计算图与自动微分

ZOMI

关于本内容

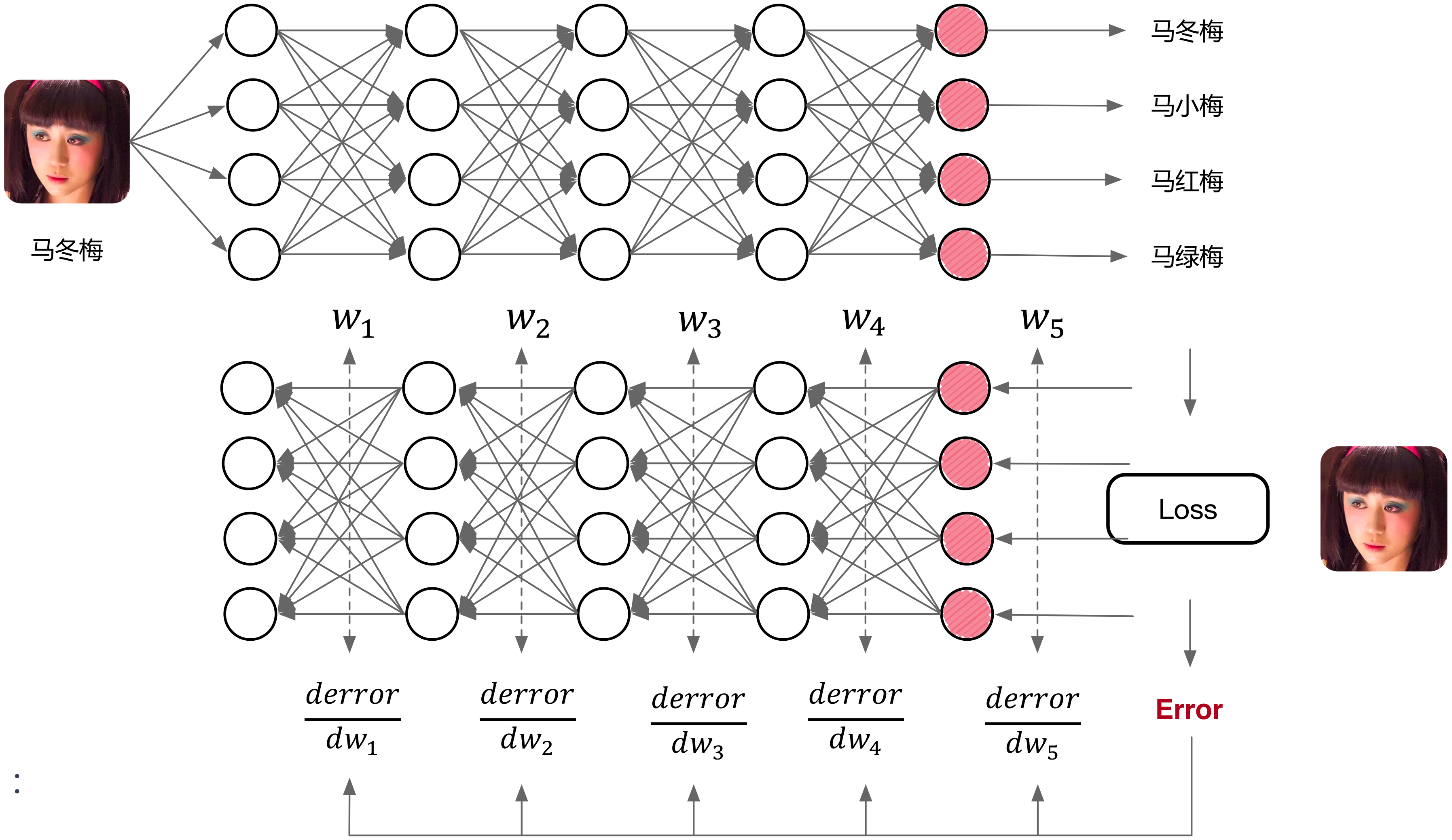
1. **内容背景**
   - 计算图基础介绍
2. **具体内容**
   - 计算图（数据流图）：AI系统化问题 – 计算图的提出
   - 计算图和自动微分：深度学习与微分 - 回顾自动微分 – 计算图表达自动微分
   - 图的调度和执行：单算子调度 – 图切多设备调度 – 控制流控制
   - 计算图的挑战与未来

