计算图、反向传播与梯度下降
初学者指南
从基础概念到实际应用

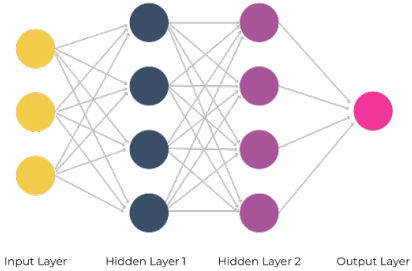
Anson

深度学习社

2025 年 10 月 14 日

Cooperated with Kimi AI

目录



什么是机器学习?

简单定义

机器学习是让计算机从数据中学习，而不需要明确编程每一个规则。

生活中的例子

- 垃圾邮件过滤
- 推荐系统（Netflix、淘宝）
- 语音识别
- 图像识别

关键问题

如何自动调整模型的参数，使其表现更好?

为什么需要计算图和反向传播?

- 现代机器学习模型（如神经网络）非常复杂
- 需要高效的计算方法来训练这些模型
- 计算图提供了清晰的数学框架
- 反向传播让我们能够高效地计算梯度
- 梯度下降帮助我们找到最优参数

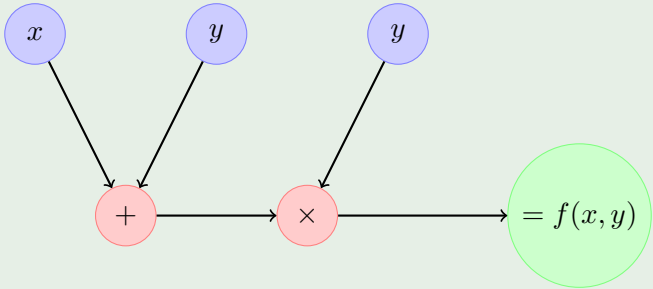
今天我们将学习这三个核心概念!

什么是计算图?

直观理解

计算图就是把数学表达式画成图的形式，让我们更容易理解和计算。

简单例子：计算 $f(x, y) = (x + y) \times y$



计算图的基本元素

节点（Nodes）

- 输入节点：变量或常数
- 操作节点：数学运算
- 每个节点代表一个计算步骤

边（Edges）

- 表示数据流动方向
- 从前一个节点的输出到后一个节点的输入
- 帮助追踪计算的依赖关系

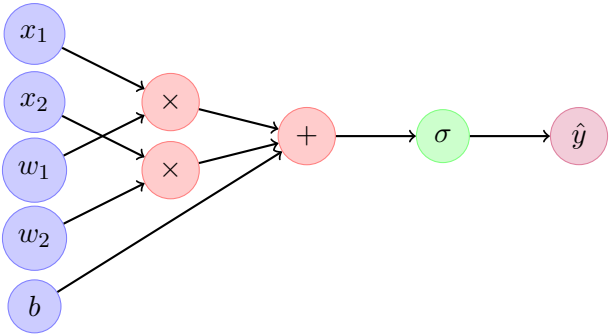
为什么使用计算图?

- 清晰：直观展示计算过程
- 灵活：容易修改和扩展
- 高效：便于自动求导
- 通用：适用于各种机器学习模型

更复杂的例子：逻辑回归

逻辑回归模型

预测概率：$\hat{y} = \sigma(w_1 x_1 + w_2 x_2 + b)$，其中$\sigma(z) = \frac{1}{1+e^{-z}}$是 sigmoid 函数



什么是前向传播?

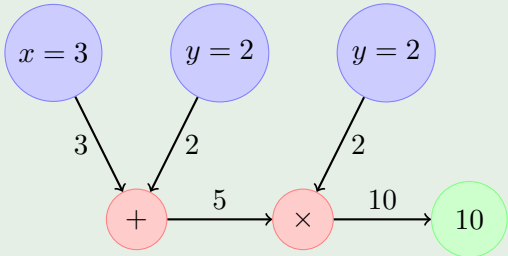
直观理解

前向传播就是从左到右沿着计算图计算每个节点的值，直到得到最终结果。

回忆简单例子：$f(x, y) = (x + y) \times y$

假设 $x = 3, y = 2$:

1. 计算 $x + y = 3 + 2 = 5$
2. 计算 $(x + y) \times y = 5 \times 2 = 10$
3. 得到最终结果：$f(3, 2) = 10$



$f(x, y) = (x + y) \times y$

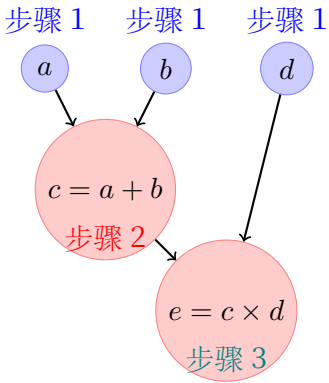
前向传播的通用步骤

算法步骤

1. 初始化：给输入节点赋值
2. 遍历：按照拓扑顺序访问每个操作节点
3. 计算：根据输入值计算当前节点的输出值
4. 存储：保存计算结果供后续使用

关键概念：拓扑顺序

确保在计算一个节点之前，它的所有输入节点都已经计算完毕。



为什么需要反向传播?

