Deep Learning I

5. 深度学习计算

WU Xiaokun 吴晓堃

xkun.wu [at] gmail

2021/02/21

数值稳定性

神经网络的梯度

考虑一个L层的神经网络

$$\mathbf{h}^t = f_t(\mathbf{h}^{t-1})$$
$$y = \ell \circ f_L \circ \cdots \circ f_1(\mathbf{x})$$

神经网络的梯度

考虑一个L层的神经网络

$$\mathbf{h}^t = f_t(\mathbf{h}^{t-1})$$
$$y = l \circ f_L \circ \cdots \circ f_1(\mathbf{x})$$

计算梯度

$$\frac{\partial l}{\partial \mathbf{W}^t} = \frac{\partial l}{\partial \mathbf{h}^L} \frac{\partial \mathbf{h}^L}{\partial \mathbf{h}^{L-1}} \cdots \frac{\partial \mathbf{h}^{t+1}}{\partial \mathbf{h}^t} \frac{\partial \mathbf{h}^t}{\partial \mathbf{W}^t}$$
$$= \frac{\partial l}{\partial \mathbf{h}^L} \prod_{i=t+1}^{L} \frac{\partial \mathbf{h}^i}{\partial \mathbf{h}^{i-1}} \frac{\partial \mathbf{h}^t}{\partial \mathbf{W}^t}$$

- 总共$L - t + 1$次矩阵乘法

数值稳定性的两个问题

梯度爆炸

- $1.5^{100} \approx 4 \times 10^{17}$

梯度消失

- $0.8^{100} \approx 2 \times 10^{-10}$

例如MLP

线性输出并激活

$$f_t(\mathbf{h}^{t-1}) = \sigma(\mathbf{W}^t\mathbf{h}^{t-1})$$

$$\frac{\partial \mathbf{h}^t}{\partial \mathbf{h}^{t-1}} = \mathrm{diag}\left(\sigma'(\mathbf{W}^t\mathbf{h}^{t-1})\right)(\mathbf{W}^t)^T$$

例如MLP

线性输出并激活

$$f_t(\mathbf{h}^{t-1}) = \sigma(\mathbf{W}^t \mathbf{h}^{t-1})$$
$$\frac{\partial \mathbf{h}^t}{\partial \mathbf{h}^{t-1}} = \operatorname{diag}\left(\sigma'(\mathbf{W}^t \mathbf{h}^{t-1})\right)(\mathbf{W}^t)^T$$

链式法则计算梯度

$$\prod_{i=t+1}^{L} \frac{\partial \mathbf{h}^i}{\partial \mathbf{h}^{i-1}} = \prod_{i=t+1}^{L} \operatorname{diag}\left(\sigma'(\mathbf{W}^i \mathbf{h}^{i-1})\right)(\mathbf{W}^i)^T$$

MLP梯度爆炸

假设使用最常用的`ReLU`激活函数

$$\sigma(x) = \max(0, x), \sigma'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

MLP梯度爆炸

假设使用最常用的`ReLU`激活函数

$$\sigma(x) = \max(0, x), \sigma'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

代入MLP的梯度计算公式

$$\begin{aligned} \prod_{i=t+1}^{L} \frac{\partial \mathbf{h}^i}{\partial \mathbf{h}^{i-1}} &= \prod_{i=t+1}^{L} \operatorname{diag}\left(\sigma'(\mathbf{W}^i \mathbf{h}^{i-1})\right) (\mathbf{W}^i)^T \\ &= \prod_{i=t+1}^{L} (\mathbf{W}^i)^T \end{aligned}$$

- 如果层数比较多，可能导致上溢

梯度爆炸本质是上溢

上溢：梯度值超出浮点数值域

- 对16位浮点数尤为严重

梯度爆炸本质是上溢

上溢：梯度值超出**浮点数值域**

- 对16位浮点数尤为严重

导致**对学习率敏感**

- 学习率稍大：参数值大 -> 梯度变得更大，正向反馈
- 限制学习率：控制更新幅度容易过度，训练不进展

梯度爆炸本质是上溢

上溢：梯度值超出**浮点数值域**

- 对16位浮点数尤为严重

导致**对学习率敏感**

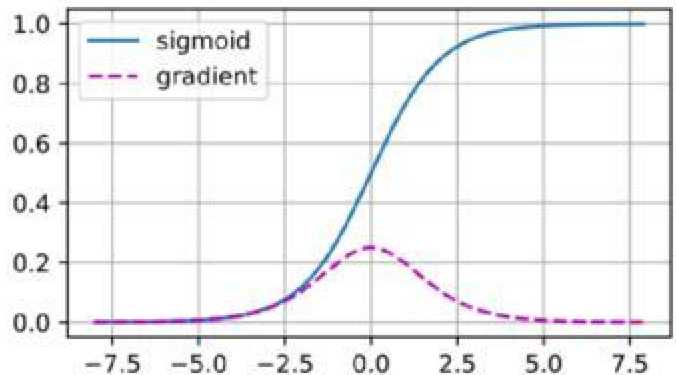
- 学习率稍大：参数值大 -> 梯度变得更大，正向反馈
- 限制学习率：控制更新幅度容易过度，训练不进展

根据实际情况：可能需要在训练过程中**动态调整**学习率

MLP梯度消失

假设使用`sigmoid`作为激活函数：导数值很小

$$\sigma(x) = \frac{1}{1 + \exp^{-x}}, \sigma'(x) = \sigma(x)(1 - \sigma(x))$$



≡

$$\prod_{i=t+1}^{L} \frac{\partial \mathbf{h}^{i}}{\partial \mathbf{h}^{i-1}} = \prod_{i=t+1}^{L} \operatorname{diag}\left(\sigma'\left(\mathbf{W}^{i}\mathbf{h}^{i-1}\right)\right)\left(\mathbf{W}^{i}\right)^{T}$$

