AI编译器-系列之前端优化

图层 IR

ZOMI

Talk Overview

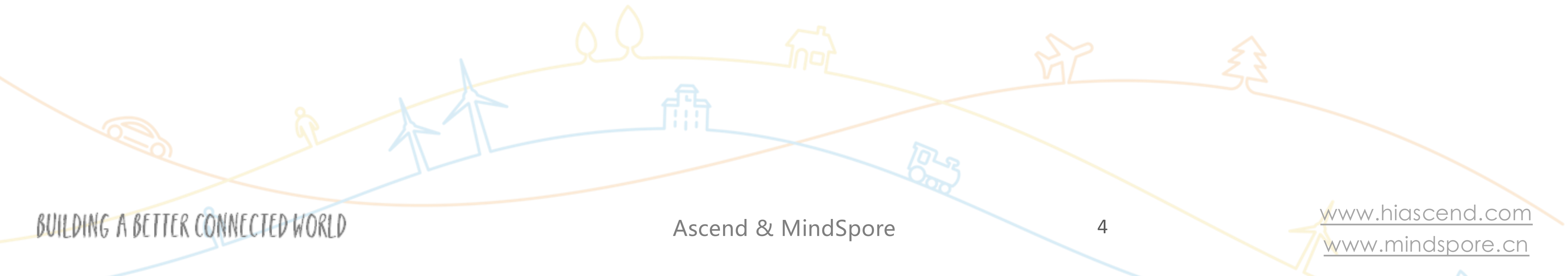
I. AI 编译器前端优化

- 图层 - Graph IR
- 算子融合 - OP Fusion
- 布局转换 - Layout Transform
- 内存分配 - Memory Allocation
- 常量折叠 – Constant Fold
- 公共子表达式消除 - CSE
- 死代码消除 - DCE
- 代数简化 - ARM

Talk Overview

I. **计算图的表示**
   - 计算图的基本构成
   - 静态计算图和动态计算图
   - AI 框架如何生成计算图
   - 对 AI 编译器的作用

计算图基本构成



Computation graph

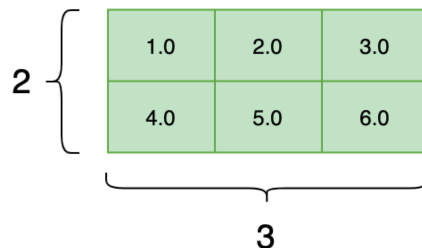
1. 用来表示深度学习网络模型在训练与推理过程中计算逻辑与状态
2. AI框架在后端会将Python构建网络模型前向与反向梯度计算以计算图的形式来表示
3. 由基本数据结构张量 (Tensor) 和基本运算单元算子 (Operator) 构成
4. 计算图中常用节点来表示算子，节点间的有向线段来表示张量状态，同时也描述了计算间的依赖关系

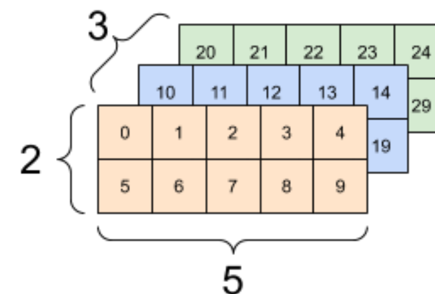
www.hiascend.com
www.mindspore.cn

基于计算图的AI框架：基本组成

基本数据结构：Tensor 张量

- Tensor 形状： [2, 3, 4, 5]
- 元素类型：int, float, string, etc.





基本运算单元：Operator 算子

- 由最基本的代数算子组成
- 根据深度学习结构组成复杂算子
- N个输入Tensor，M个输出Tensor

Add	Log	While
Sub	MatMul	Merge
Mul	Conv	BroadCast
Div	BatchNorm	Reduce
Relu	Loss	Map
Floor	Sigmoid	…..