问题背景

在机器学习中，我们需要知道如何调整参数才能让模型表现得更好。

具体例子

假设我们有一个预测模型，当前预测值是 $\hat{y} = 0.7$，但真实值是 $y = 1.0$：

- 预测误差：0.3
- 问题：哪些参数对这个误差贡献最大?
- 目标：如何调整参数来减小这个误差?

关键洞察

我们需要计算梯度（偏导数）来了解每个参数对最终误差的影响程度!

反向传播的直观理解

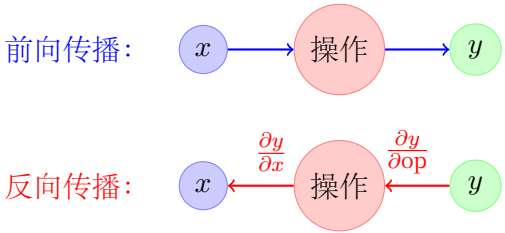
核心思想

反向传播就是从右到左沿着计算图，计算每个节点对最终输出的影响程度（梯度）。

类比：影响链

想象一个公司组织结构：

- 总经理（输出）表现不好
- 要找出哪些部门经理（中间层）负主要责任
- 再找出哪些员工（输入）需要改进



链式法则：反向传播的数学基础

链式法则（Chain Rule）

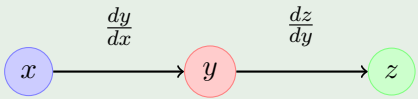
如果 $z = f(y)$ 且 $y = g(x)$，那么：

$$\frac{dz}{dx} = \frac{dz}{dy} \times \frac{dy}{dx}$$

直观解释

要计算 z 对 x 的影响，可以分两步：

1. 计算 z 对 y 的影响：$\frac{dz}{dy}$
2. 计算 y 对 x 的影响：$\frac{dy}{dx}$
3. 相乘得到总影响



$$\frac{dz}{dx} = \frac{dz}{dy} \times \frac{dy}{dx}$$

反向传播的详细步骤

步骤 1：前向传播计算所有节点值

先正常计算每个节点的值，并保存起来。

步骤 2：从输出节点开始反向传播梯度

1. 输出节点的梯度为 1：$\frac{\partial L}{\partial L} = 1$
2. 对每个节点，计算其对损失函数的梯度
3. 使用链式法则将梯度反向传播

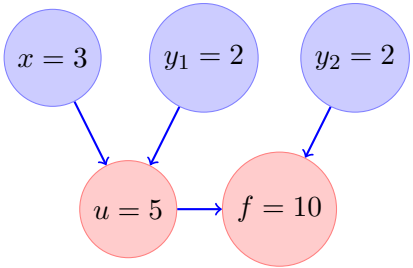
简单例子：$f(x, y) = (x + y) \times y$

假设 $x = 3, y = 2$，我们要计算 $\frac{\partial f}{\partial x}$ 和 $\frac{\partial f}{\partial y}$：

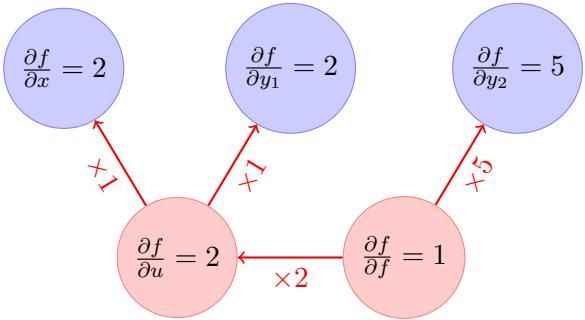
- 前向传播：$u = x + y = 5$，
  $f = u \times y = 10$
- 反向传播：$\frac{\partial f}{\partial u} = y = 2$，$\frac{\partial f}{\partial y} = u = 5$
- 继续：$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial u} \times \frac{\partial u}{\partial x} = 2 \times 1 = 2$

计算图示例：前向传播与反向传播

前向传播值



反向传播梯度



反向传播应用链式法则: $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial u} \cdot \frac{\partial u}{\partial x}$

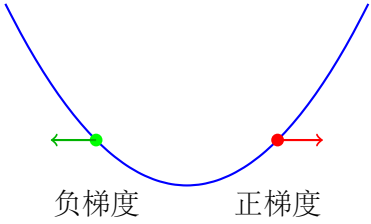
为什么需要梯度下降?

核心问题

我们已经知道如何计算梯度（偏导数），但如何利用这些信息来改进模型?

直觉思考

- 梯度告诉我们每个参数对误差的影响方向
- 正梯度：增加参数会增加误差
- 负梯度：增加参数会减小误差
- 我们应该沿着梯度的反方向调整参数!