梯度消失本质是下溢

下溢：梯度值变成0

- 对16位浮点数尤为严重

梯度消失本质是下溢

下溢：梯度值变成0

- 对16位浮点数尤为严重

训练不进展：损失值不下降

- 不管学习率如何选取

梯度消失本质是下溢

下溢：梯度值变成0

- 对16位浮点数尤为严重

训练不进展：损失值不下降

- 不管学习率如何选取

由于反向传递：对底部层尤为严重，**只能训练顶部层**

- 无法把神经网络做深：**模型容量不足**以解决问题

实验：数值稳定性

模型初始化

让训练更加稳定

目标：将梯度值控制在**合理范围**

- 常用数学技巧：取对数转换，乘法变加法
- 高级架构：ResNet, LSTM

让训练更加稳定

目标：将梯度值控制在**合理范围**

- 常用数学技巧：取对数转换，乘法变加法
- 高级架构：ResNet, LSTM

- 归一化：梯度归一化，梯度剪裁
  - 数值较小时优化更稳定

让训练更加稳定

目标：将梯度值控制在**合理范围**

- 常用数学技巧：取对数转换，乘法变加法
- 高级架构：ResNet, LSTM

- 归一化：梯度归一化，梯度剪裁
  - 数值较小时优化更稳定

- 合理的权重初始化、激活函数

控制变量的统计量

将每层的输出和梯度看成**随机变量**

- 控制一阶**统计量保持一致**：均值、方差

控制变量的统计量

将每层的输出和梯度看成**随机变量**

- 控制一阶**统计量保持一致**：均值、方差

正向积累

$$\mathbb{E}[h_i^t] = 0$$
$$Var[h_i^t] = a$$

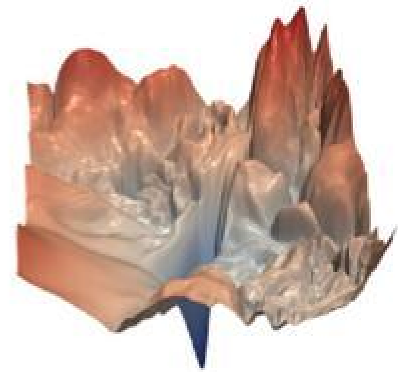
反向传递

$$\mathbb{E}[\frac{\partial l}{\partial h_i^t}] = 0$$
$$Var[\frac{\partial l}{\partial h_i^t}] = b$$

权重初始化

优化：想象成在地形图上撒小球

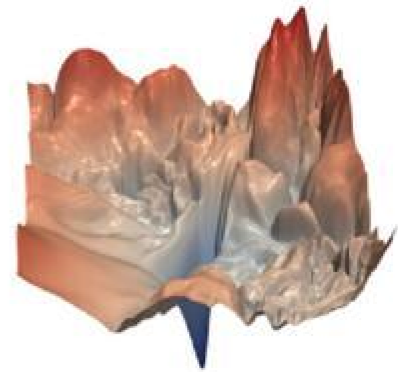
- 依靠重力滚向最低点
- 复杂地形：**初始位置**非常关键



权重初始化

优化：想象成在地形图上撒小球

- 依靠重力滚向最低点
- 复杂地形：**初始位置**非常关键



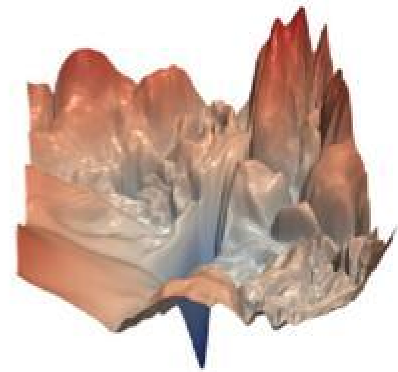
参数初始值等价于初始位置：应该控制在合理范围

- 训练开始时数值不稳定问题最严重
  - 主要是因为**远离最优解**；最优解附近通常地形相对平缓

权重初始化

优化：想象成在地形图上撒小球

- 依靠重力滚向最低点
- 复杂地形：**初始位置**非常关键



参数初始值等价于初始位置：应该控制在合理范围

- 训练开始时数值不稳定问题最严重
  - 主要是因为**远离最优解**；最优解附近通常地形相对平缓

之前使用正态分布初始化：不适合复杂神经网络

MLP 正向期望

基本假设

- $w^t_{i,j}$是 **独立同分布 i.i.d.**：$\mathbb{E}[w^t_{i,j}] = 0, Var[w^t_{i,j}] = \gamma_t$
- h^{t-1}_i独立于$w^t_{i,j}$：计算梯度时相当于常数

MLP 正向期望

基本假设

- $w_{i,j}^t$是 **独立同分布 i.i.d.**：$\mathbb{E}[w_{i,j}^t] = 0, Var[w_{i,j}^t] = \gamma_t$
- h_i^{t-1}独立于$w_{i,j}^t$：计算梯度时相当于常数

首先假设没有激活函数：$\mathbf{h}^t = \mathbf{W}^t\mathbf{h}^{t-1}, \mathbf{W}^t \in \mathbb{R}^{n_t \times n_{t-1}}$

$$\begin{aligned}\mathbb{E}[h_i^t] &= \mathbb{E}\left[\sum_j w_{i,j}^t h_j^{t-1}\right] \\ &= \sum_j \mathbb{E}[w_{i,j}^t]\mathbb{E}[h_j^{t-1}] \\ &= 0\end{aligned}$$