Huawei Confidential. Ascend & MindSpore

www.hiascend.com
www.mindspore.cn

基于数据流图（DAG）的计算框架

DAG 表示计算逻辑和状态

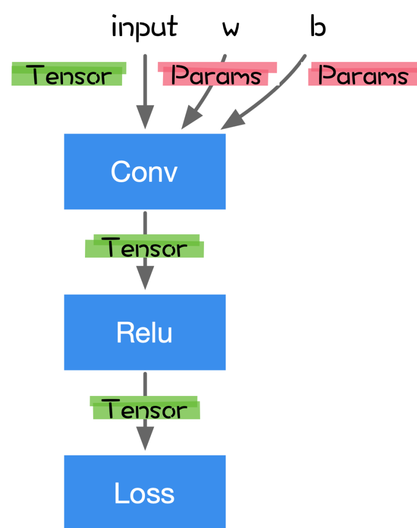
- 节点代表 Operator
- 边代表 Tensor

特殊的操作

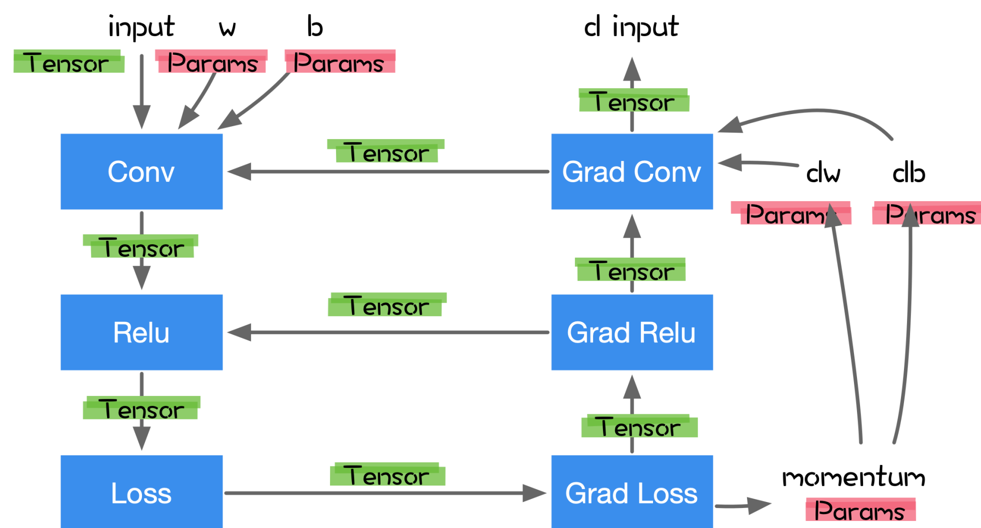
- 如：For/While 等构建控制流

特殊的边

- 如：控制边表示节点间依赖



Forward



Backward

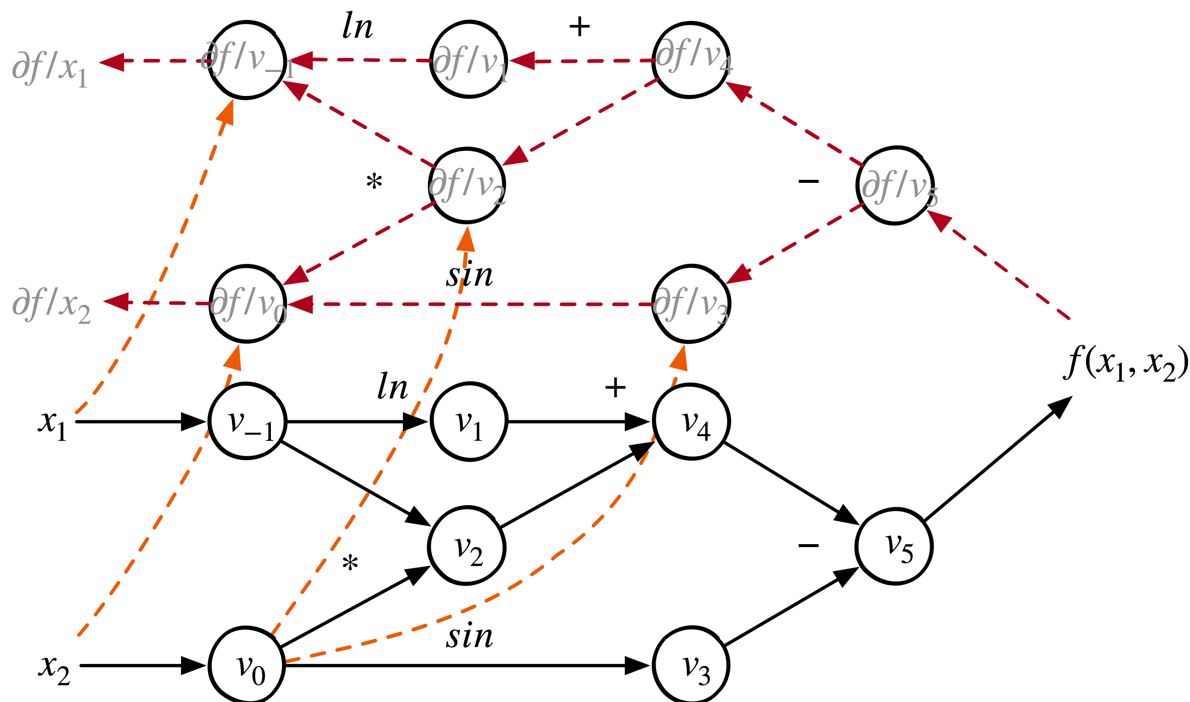
www.hiascend.com
www.mindspore.cn

AI框架如何生成计算图？

Huawei Confidential. Ascend & MindSpore
www.hiascend.com
www.mindspore.cn

计算图和自动微分

有向无环图（DAG, Directed Acyclic Graph）



Reverse Adjoint (Derivative) Trace

$\bar{\boldsymbol{x}}_1$	$= \bar{\boldsymbol{v}}_{-1}$		$= \mathbf{5.5}$
$\bar{\boldsymbol{x}}_2$	$= \bar{\boldsymbol{v}}_0$		$= \mathbf{1.716}$
$\bar{v}_{-1}$	$= \bar{v}_{-1} + \bar{v}_1 \frac{\partial v_1}{\partial v_{-1}}$	$= \bar{v}_{-1} + \bar{v}_1 / v_{-1}$	$= 5.5$
