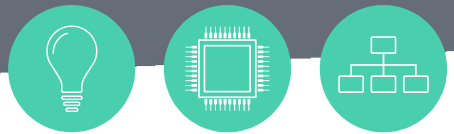
深度学习之 PyTorch 实战

优化算法 part 1

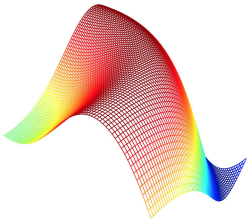


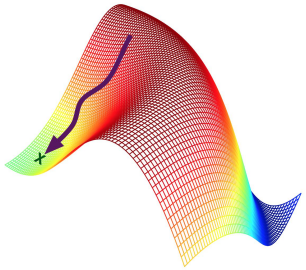
主讲老师: 土豆老师

目录

优化与深度学习

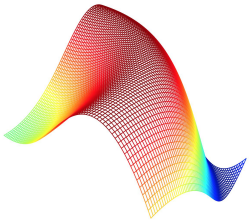
- 本讲将讨论优化与深度学习的关系，以及优化在深度学习中的挑战。
- 在一个深度学习问题中，我们通常会预先定义一个损失函数。有了损失函数以后，我们就可以使用优化算法试图将其最小化。在优化中，这样的损失函数通常被称作优化问题的目标函数（objective function）。
- 依据惯例，优化算法通常只考虑最小化目标函数。其实，任何最大化问题都可以很容易地转化为最小化问题，只需令目标函数的相反数为新的目标函数即可。

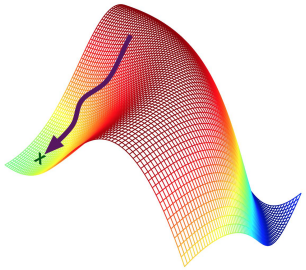




优化与深度学习

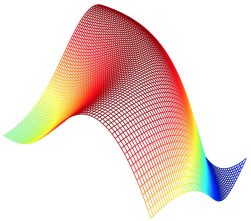
- 虽然优化为深度学习提供了最小化损失函数的方法，但本质上，优化与深度学习的目标是有区别的。
- 在「模型选择、欠拟合和过拟合」一讲中，我们区分了训练误差和泛化误差。由于优化算法的目标函数通常是一个基于训练数据集的损失函数，优化的目标在于降低训练误差。而深度学习的目标在于降低泛化误差。为了降低泛化误差，除了使用优化算法降低训练误差以外，还需要注意应对过拟合。
- 在本讲中，我们只关注优化算法在最小化目标函数上的表现，而不关注模型的泛化误差。

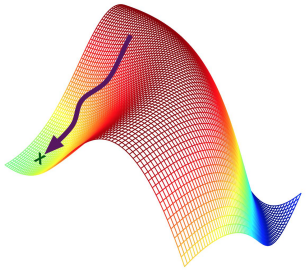




优化与深度学习

- 我们在「线性回归」节中对优化问题的解析解和数值解做了区分。深度学习中绝大多数目标函数都很复杂。因此，很多优化问题并不存在解析解，而需要使用基于数值方法的优化算法找到近似解，即数值解。
- 本讲中讨论的优化算法都是这类基于数值方法的算法。为了求得最小化目标函数的数值解，我们将通过优化算法有限次迭代模型参数来尽可能降低损失函数的值。
- 优化在深度学习中有很多挑战。下面描述了其中的两个挑战，即局部最小值和鞍点。

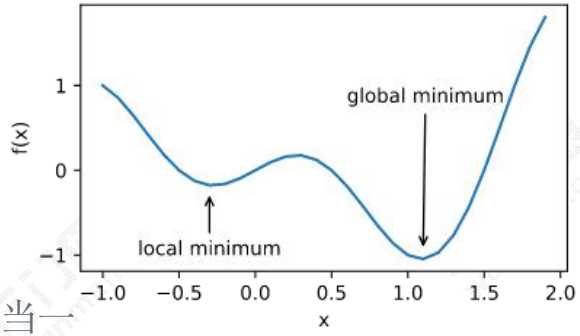




优化与深度学习

- 局部最小值
  - 对于目标函数 $f(x)$，如果 $f(x)$ 在 x 上的值比在 x 邻近的其他点的值更小，那么 $f(x)$ 可能是一个局部最小值（local minimum）。如果 $f(x)$ 在 x 上的值是目标函数在整个定义域上的最小值，那么 $f(x)$ 是全局最小值（global minimum）。
  - 举个例子，给定函数：