MLP 正向方差

$$
\begin{aligned}
Var[h_i^t] &= \mathbb{E}[(h_i^t)^2] - \mathbb{E}[h_i^t]^2 \\
&= \mathbb{E}\left[\left(\sum_j w_{i,j}^t h_j^{t-1}\right)^2\right] \\
&= \mathbb{E}\left[\sum_j \left(w_{i,j}^t\right)^2 \left(h_j^{t-1}\right)^2 + \sum_{j \neq k} w_{i,j}^t w_{i,k}^t h_j^{t-1} h_k^{t-1}\right] \\
&= \sum_j \mathbb{E}\left[\left(w_{i,j}^t\right)^2\right] \mathbb{E}\left[\left(h_j^{t-1}\right)^2\right] \\
&= \sum_j Var[w_{i,j}^t] Var[h_j^{t-1}] \\
&= n_{t-1} \gamma_t Var[h_j^{t-1}]
\end{aligned}
$$

MLP 正向方差：保持不变

$$Var[h_i^t] = n_{t-1}\gamma_t Var[h_j^{t-1}]$$

- 只有当$n_{t-1}\gamma_t = 1$时：$Var[h_i^t] = Var[h_j^{t-1}]$

MLP 反向均值、方差

$$\mathbf{h}^t = \mathbf{W}^t \mathbf{h}^{t-1} \Rightarrow \frac{\partial l}{\partial \mathbf{h}^{t-1}} = \frac{\partial l}{\partial \mathbf{h}^t} \mathbf{W}^t$$

$$\Rightarrow \left(\frac{\partial l}{\partial \mathbf{h}^{t-1}} \right)^T = (\mathbf{W}^t)^T \left(\frac{\partial l}{\partial \mathbf{h}^t} \right)^T$$

MLP 反向均值、方差

$$\mathbf{h}^t = \mathbf{W}^t \mathbf{h}^{t-1} \Rightarrow \frac{\partial l}{\partial \mathbf{h}^{t-1}} = \frac{\partial l}{\partial \mathbf{h}^t} \mathbf{W}^t$$
$$\Rightarrow \left(\frac{\partial l}{\partial \mathbf{h}^{t-1}}\right)^T = (\mathbf{W}^t)^T \left(\frac{\partial l}{\partial \mathbf{h}^t}\right)^T$$

略去中间步骤得到

$$\mathbb{E}\left[\frac{\partial l}{\partial h_t^{t-1}}\right] = 0$$
$$Var\left[\frac{\partial l}{\partial h_t^{t-1}}\right] = n_t \gamma_t Var\left[\frac{\partial l}{\partial h_t^t}\right]$$

MLP 反向均值、方差

$$\mathbf{h}^t = \mathbf{W}^t \mathbf{h}^{t-1} \Rightarrow \frac{\partial l}{\partial \mathbf{h}^{t-1}} = \frac{\partial l}{\partial \mathbf{h}^t} \mathbf{W}^t$$
$$\Rightarrow \left(\frac{\partial l}{\partial \mathbf{h}^{t-1}}\right)^T = (\mathbf{W}^t)^T \left(\frac{\partial l}{\partial \mathbf{h}^t}\right)^T$$

略去中间步骤得到

$$\mathbb{E}\left[\frac{\partial l}{\partial h_t^{t-1}}\right] = 0$$
$$Var\left[\frac{\partial l}{\partial h_t^{t-1}}\right] = n_t \gamma_t Var\left[\frac{\partial l}{\partial h_t^t}\right]$$

- 只有当$n_t \gamma_t = 1$时：$Var\left[\frac{\partial l}{\partial h_t^{t-1}}\right] = Var\left[\frac{\partial l}{\partial h_t^t}\right]$

Xavier 初始化

结论：$n_{t-1}\gamma_t = 1, n_t\gamma_t = 1$作为初始化的公式

- 限制条件太严格：只能$n_{t-1} = n_t = \frac{1}{\gamma_t}$

Xavier 初始化

结论：$n_{t-1}\gamma_t = 1, n_t\gamma_t = 1$作为初始化的公式

- 限制条件太严格：只能$n_{t-1} = n_t = \frac{1}{\gamma_t}$
- Xavier提出取平均值的方法：$(n_{t-1} + n_t)\gamma_t/2 = 1 \Rightarrow \gamma_t = 2/(n_{t-1} + n_t)$

Xavier 初始化

结论：$n_{t-1}\gamma_t = 1, n_t\gamma_t = 1$作为初始化的公式

- 限制条件太严格：只能$n_{t-1} = n_t = \frac{1}{\gamma_t}$
- Xavier提出取平均值的方法：$(n_{t-1} + n_t)\gamma_t/2 = 1 \Rightarrow \gamma_t = 2/(n_{t-1} + n_t)$

按照定义：$Var[w_{i,j}^t] = \gamma_t$

- 正态分布：$\mathcal{N}(0, \sqrt{2/(n_{t-1} + n_t)})$
- 均匀分布：$\mathcal{U}(-\sqrt{6/(n_{t-1} + n_t)}, \sqrt{6/(n_{t-1} + n_t)})$
  - 分布$\mathcal{U}(-a, a)$的方差：$a^2/3$

线性激活函数

讨论完无激活函数后，其次假设 $\sigma(x) = ax + b$

- $\mathbf{h}' = \mathbf{W}^t \mathbf{h}^{t-1}, \mathbf{h} = \sigma(\mathbf{h}')$
- $\frac{\partial l}{\partial \mathbf{h}'} = \frac{\partial l}{\partial \mathbf{h}^t} (\mathbf{W}^t)^T, \frac{\partial l}{\partial \mathbf{h}^{t-1}} = a \frac{\partial l}{\partial \mathbf{h}'}$

线性激活函数

讨论完无激活函数后，其次假设 $\sigma(x) = ax + b$

- $\mathbf{h}' = \mathbf{W}^t \mathbf{h}^{t-1}, \mathbf{h} = \sigma(\mathbf{h}')$
- $\frac{\partial l}{\partial \mathbf{h}'} = \frac{\partial l}{\partial \mathbf{h}^t} (\mathbf{W}^t)^T, \frac{\partial l}{\partial \mathbf{h}^{t-1}} = a \frac{\partial l}{\partial \mathbf{h}'}$