深度学习训练流程：主要计算阶段

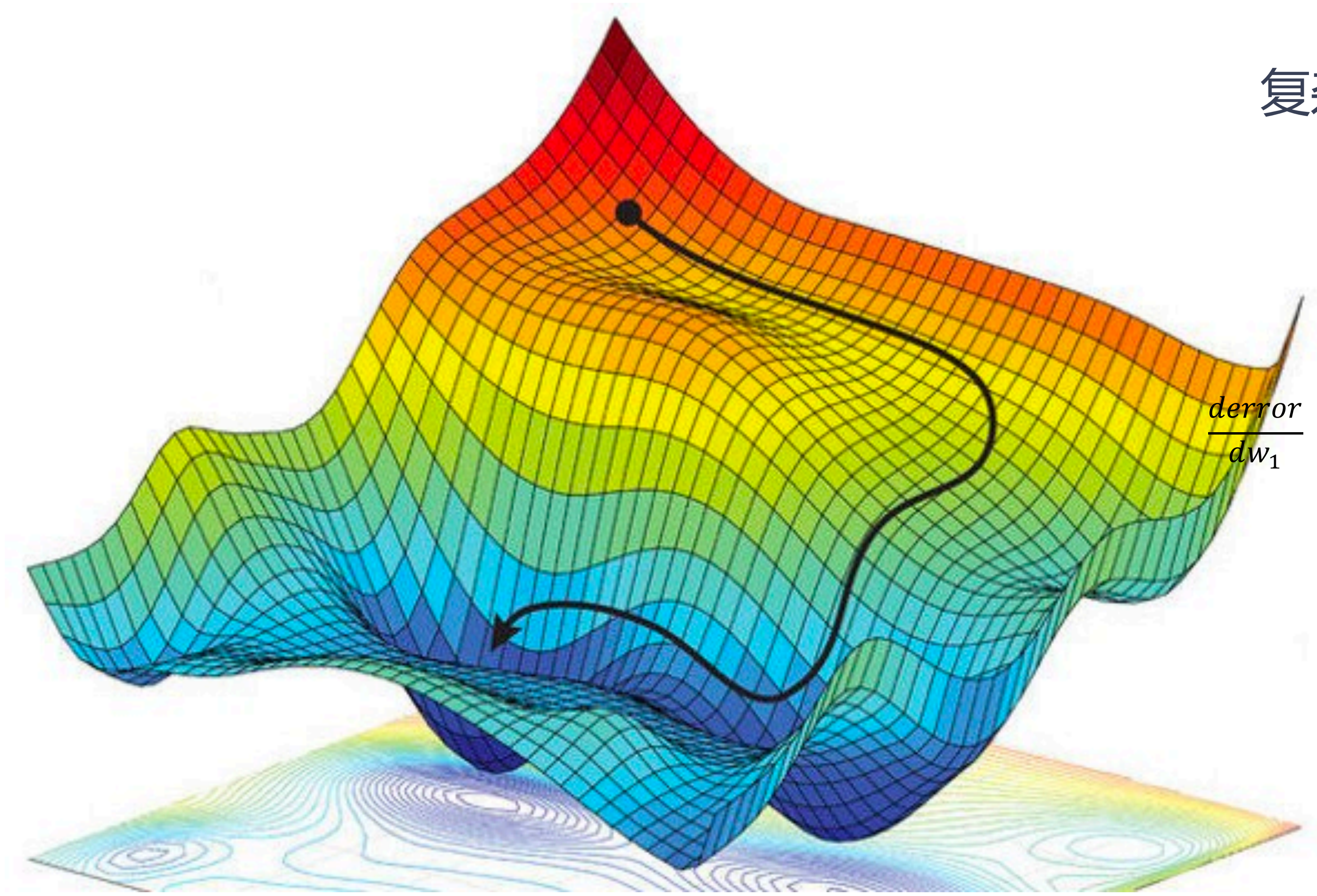
1. 前向计算:

2. 反向计算：

3. 更新可学习的权重参数：



深度学习训练流程：数学表示

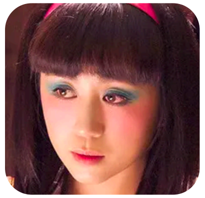


复杂的带参数的高度非凸函数

参数的一阶梯度迭代更新

$\frac{derror}{dw_1}$ $\frac{derror}{dw_2}$ $\frac{derror}{dw_3}$ $\frac{derror}{dw_4}$ $\frac{derror}{dw_5}$