$$f(x) = x \cdot \cos(\pi x), \quad -1.0 \leq x \leq 2.0$$



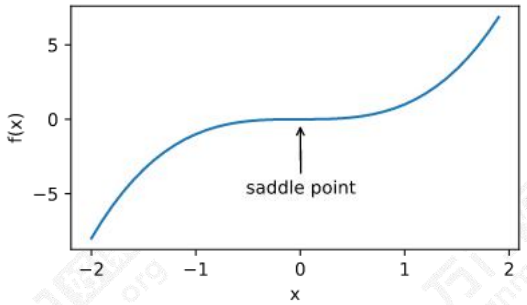
  - 深度学习模型的目标函数可能有若干局部最优值。当一个优化问题的数值解在局部最优解附近时，由于目标函数有关解的梯度接近或变成零，最终迭代求得的数值解可能只令目标函数局部最小化而非全局最小化。

优化与深度学习

- 鞍点
  - 刚刚我们提到，梯度接近或变成零可能是由于当前解在局部最优解附近造成的。事实上，另一种可能性是当前解在鞍点（saddle point）附近。
  - 举个例子，给定函数

$$f(x) = x^3$$

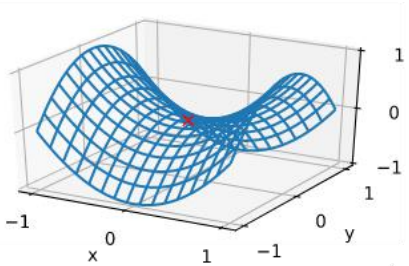
我们可以找出该函数的鞍点位置。



  - 再举个定义在二维空间的函数的例子，例如：

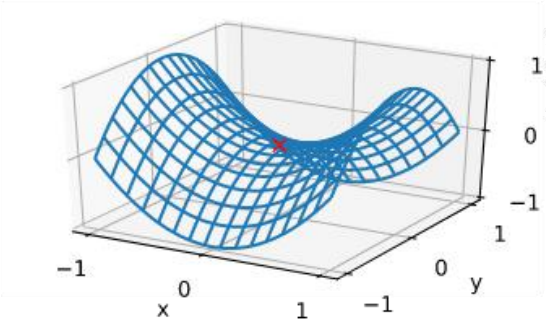
$$f(x, y) = x^2 - y^2$$

我们可以找出该函数的鞍点位置。也许你已经发现了，该函数看起来像一个马鞍，而鞍点恰好是马鞍上可坐区域的中心。



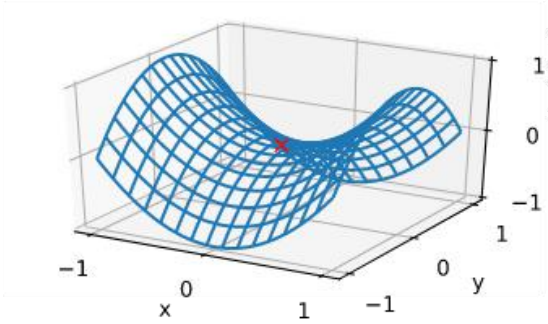
优化与深度学习

- 鞍点
  - 在图的鞍点位置，目标函数在 x 轴方向上是局部最小值，但在 y 轴方向上是局部最大值。
  - 假设一个函数的输入为 k 维向量，输出为标量，那么它的海森矩阵（Hessian matrix）有 k 个特征值。该函数在梯度为 0 的位置上可能是局部最小值、局部最大值或者鞍点。
    - 当函数的海森矩阵在梯度为零的位置上的特征值全为正时，该函数得到局部最小值。
    - 当函数的海森矩阵在梯度为零的位置上的特征值全为负时，该函数得到局部最大值。
    - 当函数的海森矩阵在梯度为零的位置上的特征值有正有负时，该函数得到鞍点。