$\bar{v}_0$	$= \bar{v}_0 + \bar{v}_2 \frac{\partial v_2}{\partial v_0}$	$= \bar{v}_0 + \bar{v}_2 \times v_{-1}$	$= 1.716$
$\bar{v}_{-1}$	$= \bar{v}_2 \frac{\partial v_2}{\partial v_{-1}}$	$= \bar{v}_2 \times v_0$	$= 5$
$\bar{v}_0$	$= \bar{v}_3 \frac{\partial v_3}{\partial v_0}$	$= \bar{v}_3 \times \cos v_0$	$= -0.284$
$\bar{v}_2$	$= \bar{v}_4 \frac{\partial v_4}{\partial v_2}$	$= \bar{v}_4 \times 1$	$= 1$
$\bar{v}_1$	$= \bar{v}_4 \frac{\partial v_4}{\partial v_1}$	$= \bar{v}_4 \times 1$	$= 1$
$\bar{v}_3$	$= \bar{v}_5 \frac{\partial v_5}{\partial v_3}$	$= \bar{v}_5 \times (-1)$	$= -1$
$\bar{v}_4$	$= \bar{v}_5 \frac{\partial v_5}{\partial v_4}$	$= \bar{v}_5 \times 1$	$= 1$
$\bar{v}_5$	$= \bar{y}$	$= 1$	