Loss



训练流程核心：求导

- 求导是一个经典的问题

$$L(x) = exp(exp(x) + exp(x)^2) + sin(exp(x) + exp(x)^2)$$

- 深度学习计算的核心：计算网络模型的参数，并更新其梯度

$$L(w) = Loss(f(w, x_i), y_i) \Rightarrow {\sigma L(w)}/{\sigma w}$$

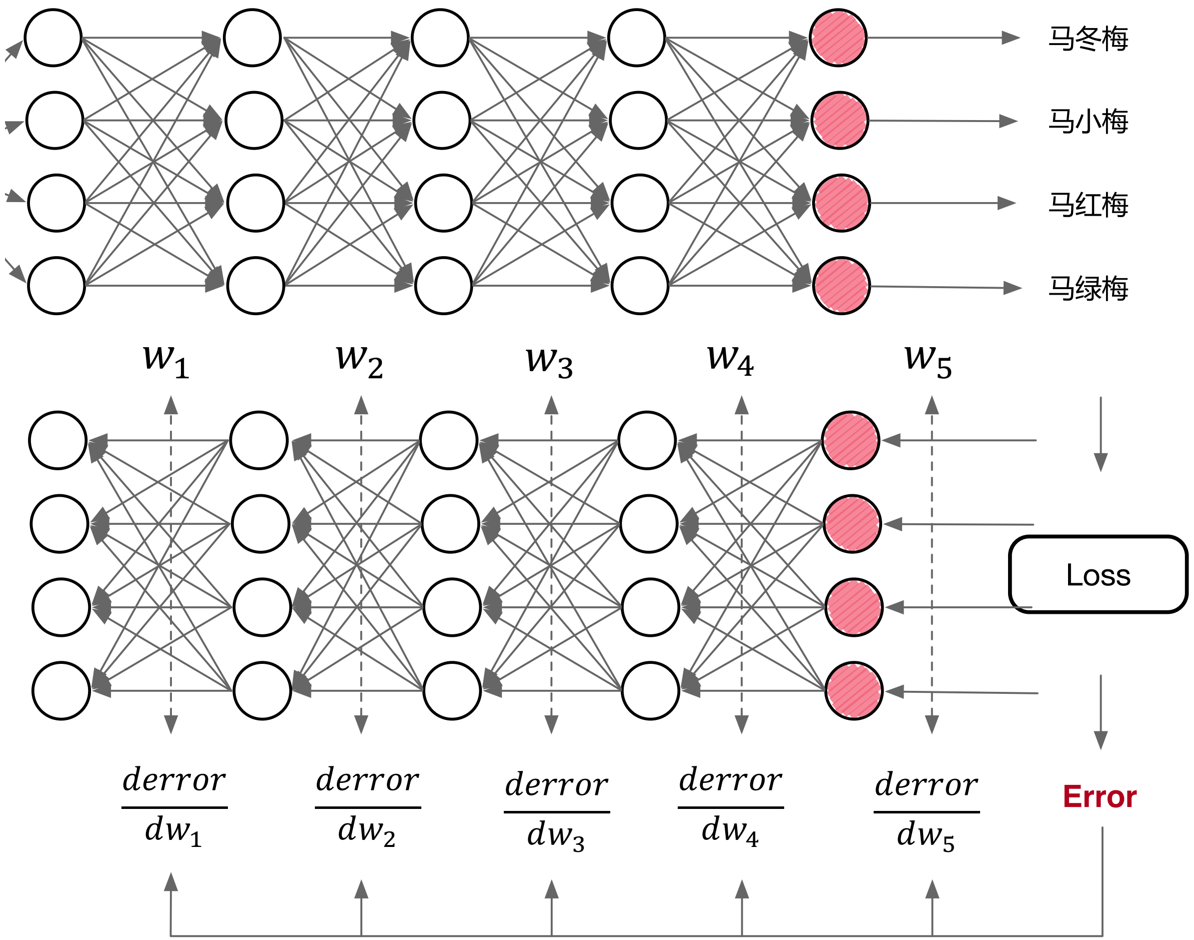
- **自动微分**：原子操作构成的复杂前向计算程序，关注自动生成高效的反向计算程序

自动微分

```python
class LeNet(nn.Module):

def __init__(self):
    super(LeNet, self).__init__()
    self.conv1 = nn.Conv2d(1, 6, 5, padding=2)
    self.conv2 = nn.Conv2d(6, 16, 5)
    self.fc1 = nn.Linear(16*5*5, 120)
    self.fc2 = nn.Linear(120, 84)
    self.fc3 = nn.Linear(84, 10)

def forward(self, x):
    x = F.max_pool2d(F.relu(self.conv1(x)), (2, 2))
    x = F.max_pool2d(F.relu(self.conv2(x)), (2, 2))
    x = x.view(-1, self.num_flat_features(x))
    x = F.relu(self.fc1(x))
    x = F.relu(self.fc2(x))
    x = self.fc3(x)
    return x
```



符号微分（ Symbolic Differentiation ）

符号微分：通过求导法则和导数变换公式，精确计算函数的导数

- 将原表达式转换为导数表达式：

$$\frac{d}{\partial x}(f(x)+g(x)) \rightarrow \frac{d}{\partial x}f(x)+\frac{d}{\partial x}g(x)$$

$$\frac{d}{\partial x}(f(x)\cdot g(x)) \rightarrow \left(\frac{d}{\partial x}f(x)\right)g(x)+f(x)\left(\frac{d}{\partial x}g(x)\right)$$

优势

- 精确数值结果

缺点

- 表达式膨胀

深度学习中的应用问题

- 深度学习网络非常大 -> 待求导函数复杂 -> 难以高效的求解
- 部分算子无法求导：如 Relu, Switch 等

数值微分（Numerical Differentiation）

数值微分：使用有限差分进行近似导数

- 可以使用有限差分来近似：

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x+he_i)-f(x)}{h}, where\ h>0$$

