深度学习中的优化器

Shwj

2023 年 2 月 20 日

参考资料：

1. https://www.bilibili.com/video/BV1X34y197mF
2. 王木头学科学

```
optimizer = optim.SGD(model.parameters(),
            lr=0.01, momentum=0.9)
```

1 前言

神经网络相当于一个函数 f，对 f 展开有

$$f(\boldsymbol{x}+d\boldsymbol{x}) \approx f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T \cdot d\boldsymbol{x} + \frac{1}{2} d\boldsymbol{x}^T \cdot H_f \cdot d\boldsymbol{x}$$

其中 H_f 是 f 的 $Hessian\,Matrix$，我们一般展开到 f 的一阶无穷小，即

$$f(\boldsymbol{x}+d\boldsymbol{x}) \approx f(\boldsymbol{x}) + \nabla f(\boldsymbol{x}) \cdot d\boldsymbol{x}$$

神经网络的 f 难以写出显示表达式，自然无法用求导的方法求得最小值。但我们可以利用迭代的方法，使其每次迭代时函数值减小，为此取 $d\boldsymbol{x} = -\eta \nabla f(\boldsymbol{x})$ 则有

$$\nabla f(\boldsymbol{x}) \cdot d\boldsymbol{x} = -\eta \nabla f(\boldsymbol{x})^2 \leq 0$$

2 牛顿法

随机梯度下降法和标准的梯度下降法的区别我暂时蒙在鼓里，或许标准的方法是把整个样本集丢进去求梯度，而 SGD 是每次取一个 batch 做 BP 求梯度？

牛顿法考虑了函数的泰勒展开的二次项，相当于在对应点拿抛物线逼近函数，它的更新公式为

$$\boldsymbol{x} = \boldsymbol{x} - H_f(\boldsymbol{x})^{-1} \cdot \nabla f(\boldsymbol{x})$$

在 $f(\boldsymbol{x}) = f(\boldsymbol{x_0}) + \nabla f(\boldsymbol{x_0})^T \cdot (\boldsymbol{x} - \boldsymbol{x_0}) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x_0})^T \cdot H_f(\boldsymbol{x_0}) \cdot (\boldsymbol{x} - \boldsymbol{x_0})$ 中，对 $\boldsymbol{x}$ 求偏导，就有 $f'(\boldsymbol{x}) = \nabla f(\boldsymbol{x_0})^T + \cdot H_f(\boldsymbol{x_0}) \cdot (\boldsymbol{x} - \boldsymbol{x_0})$，令 $f'(\boldsymbol{x}) = 0$ 就可以得到结论。

牛顿法相对于提前给定学习率 η 的梯度下降法，它能根据当前的函数值动态确定学习率，但是缺点是 $H_f^{-1}(\boldsymbol{x_0})$ 的计算量过大

3 随机梯度下降 (SGD)

4 带动量的梯度下降法

$$w_t = w_{t-1} - \eta \cdot g_w$$

$$v_t = \beta_1 \cdot v_{t-1} + (1 - \beta_1) g_w$$

$$\overline{v_t} = \frac{v_t}{1 - \beta_1^t}$$

$$w_t = w_{t-1} - \eta \cdot \overline{v_t}$$

下面的方法都是研究梯度下降的方法。梯度虽然能给出下降最快的方向，却没给出最优的下降路径。拿每次梯度更新得到的直线路径去逼近真实的下降曲线。

5 均方根传递

$$w_t = w_{t-1} - \eta \cdot g_w$$

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_2) g^2$$

$$\overline{m_t} = \frac{m_t}{1 - \beta_1^t}$$

$$w_t = w_{t-1} - \eta \cdot \frac{g_w}{\sqrt{\overline{m_t}}}$$

6 自适应矩估计 (Adam)

$$w_t = w_{t-1} - \eta \frac{\overline{v_t}}{\sqrt{\overline{m_t}}}$$

7 ceshi