正向积累

- 均值：$\mathbb{E}[h_i^t] = b \Rightarrow b = 0$
- 方差：$Var[h_i^t] = a^2 Var[h_i'] \Rightarrow a = 1$

线性激活函数

讨论完无激活函数后，其次假设 $\sigma(x) = ax + b$

- $\mathbf{h}' = \mathbf{W}^t \mathbf{h}^{t-1}, \mathbf{h} = \sigma(\mathbf{h}')$
- $\frac{\partial l}{\partial \mathbf{h}'} = \frac{\partial l}{\partial \mathbf{h}^t} (\mathbf{W}^t)^T, \frac{\partial l}{\partial \mathbf{h}^{t-1}} = a \frac{\partial l}{\partial \mathbf{h}'}$

正向积累

- 均值：$\mathbb{E}[h_i^t] = b \Rightarrow b = 0$
- 方差：$Var[h_i^t] = a^2 Var[h_i'] \Rightarrow a = 1$

反向传递

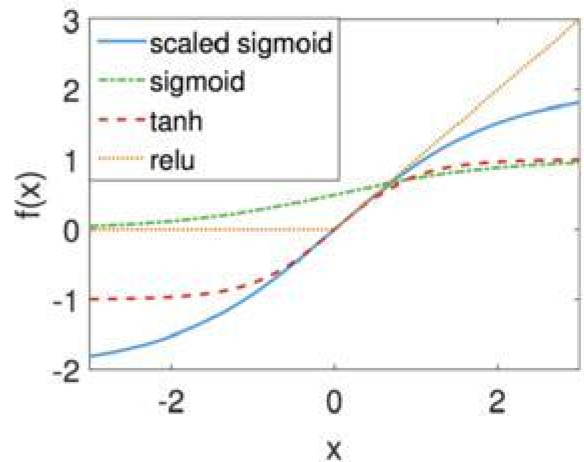
- 均值：$\mathbb{E}\left[\frac{\partial l}{\partial h_i^{t-1}}\right] = 0 \Rightarrow b = 0$
- 方差：$Var\left[\frac{\partial l}{\partial h_i^{t-1}}\right] = a^2 Var\left[\frac{\partial l}{\partial h_i'}\right] \Rightarrow a = 1$

常用激活函数

$$\text{sigmoid} = \frac{1}{2} + \frac{x}{4} - \frac{x^3}{48} + O(x^5)$$

$$\tanh = 0 + x - \frac{x^3}{3} + O(x^5)$$

$$\text{relu} = 0 + x, \text{for } x \geq 0$$

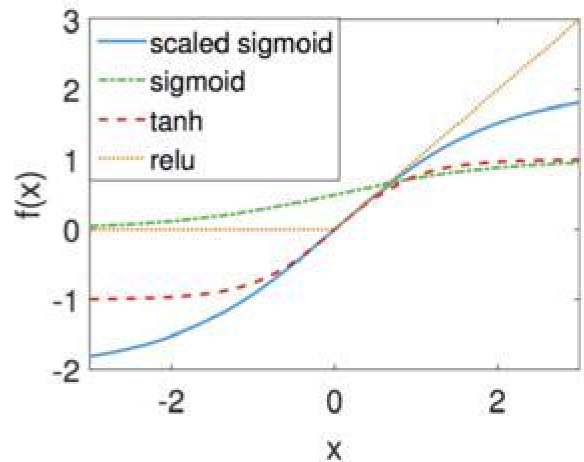


常用激活函数

$$\text{sigmoid} = \frac{1}{2} + \frac{x}{4} - \frac{x^3}{48} + O(x^5)$$

$$\tanh = 0 + x - \frac{x^3}{3} + O(x^5)$$

$$\text{relu} = 0 + x, \text{for } x \geq 0$$



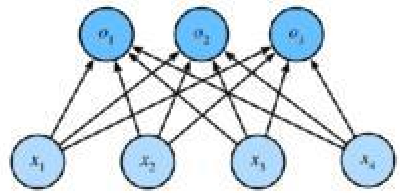
由线性激活的启发：可以调整`sigmoid`：$4 \times \text{sigmoid} - 2$

层和块

模型架构

线性模型的基本元素：

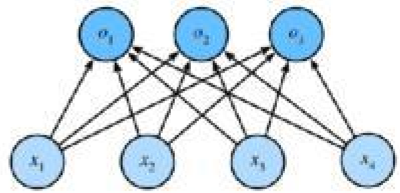
- 单个标量输出：（1）输入；（2）输出；（3）参数
- 多个概率输出：（1）输入；（2）输出；（3）参数