优势

- 容易实现

缺点

- 计算结果不精确
- 计算复杂度高
- 对 h 的要求高

深度学习中的应用问题

- 数值计算中的截断和近似问题导致无法得到精确导数
- 部分算子无法求导：如 Relu, Switch 等

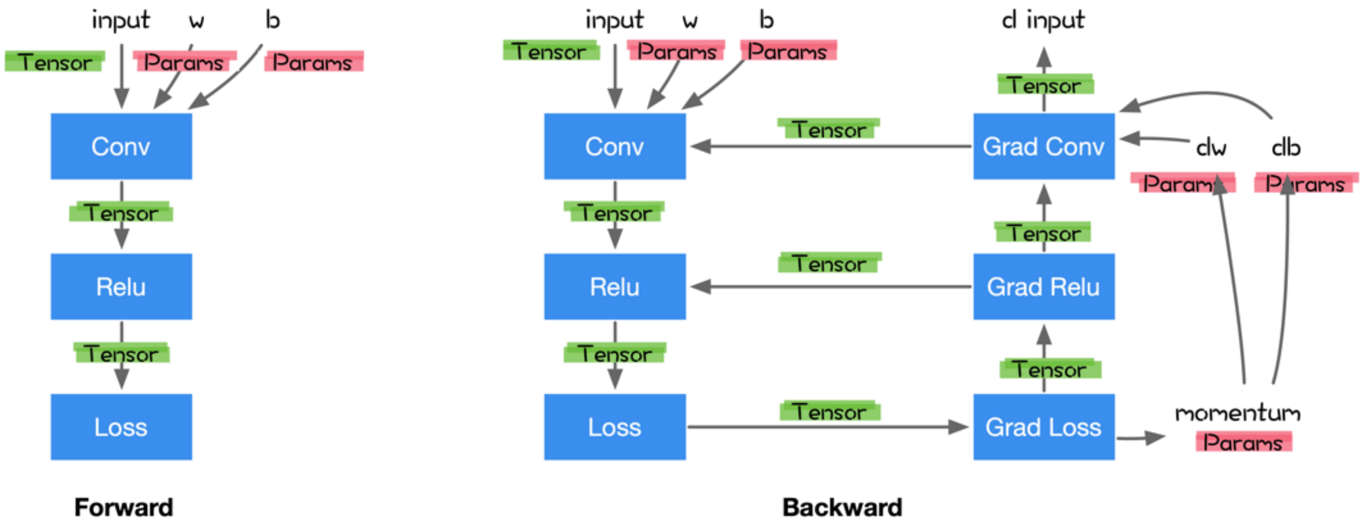
自动微分（Auto Differentiation）I

自动微分：所有数值计算都由有限的基本运算组成

基本运算的导数表达式是已知的

通过链式法则将数值计算各部分组合成整体

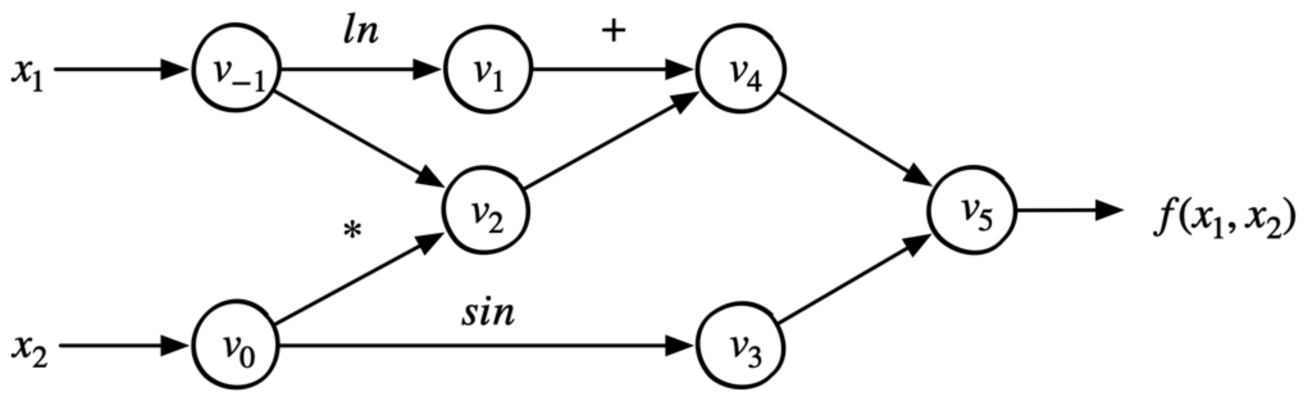
表达式追踪（Evaluation Trace）：追踪数值计算过程的<u>中间变量</u>



