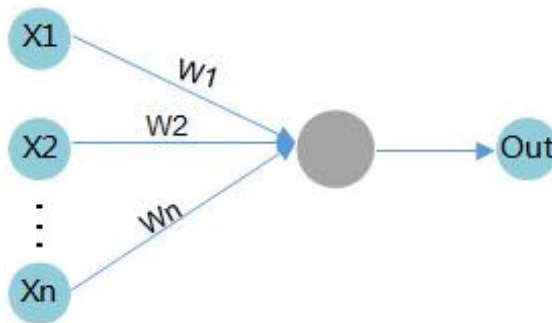
↑ ↑ ↑ ↑
Axis: 0 1 2 3

TBE基本概念——算子（续）

- 算子属性

- 权重 (Weight) :

- 当输入数据进入计算单元时，会乘以一个权重。例如，如果一个算子有两个输入，则每个输入会分配一个关联权重，一般将认为较重要数据赋予较高的权重，不重要的数据赋予较小的权重，为零的权重则表示特定的特征是无需关注的。
 - 如下图所示，假设输入数据为 $X1$ ，与其相关联的权重为 $W1$ ，那么在通过计算单元后，数据变为了 $X1*W1$ 。

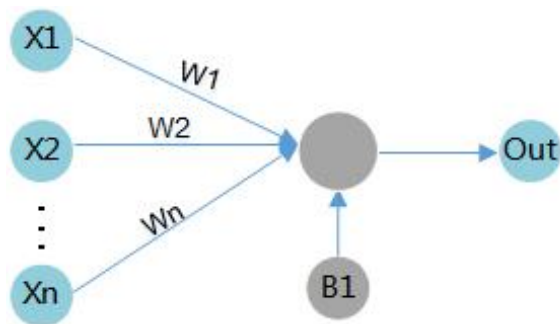


TBE基本概念——算子（续）

- 算子属性

- 偏差 (Bias) :

- 偏差是除了权重之外，另一个被应用于输入数据的线性分量。它被加到权重与输入数据相乘的结果中，用于改变权重与输入相乘所得结果的范围。
 - 如图所示，假设输入数据为 $X1$ ，与其相关联的权重为 $W1$ ，偏差为 $B1$ ，那么在通过计算单元后，数据变为了 $X1*W1+B1$ 。



TBE基本概念——算子（续）

- 升维—广播机制（Broadcast）：
 - TBE支持的广播规则：可以将一个数组的每一个维度扩展为一个固定的shape，需要被扩展的数组的每个维度的大小或者与目标shape相等，或者为1，广播会在元素个数为1的维度上进行。
 - 例如：原数组a的维度为（2, 1, 64），目标shape为（2, 128, 64），则通过广播可以将a的维度扩展为目标shape（2, 128, 64）。
 - TBE的计算接口加、减、乘、除等不支持自动广播，要求输入的两个Tensor的shape相同，所以操作前，我们需要先计算出目标shape，然后将每个输入Tensor广播到目标shape再进行计算。
 - 例如，Tensor A的shape为（4, 3, 1, 5），Tensor B的shape为（1, 1, 2, 1），执行Tensor A + Tensor B，具体计算过程如下：



TBE基本概念——算子（续）

- 1. 计算出目标shape。

$$\begin{matrix} A(4, 3, 1, 5) \\ B(1, 1, 2, 1) \end{matrix} \longrightarrow \begin{matrix} A(4, 3, 1, 5) \\ B(1, 1, 2, 1) \\ C(4, 3, 2, 5) \end{matrix}$$

- 取Tensor A与Tensor B中每个维度的大值，作为目标Shape，C (4, 3, 2, 5)。
- 2. 调用广播接口分别将Tensor A与Tensor B扩展到目标Shape C。
- 3. 调用计算接口，进行Tensor A + Tensor B。
 - 如果需要广播的tensor的shape不满足每一个维度的大小与目标shape相等或者为1的要求，则无法进行广播，如下示例所示：

$$\begin{matrix} (4, 3, 2, 5) \\ (1, 3, 2, 3) \end{matrix} \quad \begin{matrix} (4, 3, 2, 5) \\ (1, 3, 2, 1) \end{matrix}$$

TBE基本概念——算子（续）

- 降维（Reduction）：
 - 前面我们介绍升维-广播机制，有升就有降，Reduction是将多维数组的指定轴做降维操作。
 - Reduction算子的关键属性：

- **ReductionOp**: 常见支持的操作类型，包含四种类型：

算子类型	说明
SUM	对被reduce的所有轴求和。
MIN	对被reduce的所有轴求最小值。
MAX	对被reduce的所有轴求最大值。
PROD	对被reduce的所有轴求乘积。