模型架构

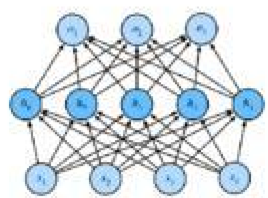
线性模型的基本元素：

- 单个标量输出：（1）输入；（2）输出；（3）参数
- 多个概率输出：（1）输入；（2）输出；（3）参数



多层感知机：每个层可以看成线性模型

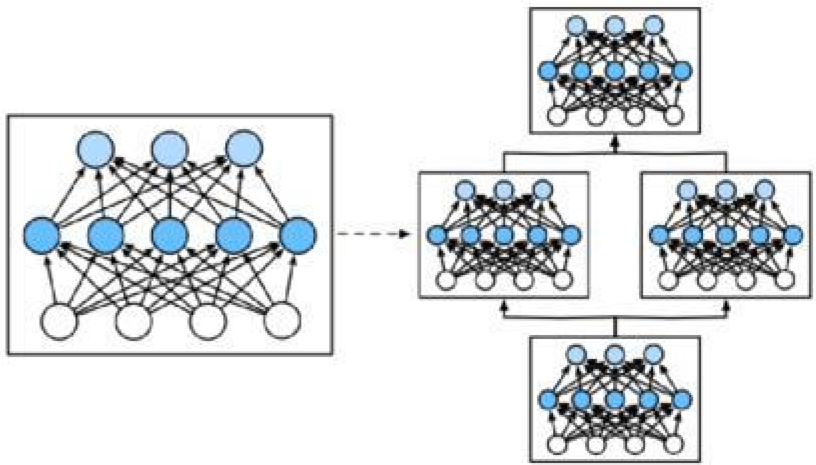
- （1）输入；（2）输出；（3）参数



块

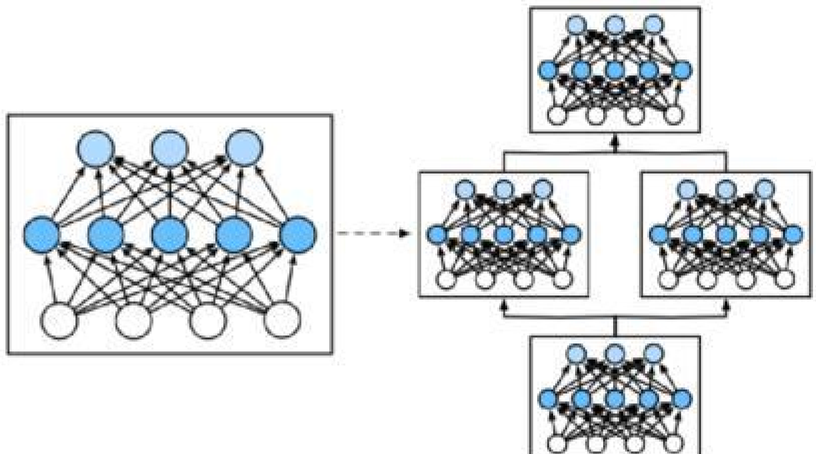
块 block可以描述单个层、由多个层组成的组件或整个模型本身

- 模型的构建是递归的



块的基本功能（需求）

1. 将**输入**数据作为其前向传播函数的参数
2. 通过前向传播函数来生成**输出**
3. 自动计算其输出关于输入的**梯度**，可通过其反向传播函数进行访问
4. **存储和访问**前向传播计算所需的参数
5. 根据需要**初始化**模型参数

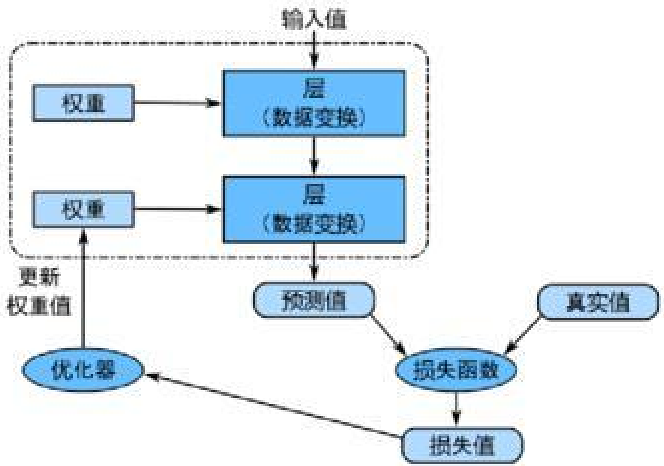


实验：模型构建

参数管理

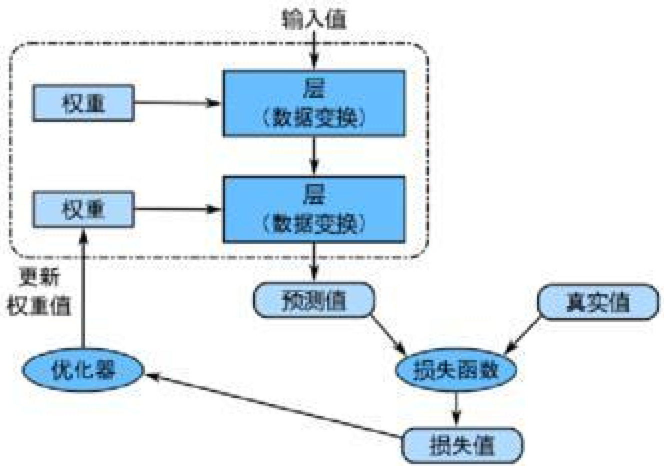
为什么要管理参数?

之前模型：参数由优化器**自动更新**



为什么要管理参数?