自动微分（Auto Differentiation）II

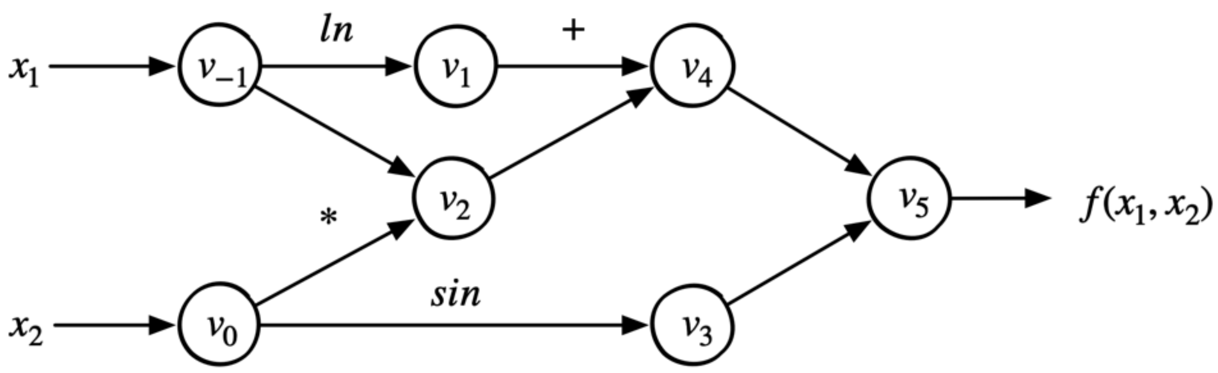
- 引入中间变量将一个复杂的函数，分解成一系列基本函数
- 将这些基本函数构成一个计算流图

$$f(x_1, x_2) = \ln(x_1) + x_1 x_2 - \sin(x_2)$$



自动微分（Auto Differentiation ）(II)

- 引入中间变量将一个复杂的函数，分解成一系列基本函数
- 将这些基本函数构成一个计算流图

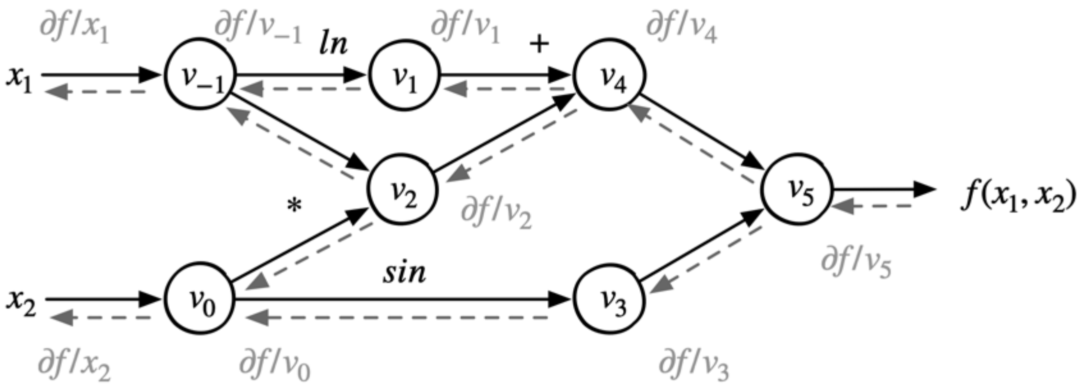


Forward Primal Trace

v_{-1}	$= x_1$	$= 2$
v_0	$= x_2$	$= 5$
v_1	$= \ln v_{-1}$	$= \ln 2$
v_2	$= v_{-1} \times v_0$	$= 2 \times 5$
v_3	$= \sin v_0$	$= \sin 5$
v_4	$= v_1 + v_2$	$= 0.693 + 10$
v_5	$= v_4 - v_3$	$= 10.693 + 0.959$
y	$= v_5$	$= 11.652$

自动微分（Auto Differentiation ）(II)

- 引入中间变量将一个复杂的函数，分解成一系列基本函数
- 将这些基本函数构成一个计算流图



Reverse Adjoint (Derivative) Trace

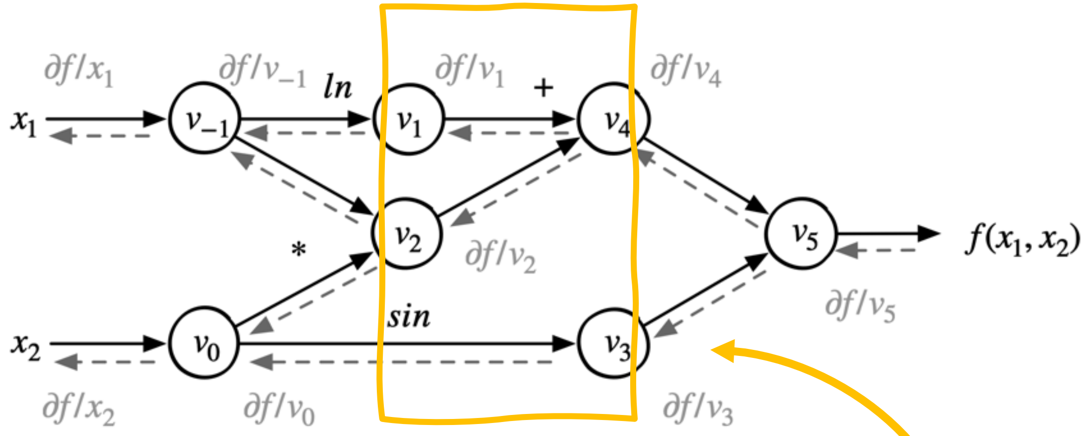
$$
\begin{array}{llll}
\boldsymbol{\bar{x}_1} & \boldsymbol{= \bar{v}_{-1}} & & \boldsymbol{= 5.5} \\
\boldsymbol{\bar{x}_2} & \boldsymbol{= \bar{v}_0} & & \boldsymbol{= 1.716} \\
\hline
\bar{v}_{-1} & = \bar{v}_{-1} + \bar{v}_1 \frac{\partial v_1}{\partial v_{-1}} & = \bar{v}_{-1} + \bar{v}_1 / v_{-1}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