优化与深度学习

- 鞍点
  - 随机矩阵理论告诉我们，对于一个大的高斯随机矩阵来说，任一特征值是正或者是负的概率都是 0.5。那么，以上第一种情况的概率为 0.5^k。由于深度学习模型参数通常都是高维的（k 很大），目标函数的鞍点通常比局部最小值更常见。
  - 在深度学习中，虽然找到目标函数的全局最优解很难，但这并非必要。我们将在本讲接下来的几节中逐一介绍深度学习中常用的优化算法，它们在很多实际问题中都能够训练出十分有效的深度学习模型。



梯度下降和随机梯度下降

- 在本节中，我们将介绍梯度下降（gradient descent）的工作原理。虽然梯度下降在深度学习中很少被直接使用，但理解梯度的意义以及沿着梯度反方向更新自变量可能降低目标函数值的原因是学习后续优化算法的基础。随后，我们将引出随机梯度下降（stochastic gradient descent）。
- 一维梯度下降
  - 我们先以简单的一维梯度下降为例，解释梯度下降算法可能降低目标函数值的原因。假设连续可导的函数 $f:\mathbb{R}\rightarrow\mathbb{R}$ 的输入和输出都是标量。给定绝对值足够小的数 ϵ，根据泰勒展开公式，我们得到以下的近似：

    $$f(x+\epsilon)\approx f(x)+\epsilon f'(x)$$

    这里 $f'(x)$ 是函数 f 在 x 处的梯度。一维函数的梯度是一个标量，也称导数。
  - 接下来，找到一个常数 $\eta>0$，使得 $|\eta f'(x)|$ 足够小，那么可以将 ϵ 替换为 $-\eta f'(x)$ 并得到

    $$f\big(x-\eta f'(x)\big)\approx f(x)-\eta f'(x)^2.$$

梯度下降和随机梯度下降

- 一维梯度下降

$$f\big(x - \eta f'(x)\big) \approx f(x) - \eta f'(x)^2.$$

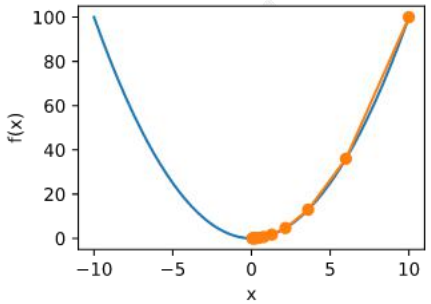
  - 如果导数 $f'(x) \neq 0$，那么 $\eta f'(x)^2 > 0$，所以

$$f\big(x - \eta f'(x)\big) \lesssim f(x)$$

    这意味着，如果通过

$$x \leftarrow x - \eta f'(x)$$

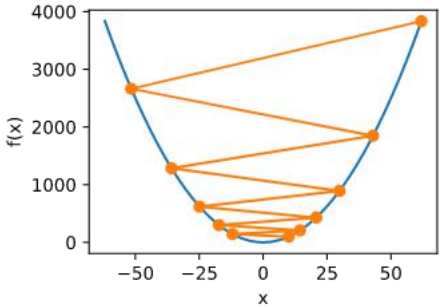
    来迭代 x，函数 $f(x)$ 的值可能会降低。因此在梯度下降中，我们先选取一个初始值 x 和常数 $\eta > 0$，然后不断通过上式来迭代 x，直到达到停止条件，例如 $f'(x)^2$ 的值已足够小或迭代次数已达到某个值。



“Talk is cheap. Show me the code.”