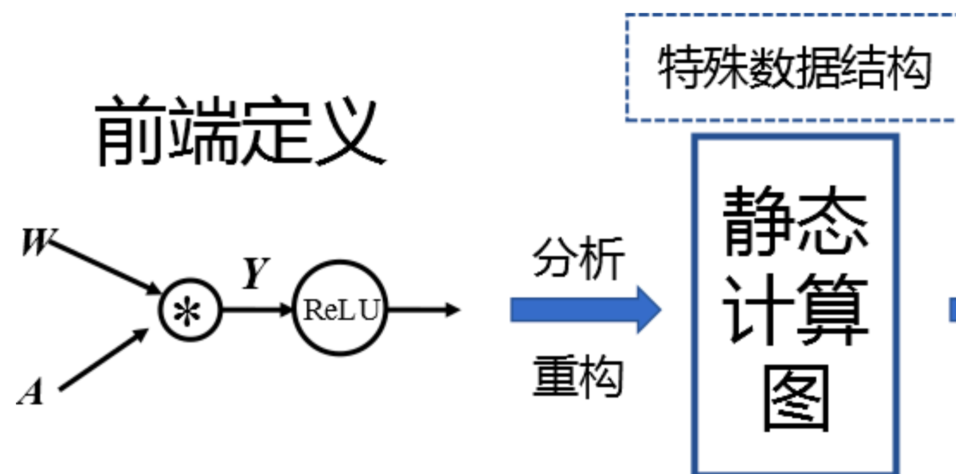
Huawei Confidential. Ascend & MindSpore

www.hiascend.com
www.mindspore.cn

AI框架生成静态计算图

- 静态图模式下：使用前端语言定义模型形成完整的程序表达后，不使用前端语言解释器进行执行，将前端描述的完整模型交给AI框架。AI框架先对API代码分析，获取网络层之间的连接拓扑关系以及参数变量设置、损失函数等，接着用静态数据结构来描述拓扑结构及其他神经网络模型组件。

```
1   class Network(nn.Cell):
2       def __init__(self):
3           super().__init__()
4           self.flatten = nn.Flatten()
5           self.dense_relu_sequential = nn.SequentialCell(
6               nn.Dense(28*28, 512),
7               nn.ReLU())
8
9       def construct(self, x):
10          x = self.flatten(x)
11          logits = self.dense_relu_sequential(x)
12          return logits
13
```



www.hiascend.com
www.mindspore.cn

AI框架生成动态计算图

- 动态图采用前端语言自身的解释器对代码进行解析，利用AI框架本身的算子分发功能，算子会即刻执行并输出结果。动态图模式采用用户友好的命令式编程范式，使用前端语言构建神经网络模型更加简洁。

- **优点**：动态图模式灵活的执行计算特性，动态生成可以使用前端语言的原生控制流，充分发挥前端语言的编程友好性特性。
- **缺点**：不过动态生成中完整的网络结构在执行前是未知的，不能使用静态图中的图优化技术来提高计算执行性能。

www.hiascend.com
www.mindspore.cn

动态图和静态图

www.hiascend.com
www.mindspore.cn

动态图和静态图比较

特性	静态图	动态图
即时获取中间结果	否	是
代码调试难易	难	简单
控制流实现方式	特定的语法	前端语言语法
性能	优化策略多，性能更佳	图优化受限，性能较差
内存占用	内存占用少	内存占用相对较多
部署能力	可直接部署	不可直接部署

www.hiascend.com
www.mindspore.cn

动态图转换为静态图

- **基于追踪转换** ：以动态图模式执行并记录调度的算子，构建和保存为静态图模型。
- **基于源码转换** ：分析前端代码将动态图代码自动转为静态图代码，底层使用静态图执行运行。