- **Axis**: Reduction需要指定若干个轴，会对这些轴进行reduce操作，取值范围为：[-N, N-1]。
- **keepdim**: 对指定轴降维后，该轴是被消除还是保留为1，即此维度是否保留。取值为0表示消除，1表示保留。
 - 比如，输入的张量的形状为(5,6,7,8)。
 - ~ 如果指定的轴是3，keepdim是0，则输出tensor的形状为(5,6,7)。
 - ~ 如果指定的轴是2，keepdim是1，则输出tensor的形状为(5,6,1,8)。
 - ~ 如果指定的轴是{1,2}，keepdim是0，则输出tensor的形状为(5,8)。
 - ~ 如果指定的轴是{0,1,2}，keepdim是1，则输出tensor的形状为(1,1,1,8)。

TBE基本概念——算子（续）

- 下面通过示例了解降维操作。

- 如果Axis=0 进行降维, 对2维矩阵来说就是行, 也就是对这个2维矩阵每行对应的数据进行相加, 得到[2,2,2], 降为1维, 如图所示:

$$\begin{array}{c} [[1, 1, 1] \\ + \quad + \quad + \\ [1, 1, 1] \\ = [2, 2, 2] \end{array}$$

- 如果Axis=1, 就是每列对应的数据进行相加。

$$\begin{array}{c} [[1, +1, +1] \\ [1, +1, +1]] \end{array} = \begin{array}{c} [3, \\ 3] \end{array}$$

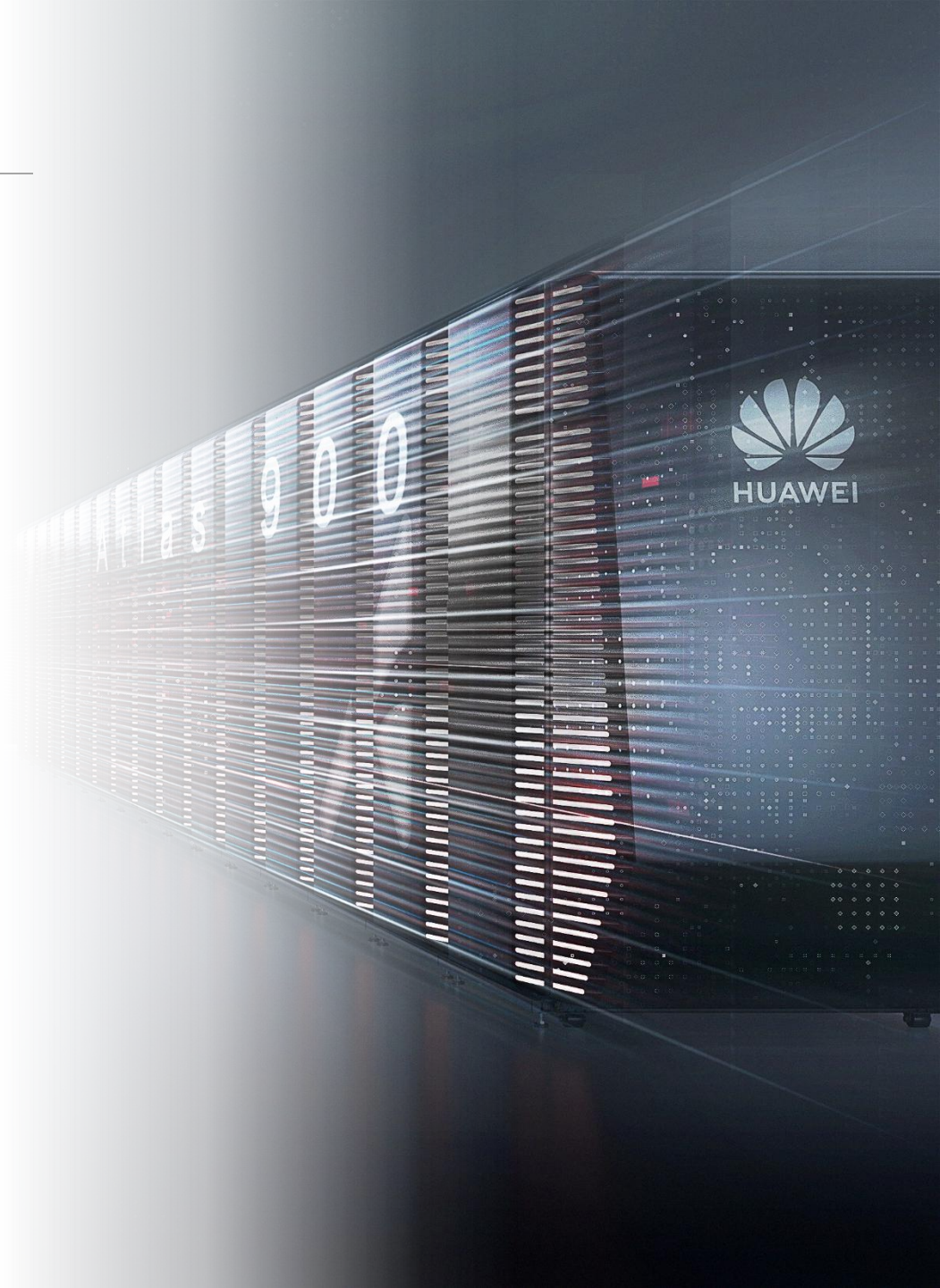
- 如果Axis=[0,1], 可以理解为先对轴0进行降维求和得到[2,2,2], 再对[2,2,2]继续降维求和得到6, 最后得到是0维;
- 如果Axis=[], 就是不降维, 原样输出。
- 如果Axis为空, 就是对所有维度进行降维, 最后得到0维的标量。