梯度下降和随机梯度下降

- 学习率
  - 上述梯度下降算法中的正数 η 通常叫作学习率。这是一个超参数，需要人工设定。
  - 如果使用过小的学习率，会导致 x 更新缓慢从而需要更多的迭代才能得到较好的解。
  - 如果使用过大的学习率，$|\eta f'(x)|$ 可能会过大从而使前面提到的一阶泰勒展开公式不再成立：这时我们无法保证迭代 x 会降低 $f(x)$ 的值。



“ Talk is cheap. Show me the code. ”

梯度下降和随机梯度下降

- 多维梯度下降
  - 在了解了一维梯度下降之后, 我们再考虑一种更广义的情况：目标函数的输入为向量，输出为标量。假设目标函数 $f: \mathbb{R}^d \to \mathbb{R}$ 的输入是一个 d 维向量 $\boldsymbol{x} = [x_1, x_2, \ldots, x_d]^\top$ 。目标函数 $f(\boldsymbol{x})$ 有关 $\boldsymbol{x}$ 的梯度是一个由 d 个偏导数组成的向量:

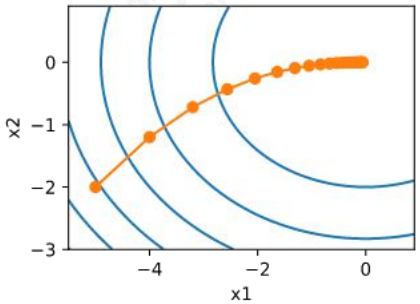
$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \left[\frac{\partial f(\boldsymbol{x})}{\partial x_1}, \frac{\partial f(\boldsymbol{x})}{\partial x_2}, \ldots, \frac{\partial f(\boldsymbol{x})}{\partial x_d}\right]^\top$$

  - 同样道理，我们可能通过梯度下降算法来不断降低目标函数 f 的值:

$$\boldsymbol{x} \leftarrow \boldsymbol{x} - \eta \nabla f(\boldsymbol{x})$$

同样，其中 η (取正数) 称作学习率。

“Talk is cheap. Show me the code.”



梯度下降和随机梯度下降

- 随机梯度下降
  - 在深度学习里，目标函数通常是训练数据集中有关各个样本的损失函数的平均。设 $f_i(\boldsymbol{x})$ 是有关索引为 i 的训练数据样本的损失函数，n 是训练数据样本数，$\boldsymbol{x}$ 是模型的参数向量，那么目标函数定义为

$$f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x})$$

  - 目标函数在 $\boldsymbol{x}$ 处的梯度计算为

$$\nabla f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{x})$$

  - 如果使用梯度下降，每次自变量迭代的计算开销为 $\mathcal{O}(n)$，它随着 n 线性增长。因此，当训练数据样本数很大时，梯度下降每次迭代的计算开销很高。

梯度下降和随机梯度下降

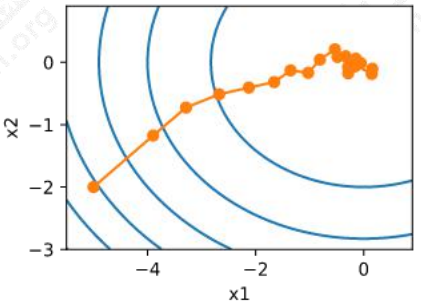
- 随机梯度下降
  - 随机梯度下降（stochastic gradient descent，SGD）减少了每次迭代的计算开销。在随机梯度下降的每次迭代中，我们随机均匀采样的一个样本索引 $i \in \{1, \ldots, n\}$，并计算梯度 $\nabla f_i(\boldsymbol{x})$ 来迭代 $\boldsymbol{x}$：

$$\boldsymbol{x} \leftarrow \boldsymbol{x} - \eta \nabla f_i(\boldsymbol{x})$$

  - 这里 η 同样是学习率。可以看到每次迭代的计算开销从梯度下降的 $\mathcal{O}(n)$ 降到了常数 $\mathcal{O}(1)$。值得强调的是，随机梯度 $\nabla f_i(\boldsymbol{x})$ 是对梯度 $\nabla f(\boldsymbol{x})$ 的无偏估计：