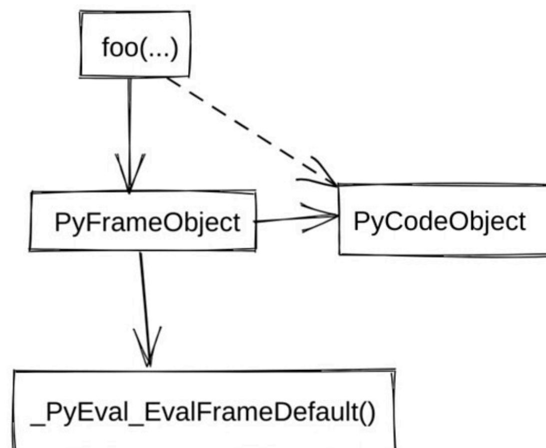
动态图转换为静态图

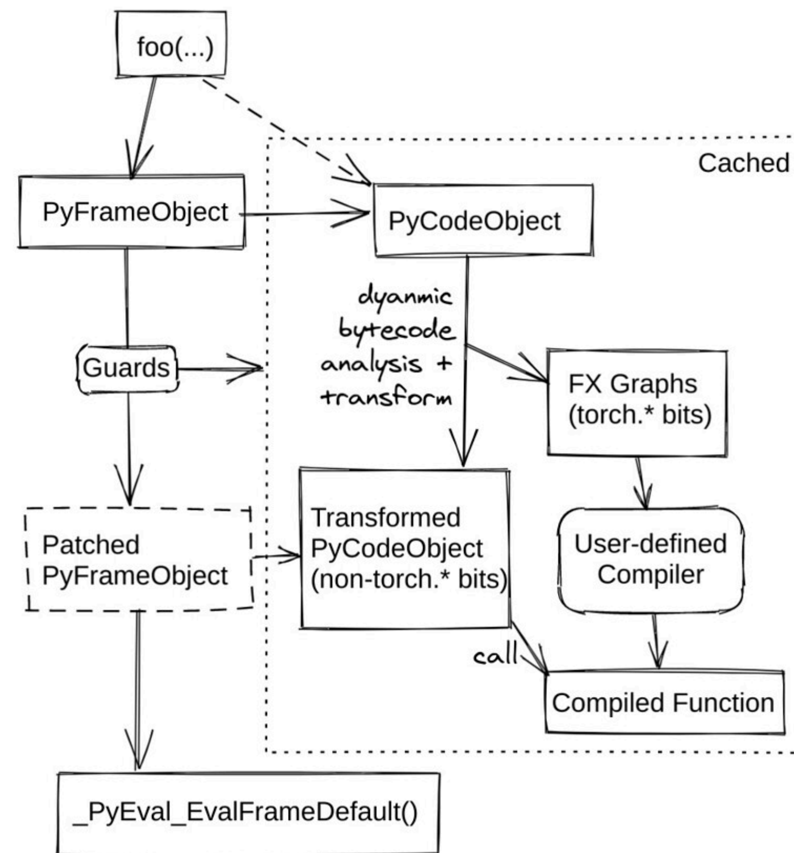
PyTorch 1.X

- TorchScript
- Torch JIT
- Torch FX
- Lazy Tensor

PyTorch 2.0