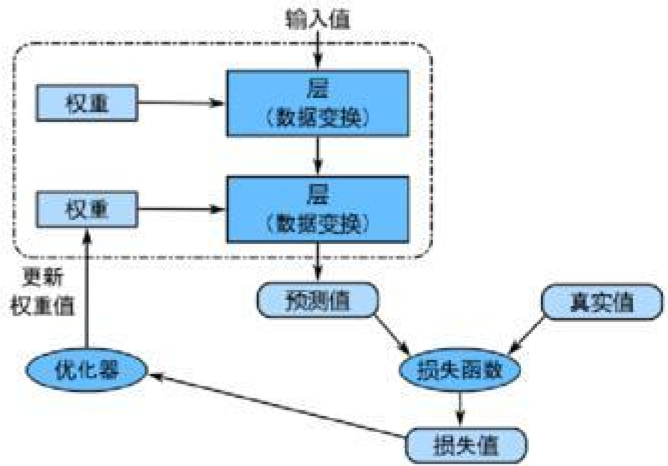
之前模型：参数由优化器**自动更新**



- 保存参数，并在不同模型组件间**共享参数**
  - 传入参数：参数初始化

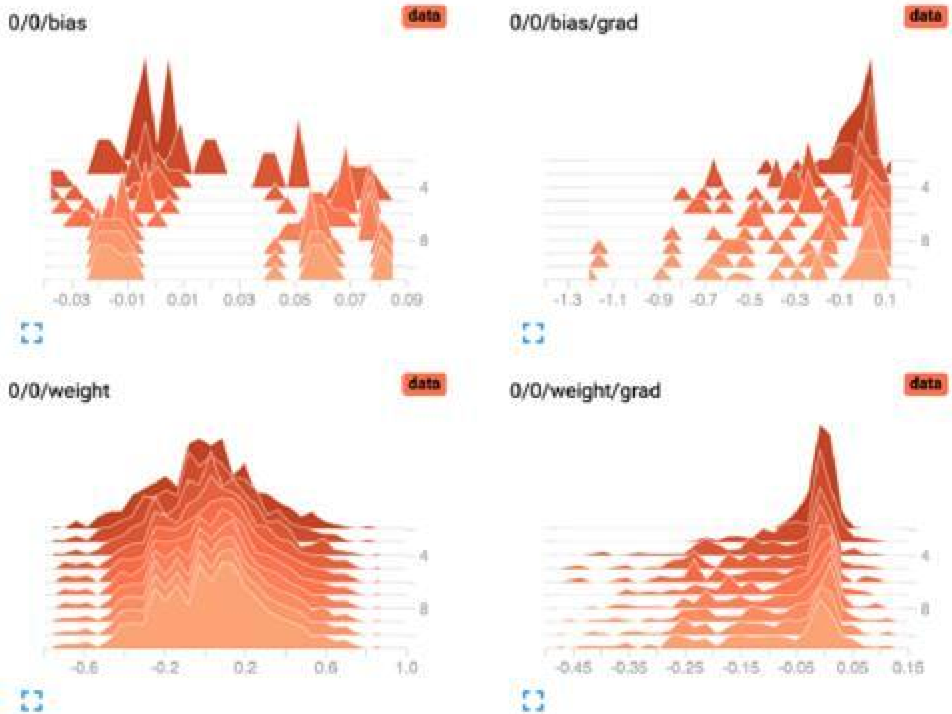
为什么要管理参数?

之前模型：参数由优化器**自动更新**



- 保存参数，并在不同模型组件间**共享参数**
  - 传入参数：参数初始化
- 取出参数：**调试**、**诊断**和可视化
  - 查看分布；判断过拟合（下页）

参数可视化



0/0/bias
data
-0.03
-0.01
0.01
0.03
0.05
0.07
0.09
4
8
0/0/bias/grad
data
-1.3
-1.1
-0.9
-0.7
-0.5
-0.3
-0.1
0.1
4
8
0/0/weight
data
-0.6
-0.2
0.2
0.6
1.0
4
8
0/0/weight/grad
data
-0.45
-0.35
-0.25
-0.15
-0.05
0.05
0.15
4
8

实验： 参数管理

自定义层

为什么需要自定义层?

设计出适用于各种任务的架构

- 模块**组合设计**：创造性的方式组合不同的层
- 应对不同的数据格式：图像、文本

为什么需要自定义层?

设计出适用于各种任务的架构

- 模块**组合设计**：创造性的方式组合不同的层
- 应对不同的数据格式：图像、文本

创新设计：发明一个在当前深度学习框架中还不存在的模块

实验： 自定义层

读写文件

什么情况下需要读写文件?

保存文件是良好习惯

- 服务器突发异常时：读取参数继续训练

什么情况下需要读写文件?

保存文件是良好习惯

- 服务器突发异常时：读取参数继续训练
- 模型过拟合：读取参数历史快照

什么情况下需要读写文件?

保存文件是良好习惯

- 服务器突发异常时：读取参数继续训练
- 模型过拟合：读取参数历史快照

迁移学习：将模型参数应用到不同任务

什么情况下需要读写文件?

保存文件是良好习惯

- 服务器突发异常时：读取参数继续训练
- 模型过拟合：读取参数历史快照

迁移学习：将模型参数**应用到不同任务**

部署应用：使用训练好的模型

- 注意：**仅保存模型参数**，而不是整个模型
  - 读取需要预知模型定义，并实例化

实验：读写文件

GPU

使用前提

个人电脑

- NVIDIA 显卡，且支持CUDA
- 安装并行计算的驱动程序和CUDA

使用前提

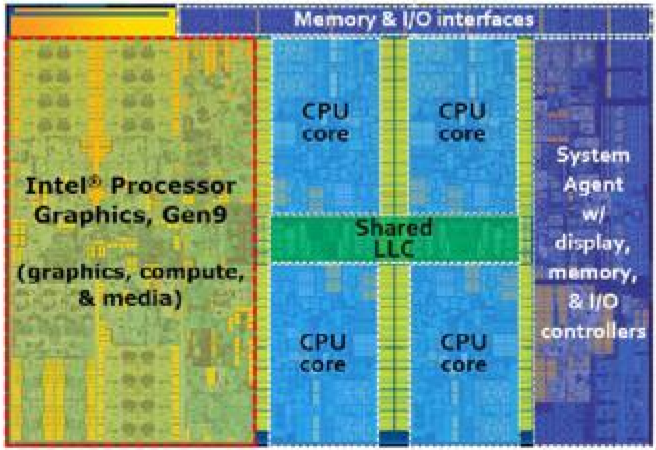
个人电脑

- NVIDIA 显卡，且支持CUDA
- 安装并行计算的驱动程序和CUDA

云计算服务：通常有自动化解决方案、客服

计算芯片

Intel i7-6700K：0.15 TFLOPS



- K：未锁频；无F：封装显卡

Nvidia GP104 (Pascal)：12 TFLOPS

- CUDA：只能用N卡
- OpenCL：取决于硬件厂商

计算性能对比

	CPU + 内存	GPU
#核	4 - 64	2k - 4k
TFLOPS	0.2 - 1	10 - 100
内存	32GB - 1TB	16GB - 32GB
内存带宽	30GB/s - 100 GB/s	400GB/s - 1TB/s
控制流	强	弱