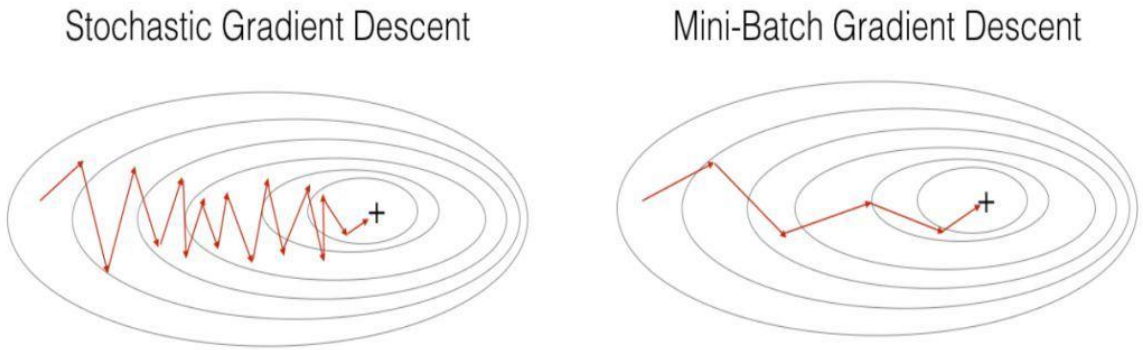
$$E_i \nabla f_i(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(\boldsymbol{x}) = \nabla f(\boldsymbol{x})$$

  - 这意味着，平均来说，随机梯度是对梯度的一个良好的估计。



小批量随机梯度下降

- 在每一次迭代中，梯度下降使用整个训练数据集来计算梯度，因此它有时也被称为批量梯度下降（batch gradient descent）。而随机梯度下降在每次迭代中只随机采样一个样本来计算梯度。正如我们在前几讲中所看到的，我们还可以在每轮迭代中随机均匀采样多个样本来组成一个小批量，然后使用这个小批量来计算梯度。下面就来描述小批量随机梯度下降（mini-batch gradient descent）。



小批量随机梯度下降

- 设目标函数 $f(\boldsymbol{x}): \mathbb{R}^d \rightarrow \mathbb{R}$。在迭代开始前的时间步设为 0。该时间步的自变量记为 $\boldsymbol{x}_0 \in \mathbb{R}^d$，通常由随机初始化得到。在接下来的每一个时间步 $t > 0$ 中，小批量随机梯度下降随机均匀采样一个由训练数据样本索引组成的小批量 $\mathcal{B}_t$。我们可以通过重复采样 (sampling with replacement) 或者不重复采样 (sampling without replacement) 得到一个小批量中的各个样本。前者允许同一个小批量中出现重复的样本，后者则不允许如此，且更常见。对于这两者间的任一种方式，都可以使用

$$\boldsymbol{g}_t \leftarrow \nabla f_{\mathcal{B}_t}(\boldsymbol{x}_{t-1}) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}_t} \nabla f_i(\boldsymbol{x}_{t-1})$$

来计算时间步 t 的小批量 $\mathcal{B}_t$ 上目标函数位于 $\boldsymbol{x}_{t-1}$ 处的梯度 $\boldsymbol{g}_t$。这里 $|\mathcal{B}|$ 代表批量大小，即小批量中样本的个数，是一个超参数。同随机梯度一样，重复采样所得的小批量随机梯度 $\boldsymbol{g}_t$ 也是对 $\nabla f(\boldsymbol{x}_{t-1})$ 的无偏估计。给定学习率 η_t（取正数），小批量随机梯度下降对自变量的迭代如下

$$\boldsymbol{x}_t \leftarrow \boldsymbol{x}_{t-1} - \eta_t \boldsymbol{g}_t$$

“Talk is cheap. Show me the code.”

小结

- 由于优化算法的目标函数通常是一个基于训练数据集的损失函数，优化的目标在于降低训练误差。
- 由于深度学习模型参数通常都是高维的，目标函数的鞍点通常比局部最小值更常见。
- 使用适当的学习率，沿着梯度反方向更新自变量可能降低目标函数值。梯度下降重复这一更新过程直到得到满足要求的解。
- 学习率过大或过小都有问题。一个合适的学习率通常是需要通过多次实验找到的。
- 当训练数据集的样本较多时，梯度下降每次迭代的计算开销较大，因而随机梯度下降通常更受青睐。
- 小批量随机梯度每次随机均匀采样一个小批量的训练样本来计算梯度。
- 在实际中，（小批量）随机梯度下降的学习率可以在迭代过程中自我衰减。
- 通常，小批量随机梯度在每个迭代周期的耗时介于梯度下降和随机梯度下降的耗时之间。

谢谢观看

更多好课，请关注万门好课APP