自动微分（Auto Differentiation ）(III)

$$\boldsymbol{Y} = G(\boldsymbol{X})$$

$$\boldsymbol{J}_f = \begin{bmatrix} \frac{\partial \boldsymbol{y}}{\partial x_1} & \cdots & \frac{\partial \boldsymbol{y}}{\partial x_1} \end{bmatrix} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

雅克比矩阵 J



$$\boldsymbol{J}^{\mathrm{T}} \cdot \vec{\boldsymbol{v}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial l}{\partial y_1} \\ \vdots \\ \frac{\partial l}{\partial y_m} \end{bmatrix} = \begin{bmatrix} \frac{\partial l}{\partial x_1} \\ \vdots \\ \frac{\partial l}{\partial x_n} \end{bmatrix}$$

vector-Jacobian的乘积

$$\vec{\boldsymbol{v}} = \begin{bmatrix} \frac{\partial l}{\partial y_1} & \cdots & \frac{\partial l}{\partial y_m} \end{bmatrix}^{\mathrm{T}}$$

后一层损失函数对当前层输出的导数

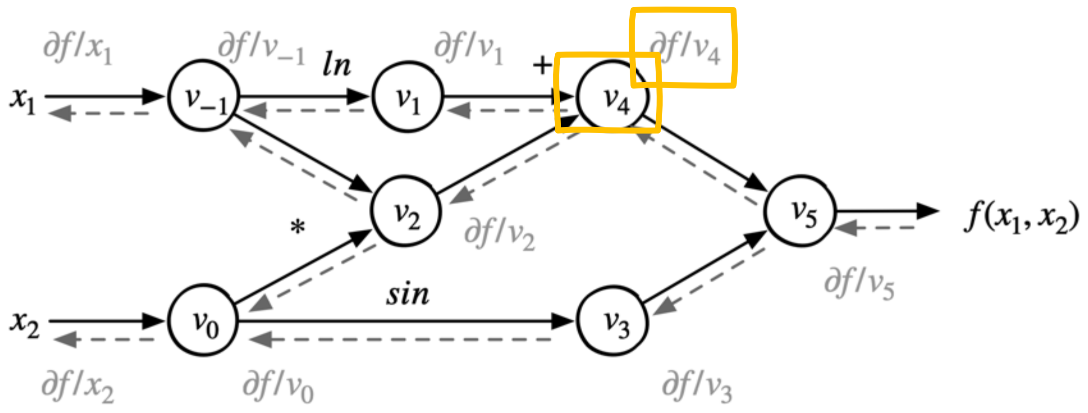
自动微分（Auto Differentiation ）(III)