我们应该向左移动来减小函数值

梯度下降算法

基本思想

沿着梯度的反方向小步移动，逐步减小损失函数。

数学表达

对于参数 θ，更新规则是：

$$\theta_{\text{新}} = \theta_{\text{旧}} - \alpha \frac{\partial L}{\partial \theta}$$

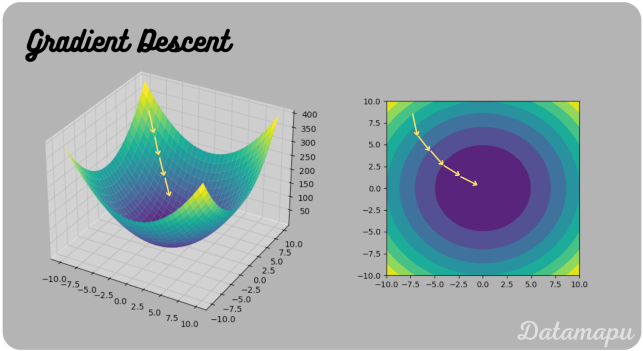
其中：

- α 是学习率（步长大小）
- $\frac{\partial L}{\partial \theta}$ 是损失函数对参数的梯度
- 减号表示沿着梯度的反方向移动

类比：下山

- 你站在山上，想下到最低点
- 观察周围最陡的方向（梯度）
- 沿着相反方向走一小步
- 重复这个过程直到到达谷底

梯度下降的完整过程



算法步骤

1. 初始化参数 θ
2. 计算当前梯度 $\frac{\partial L}{\partial \theta}$
3. 更新参数：$\theta = \theta - \alpha \frac{\partial L}{\partial \theta}$
4. 重复步骤 2-3 直到收敛

学习率的重要性

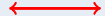
学习率太小

- 收敛速度很慢
- 需要很多迭代
- 可能陷入局部最优

小步移动

学习率太大

- 可能错过最优解
- 在最优解附近震荡
- 甚至发散（不收敛）



来回震荡

选择合适的学习率

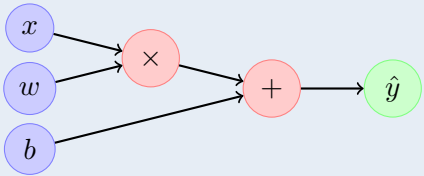
- 通常从 0.01, 0.001 等值开始尝试
- 可以随着训练逐渐减小
- 使用自适应方法（如 Adam, RMSprop）

完整例子：训练一个简单的模型

问题设定

我们要训练一个模型 $f(x) = wx + b$ 来拟合数据点 $(2, 5)$。

模型结构



训练步骤

1. 前向传播：计算 $\hat{y}$
2. 计算损失：$L = (\hat{y} - y)^2$
3. 反向传播：计算梯度
4. 梯度下降：更新 w 和 b
5. 重复直到收敛

具体计算过程

初始化

设 $w = 1, b = 0$，学习率 $\alpha = 0.1$

结果

新的参数：$w = 2.2, b = 0.6$，预测值更接近真实值了！

下一步

下一步，我们只需要重复这个过程，直到损失函数收敛到一个较小的值。

第 1 轮迭代

1. 前向传播：$x = 2 \Rightarrow \hat{y} = 1 \times 2 + 0 = 2$
2. 计算损失：$L = (\hat{y} - y)^2 = (2 - 5)^2 = 9$
3. 反向传播：
   - $\frac{\partial L}{\partial \hat{y}} = 2(\hat{y} - y) = 2(2 - 5) = -6$
   - $\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial w} = -6 \times 2 = -12$
   - $\frac{\partial L}{\partial b} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial b} = -6 \times 1 = -6$
4. 梯度下降：
   - $w_{\text{新}} = w - \alpha \frac{\partial L}{\partial w} = 1 - 0.1 \times (-12) = 2.2$
   - $b_{\text{新}} = b - \alpha \frac{\partial L}{\partial b} = 0 - 0.1 \times (-6) = 0.6$

今天我们学到了什么?

1. 计算图

- 将数学表达式可视化为图结构
- 清晰展示计算过程和依赖关系
- 为自动求导奠定基础

2. 反向传播

- 高效计算梯度的算法
- 基于链式法则从后向前传播梯度
- 让我们知道每个参数对误差的影响

3. 梯度下降

- 利用梯度信息优化参数
- 沿着负梯度方向小步前进
- 逐步减小损失函数，提高模型性能

在深度学习中的应用

现代深度学习框架

这些概念是所有深度学习框架的核心：

- PyTorch
- TensorFlow
- JAX
- MXNet

- 自动构建计算图
- 自动微分（反向传播）
- 内置优化器（梯度下降变体）
- 让我们专注于模型设计

实际应用

- 计算机视觉：图像分类、目标检测
- 自然语言处理：机器翻译、文本生成
- 语音识别、推荐系统、游戏 AI
- 科学计算、医疗诊断、金融预测

下一步学习建议

巩固基础

1. 动手实现简单的神经网络
2. 使用 PyTorch 或 TensorFlow 练习
3. 可视化梯度流动和参数更新
4. 调试和优化训练过程

深入拓展

- 不同类型的优化器（Adam, RMSprop）
- 梯度消失和梯度爆炸问题
- 更复杂的网络结构（CNN, RNN, Transformer）
- 正则化技术和训练技巧

谢谢大家！

有问题欢迎提问！