TBE基本概念——算子（续）

- 转换算子：
 - 转换算子是一种特殊的算子，用于转换Tensor的Data Type与Format等信息，使得图中上下游节点能够被正常处理。
 - 图经过一系列处理后，如下沉、优化、融合等，节点中的Tensor信息可能发生了变化，所以需要插入转换算子使图能够继续被正常处理。

1 TBE基本概念

2 TBE开发方式



TBE算子开发方式

开发方式	使用场景
TBE DSL	<p>开发难度： 易</p> <p>适用人群： 入门开发者， 仅需要了解神经网络和TBE DSL相关知识</p> <p>特点： TBE工具提供自动优化机制， 给出较优的调度流程</p>
TIK	<p>开发难度： 难</p> <p>适用人群： 高级开发者， 需要掌握Davinci硬件缓冲区架构， 对性能有较高要求</p> <p>特点： 接口偏底层， 用户需要自己控制数据流以及算子的硬件调度（schedule）</p>

TBE算子开发方式——DSL

接口分类	简介	示例
Math	对Tensor中每个原子值分别做相同操作的计算接口	vadd(lhs, rhs): 两个Tensor逐元素相加
NN	神经网络相关计算接口	broadcast(var, shape, output_dtype=None): Tensor做广播操作
Cast	取整计算接口, 对输入Tensor中的每个元素按照一定的规则进行取整操作	floor(raw_tensor): 求Tensor的向下取整
Reduce	对Tensor按轴进行相关操作的计算接口	reduce_max(raw_tensor, axis, keepdims=False): 指定轴求最大值
卷积	包含2D卷积运算和3D卷积运算的相关接口	conv(data, weight, para_dict, optim_dict=None, dsl_flag=True): 计算float16的2-D卷积
auto schedule	调度接口	auto_schedule(outs, option=None): 把定义好的计算过程生成schedule对象

TBE算子开发方式——TIK

接口分类	简介	示例
芯片配置管理	管理和配置昇腾AI处理器相关信息	tik.Dprofile()
TIK容器管理	用于创建TIK DSL容器	tik.Tik(tik.Dprofile())
	将TIK DSL容器中的语句编译成为D芯片可执行代码	tik_instance.BuildCCE(kernel_name, inputs, outputs)
程序控制	If语句	tik.Tik.if_scope(cond)
	For语句	tik.Tik.for_range(begint, endt, name= "i" , thread_num = 1, thread_type = "whole", bock_num = 1, dtype = "int32" , for_type= "serial")
指令API	数据搬运	data_move (dst, src, sid, nburst, burst, src_stride, dst_stride, *args, **argv)
	矢量运算	单目矢量如vec_In、双目矢量如vec_add、归约运算如vec_reduce_max等
	矩阵运算	conv2d、fixpipe、matmul
	数据转换	vec_trans、vec_trans_scatter

Thank you.

昇腾开发者社区



<http://hiascend.com>

把数字世界带入每个人、每个家庭、
每个组织，构建万物互联的智能世界。

**Bring digital to every person, home, and
organization for a fully connected,
intelligent world.**

**Copyright©2020 Huawei Technologies Co., Ltd.
All Rights Reserved.**

The information in this document may contain
predictive
statements including, without limitation, statements
regarding
the future financial and operating results, future
product
portfolio, new technology, etc. There are a number of
factors that
could cause actual results and developments to differ
materially
from those expressed or implied in the predictive
statements.
Therefore, such information is provided for reference
purpose
only and constitutes neither an offer nor an