注册前向计算结点和反向计算结点

前向结点接受输入计算输出

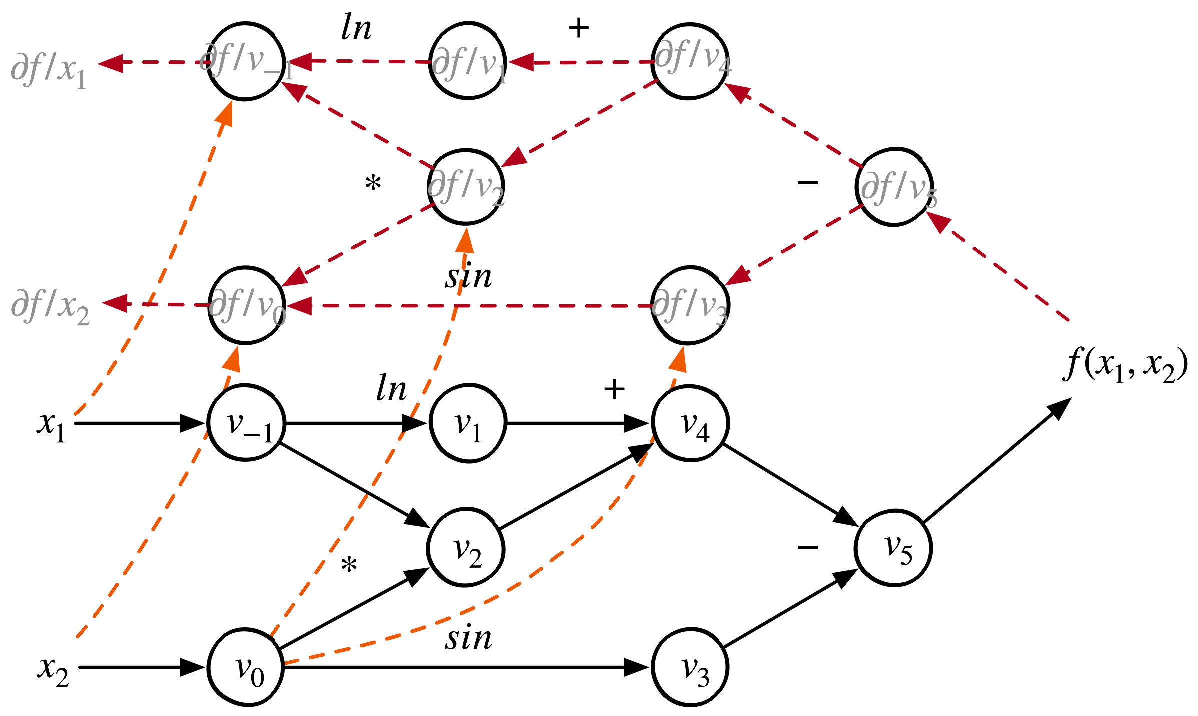
反向结点接受损失函数对当前张量操作输出的梯度 v

计算当前张量操作每个输入的vector-Jacobian乘积



自动微分（Auto Differentiation ）(III)

有向无环图（DAG, Directed Acyclic Graph）



Reverse Adjoint (Derivative) Trace

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

思考

- 正向模式 和 反向模式 计算量是否相等？
- AI框架或者深度学习任务中为什么大多使用反向模式？

在深度学习框架中实现自动微分（I）

前向计算并保留中间计算结果

根据反向模式的原理依次计算出中间导数

表达式追踪（Evaluation Trace）：追踪数值计算过程的中间变量

主要问题：

- 需要保存大量中间计算结果
- 方便跟踪计算过程

PyTorch

Forward Primal Trace		
$v_{-1} = x_1$		$= 2$
$v_0 = x_2$		$= 5$
$v_1 = \ln v_{-1}$		$= \ln 2$
$v_2 = v_{-1} \times v_0$		$= 2 \times 5$
$v_3 = \sin v_0$		$= \sin 5$
$v_4 = v_1 + v_2$		$= 0.693 + 10$
$v_5 = v_4 - v_3$		$= 10.693 + 0.959$
$y = v_5$		$= 11.652$