- TorchDynamo

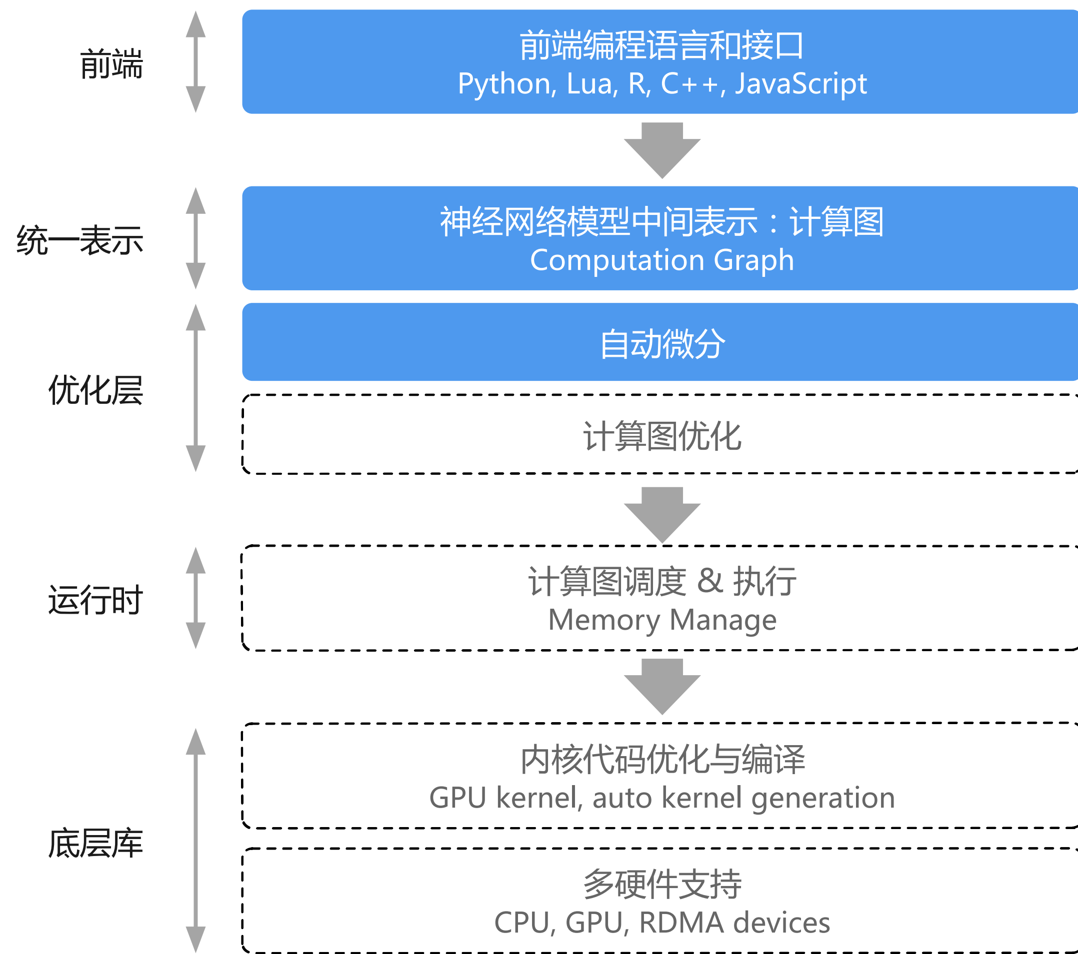




www.hiascend.com
www.mindspore.cn

Benefit（I）：方便底层编译优化

- 统一表示来描述神经网络训练的全过程
- 可以序列化保存，不需要再次编译前端源代码