计算性能对比

	CPU + 内存	GPU
#核	4 - 64	2k - 4k
TFLOPS	0.2 - 1	10 - 100
内存	32GB - 1TB	16GB - 32GB
内存带宽	30GB/s - 100 GB/s	400GB/s - 1TB/s
控制流	强	弱

- 内存层级：主存 -> L3-L1 -> 寄存器
  - 本地性：按块读取（例如64B），预读

计算性能对比

	CPU + 内存	GPU
#核	4 - 64	2k - 4k
TFLOPS	0.2 - 1	10 - 100
内存	32GB - 1TB	16GB - 32GB
内存带宽	30GB/s - 100 GB/s	400GB/s - 1TB/s
控制流	强	弱

- 内存层级：主存 -> L3-L1 -> 寄存器
  - 本地性：按块读取（例如64B），预读
- 超线程不一定提升性能：共享寄存器

计算设备

默认情况下，张量是**在内存中**创建，然后使用CPU计算

- `torch.device('cpu')`：所有物理CPU和内存

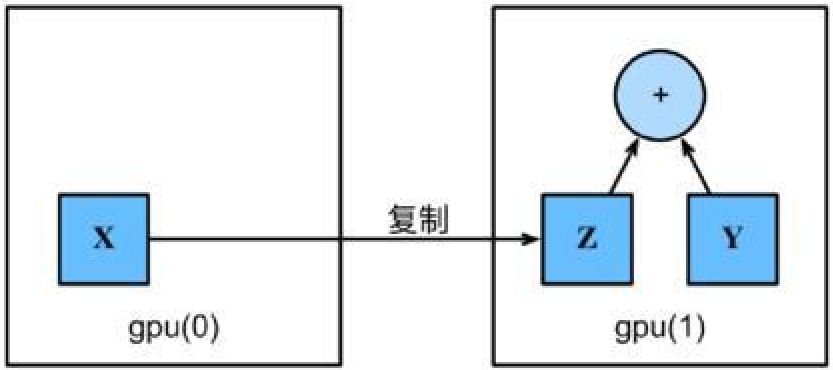
计算设备

默认情况下，张量是**在内存中**创建，然后使用CPU计算

- `torch.device('cpu')`：所有物理CPU和内存
- `torch.device('cuda')`：一个卡和相应的显存
  - `torch.device(f'cuda:{i}')`：第 i 块GPU（i 从0开始）
  - `cuda:0`和`cuda`是等价的

张量与计算

注意：操作必须在同一设备上！



GPU并行计算原则

少用控制语句：支持有限

- 实现：分支全部执行
- **计算同步**开销大：等待最慢的分支

GPU并行计算原则

少用控制语句：支持有限

- 实现：分支全部执行
- **计算同步**开销大：等待最慢的分支

尽量不要在设备（CPU、GPU和其他机器）之间传输数据

- 传输带宽差异导致瓶颈
- 等待**数据同步**
- 读取主存却没有副本时会复制

GPU并行计算原则

少用控制语句：支持有限

- 实现：分支全部执行
- **计算同步**开销大：等待最慢的分支

尽量不要在设备（CPU、GPU和其他机器）之间传输数据

- 传输带宽差异导致瓶颈
- 等待**数据同步**
- 读取主存却没有副本时会复制

如果必要，**合并操作**

- 利用并行性：上千个线程打包处理
- 连续、区块访问优于多次、小批量访问

TPU

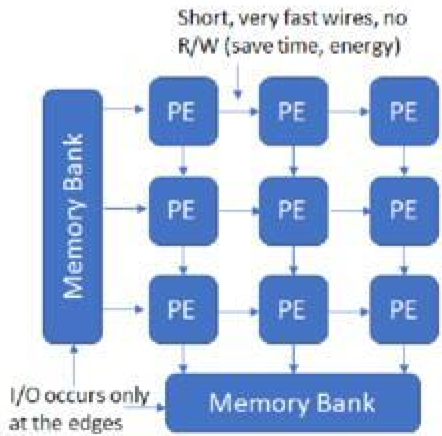
专用集成电路 Application-Specific Integrated Circuit (ASIC)

- 深度学习：**反向促进硬件研发**，大厂都在做

TPU

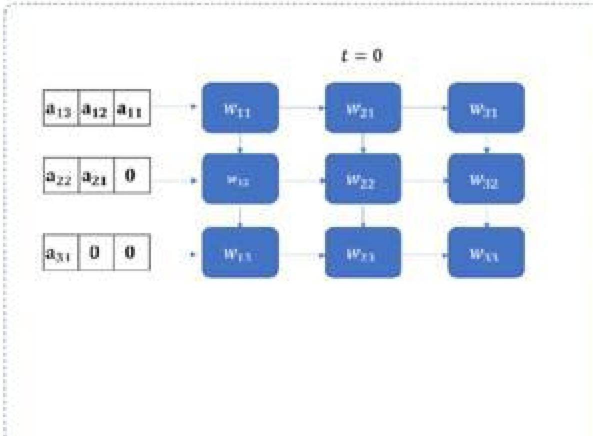
专用集成电路 Application-Specific Integrated Circuit (ASIC)