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

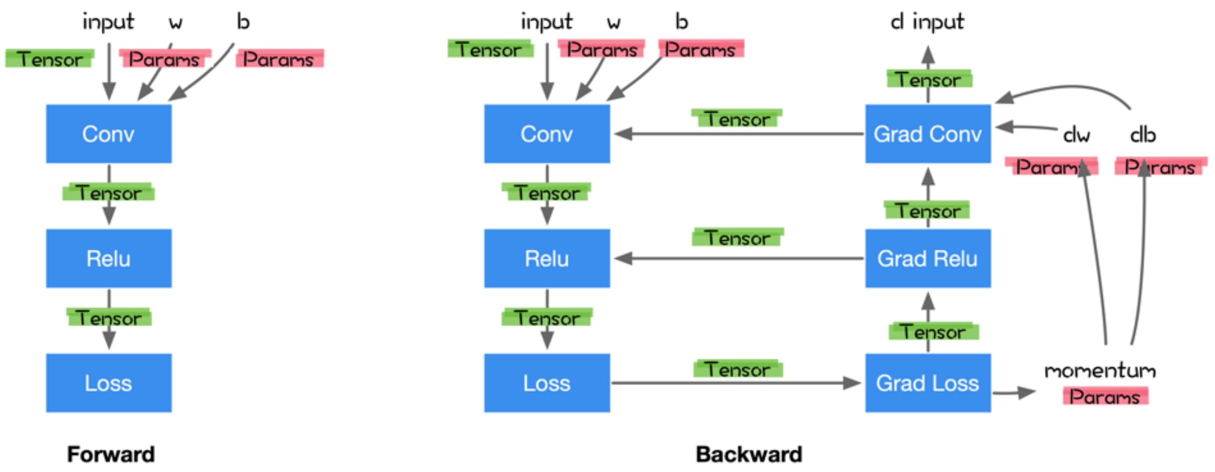
在深度学习框架中实现自动微分（II）

将导数的计算也表示成计算图

通过 Graph IR 来对计算图进行统一表示

主要特点：

- 不便于调试跟踪计算和数学表达过程
- 方便全局图优化
- 节省内存

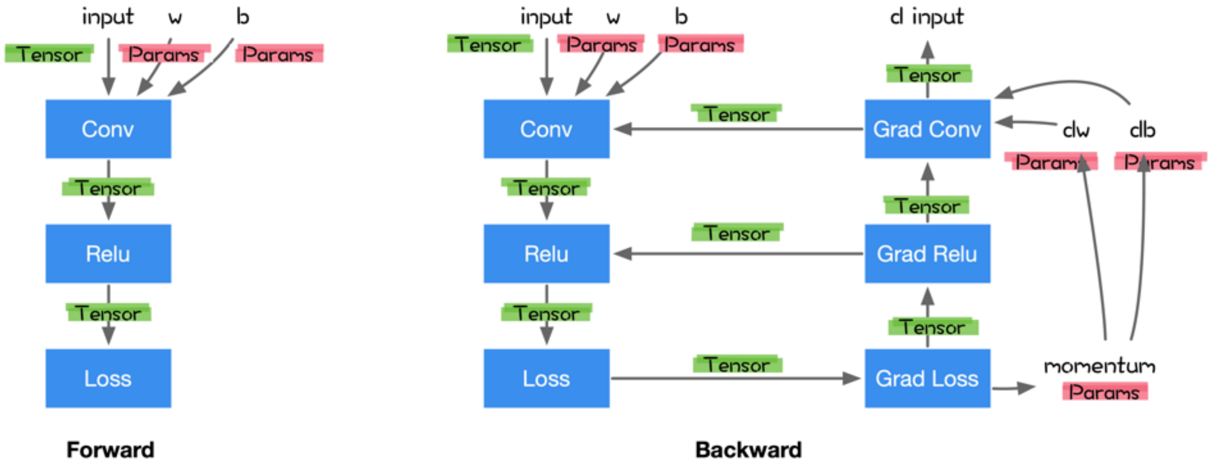


MindSpore TensorFlow

在深度学习框架中实现自动微分（III）

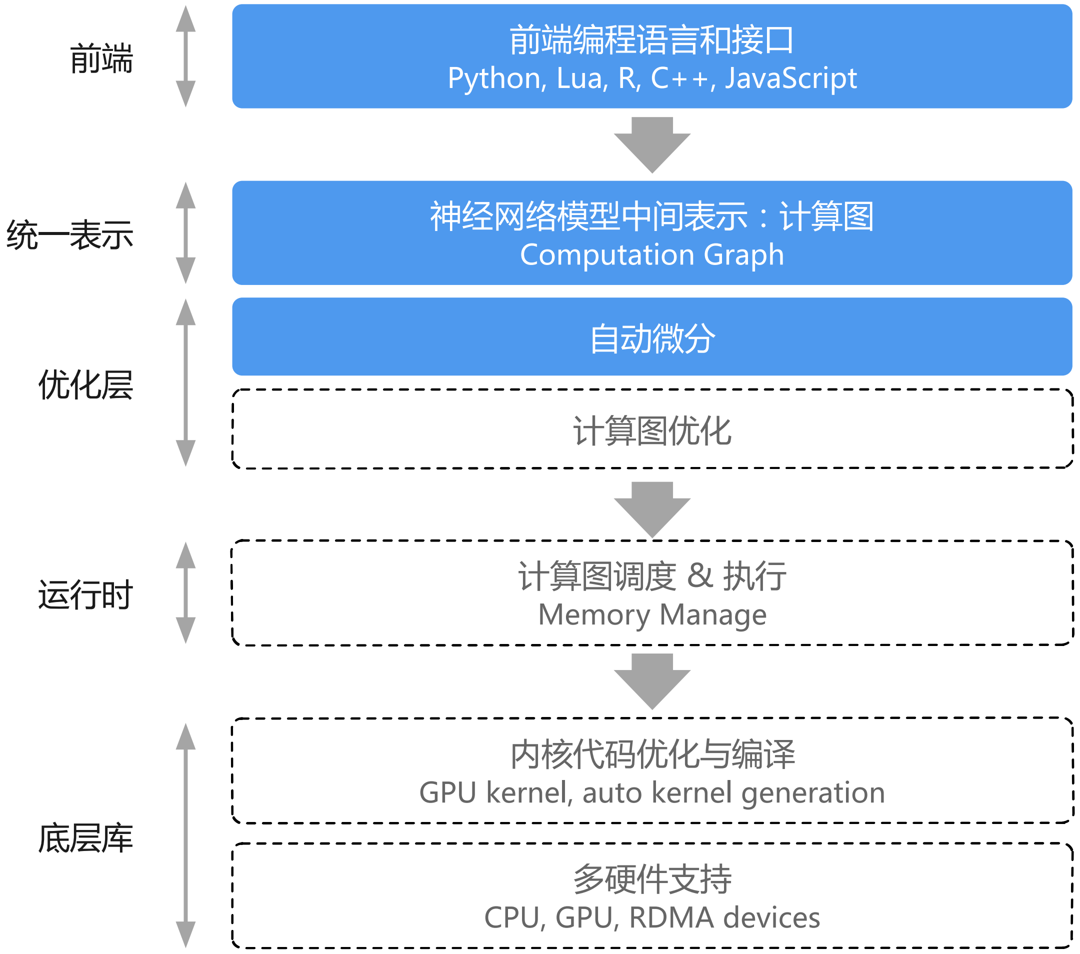
优化Pass：

- 给定前向数据流图
- 以损失函数为根节点广度优先遍历前向数据流图
- 按照对偶结构自动生成出求导数据流图



Review

- **模型表示**：计算图
- **前端语言**：用来构建计算图
- **自动微分**：基于反向模式的原理，构建计算图



Summary

1. 了解神经网络/AI系统中训练流程跟微分之间的关系
2. 回顾自动微分的正反向模式和计算图中的自动微分
3. 了解自动微分在深度学习中的一个实现表示

BUILDING A BETTER CONNECTED WORLD

THANK YOU