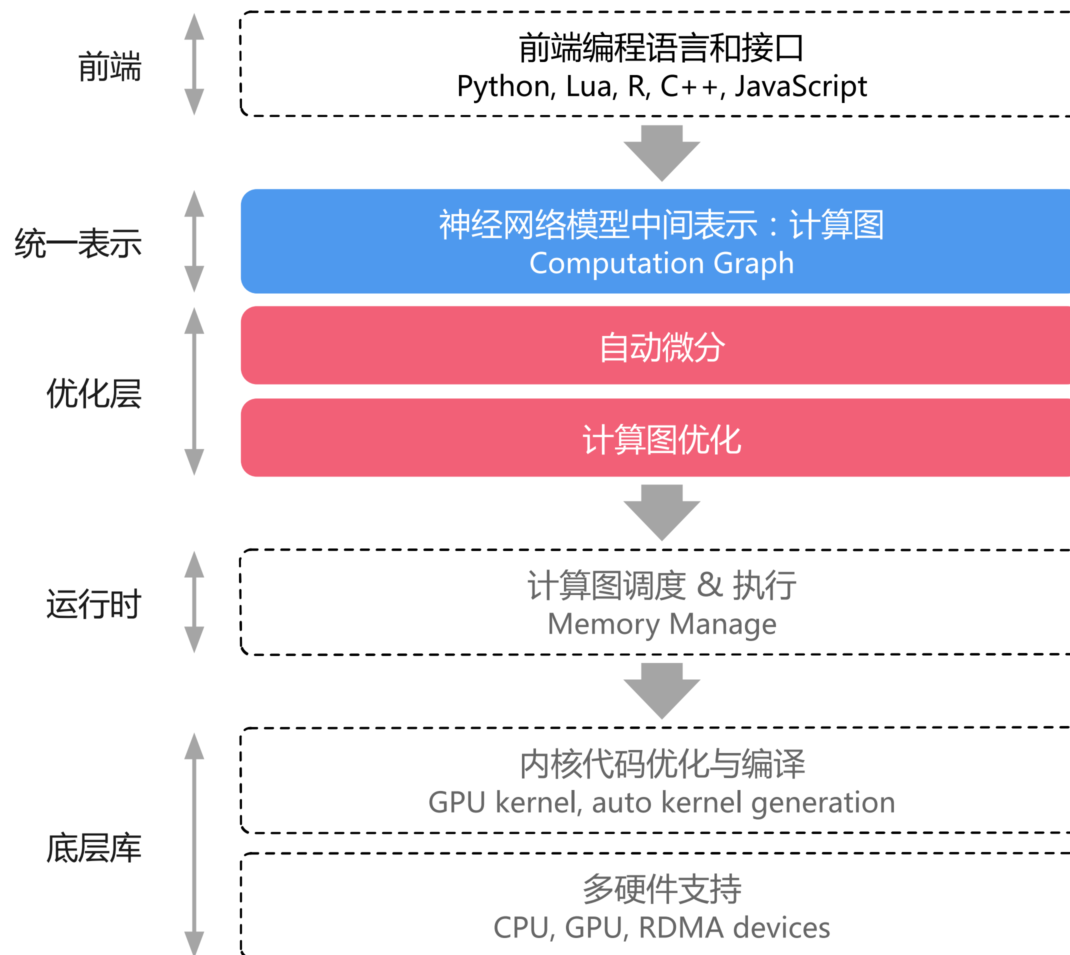


对编译器作用

图优化

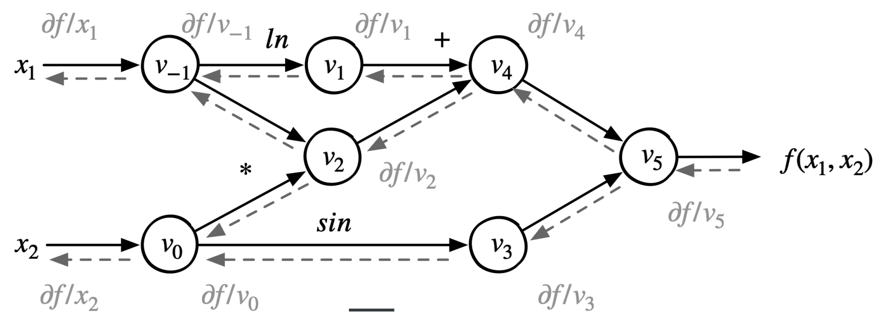
- 计算图的出现允许 AI 框架在执行前看到深度学习模型定义全局信息
- 计算图作为 AI 框架中的高层中间表示，可以通过图优化 Pass 去化简计算图或提高执行效率