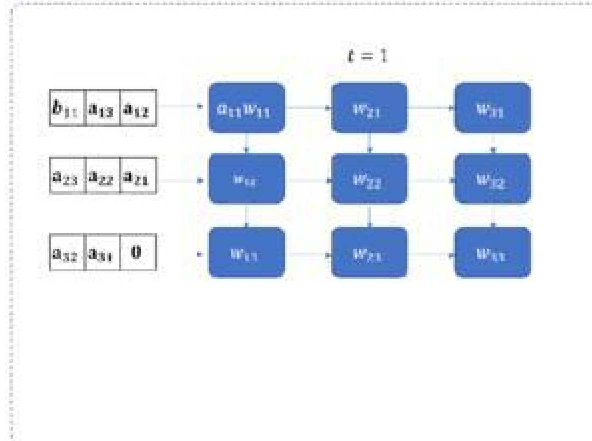
- 深度学习：**反向促进硬件研发**，大厂都在做
- Google TPU：可媲美Nvidia GPU性能
  - Google内部大量部署
  - 核心是**Systolic Array**



Systolic Array

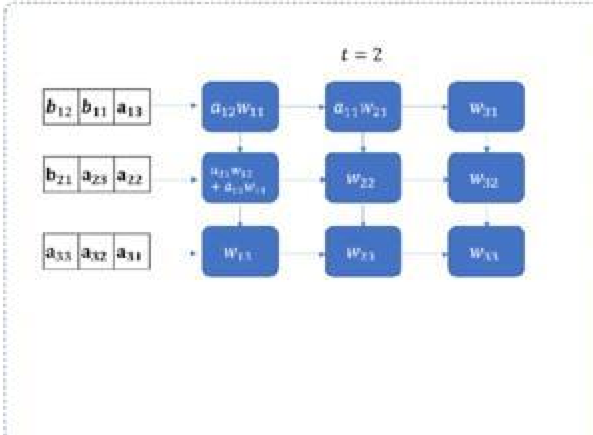
中间结果向右传；输出向下传

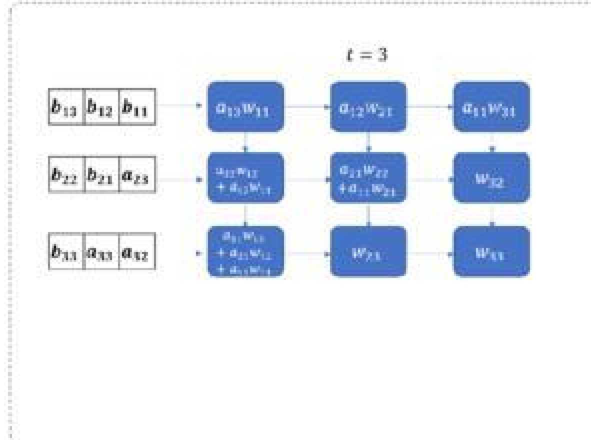




Systolic Array I

中间结果向右传；输出向下传

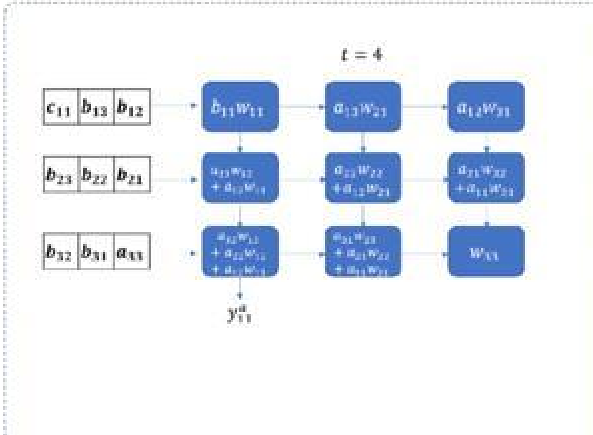


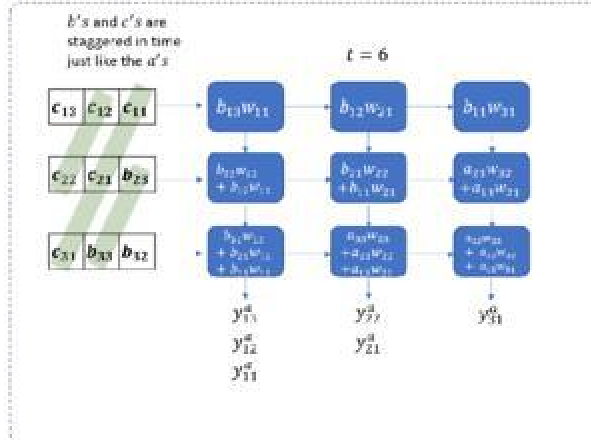


- $y_{11} = a_{11}w_{11} + a_{21}w_{12} + a_{31}w_{13}$

Systolic Array II

中间结果向右传；输出向下传





- $y_{11} = a_{11}w_{11} + a_{21}w_{12} + a_{31}w_{13}$
- 同样方式继续计算B、C

实验：GPU

Review

本章内容

数值稳定性。Xavier 初始化。层、块。参数管理。自定义层。读写文件。并行架构。

重点：数值不稳定的两个表现：梯度爆炸、梯度消失；Xavier 初始化的结论；层、块的构建方法；参数管理的方法；自定义层的方法；读写文件的方法；并行架构的使用方法。

难点：数值不稳定可能带来的问题；初始化的重要性；并行计算原则。

学习目标

- 理解数值不稳定的两个表现：梯度爆炸、梯度消失，及其可能带来的问题。
- 理解优化问题中初始化的重要性，及 Xavier 初始化的结论。
- 理解层、块的基本功能需求，并掌握构建方法。
- 理解参数管理的原因，并掌握方法。
- 理解自定义层的原因，并掌握方法。
- 理解读写文件的原因，并掌握方法。
- 了解并行架构（GPU、TPU），理解并行计算原则，并掌握使用方法。

问题

简述数值不稳定的两个表现，及其可能带来的问题。

简述优化问题中初始化的重要性。

简述并行架构特点、计算原则。

Appendix

随机梯度下降

小批量随机梯度下降

动量法