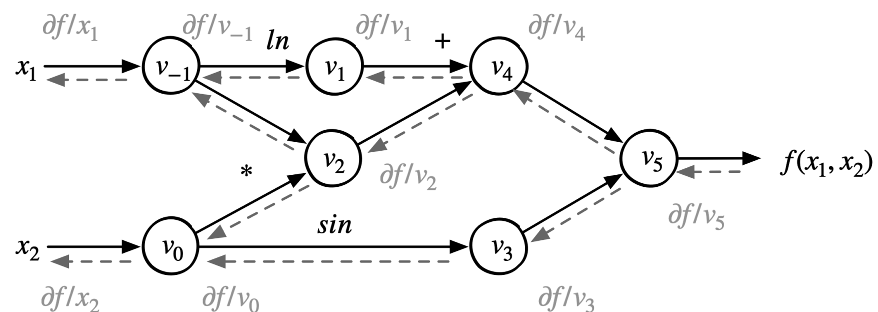
利用反向微分计算梯度通常实现为数据流图上的一个优化 Pass



www.hiascend.com
www.mindspore.cn

图优化



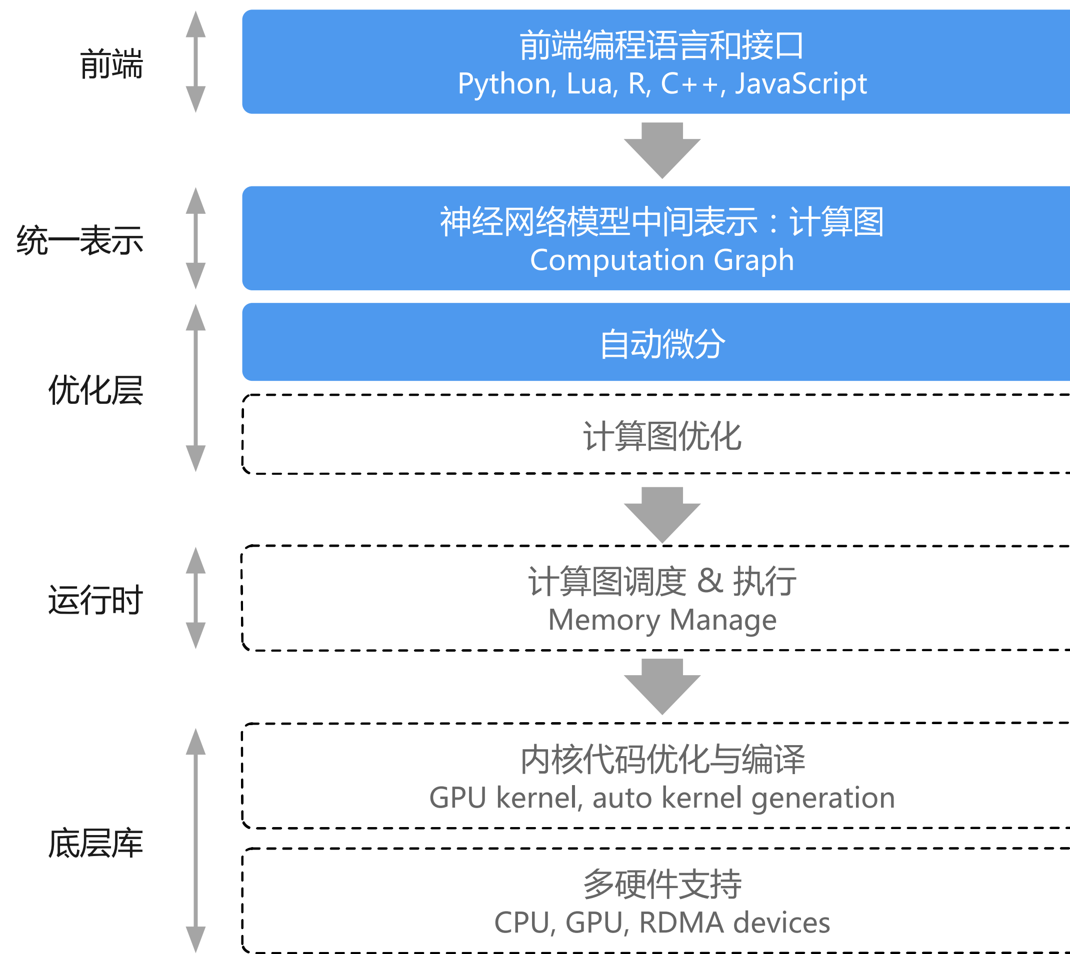


www.hiascend.com
www.mindspore.cn

Benefit（I）：方便底层编译优化

- 统一表示来描述神经网络训练的全过程
- 可以序列化保存，不需要再次编译前端源代码
- 将神经网络模型中间表示转换为不同硬件代码
- 直接部署在硬件上，提供高效的推理服务

- 编译期可对计算过程的数据依赖进行分析：
  - 简化数据流图
  - 动态和静态内存优化
  - 预计算算子间的调度策略
  - 改善运行时Runtime性能



www.hiascend.com
www.mindspore.cn

Benefit（II）：分层优化，便于扩展

- 切分出三个解耦的优化层：
  - 计算图优化
  - 运行时调度优化
  - 算子/内核执行优化
- 新网络模型结构/新训练算法，扩展步骤：
  - 计算图层添加新算子
  - 针对不同硬件内核，实现计算优化
  - 注册算子和内核函数，运行时派发硬件执行

www.hiascend.com
www.mindspore.cn

Cons

静态图+AI编译器流程：

- 生成采用先编译后执行的方式，编译阶段和执行阶段分离，前端语言构建的神经网络模型经过编译后，计算图结构便固定执行阶段不再改变

Cons：

1. 使用前端语言编写神经网络模型以及定义模型训练过程代码较为繁琐，掌握图控制流方法有一定学习成本，因此熟练掌握并使用静态图模式对于初学者并不友好
2. 经过优化用于执行的计算图结构与原始代码有较大的差距，导致代码中的错误难以定位到准确位置，增加了代码调试难度。在神经网络模型开发迭代环节，不能即时打印中间结果。
3. 在源码中添加输出中间结果的代码，需要将源码重新编译，再调用执行器才能获取相关信息，降低了代码调试效率。

www.hiascend.com
www.mindspore.cn

What about The Future ?

1. TorchDynamo能够解决99%的场景问题吗？
2. MindSpore的优化主要针对静态图+AI编译器，动态图转静态图+AI编译器是否有更好的方案？

www.hiascend.com
www.mindspore.cn

Inference

THANK YOU

Copyright©2014 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 acceptance. Huawei may change the information at any time without notice.