

## 【译】cs231n第四课：理解后向传播

📅 2016-07-25 | 👁 2

本来是要学习cs224d的，但cs224d要求学几节cs231n的课程作为预备课程，故有如此翻译。

[原文链接](#)

## 介绍

本节讲解反向传播（backpropagation），它反复利用链式法则（chain rule）计算梯度。理解反向传播的过程对理解、开发、debug神经网络非常重要。

重述一下问题，对于方程 $f(x)$ 和一个向量输入 $x$ ，我们需要计算出 $f$ 在 $x$ 处的导数（比如 $\nabla f(x)$ ）

之所以需要这么做，是因为在神经网络中，会有一个损失函数(就当成上面的 $f$ )，还有训练集、神经网络的权重作为输入之间 $x$ 。举例来说，损失函数可以为SVM损失函数，输入包括训练集 $(x_i, y_i), i = 1 \dots N$ 和权重矩阵 $W$ 、偏置矩阵 $b$ 。在通常的机器学习的任务中，训练集是固定的，权重才是我们需要调整的参数。因此，尽管我们也能使用反向传播算出样本点 $x_i$ 的梯度，实际上我们也只需要算出权重和偏置的梯度，这样我们能轻易的进行参数更新，让损失函数的值更小。之后的课程我们会发现算出 $x_i$ 处的梯度也是非常有用的，比如可视化和解释神经网络到底在做什么。

## 用简单的表达式理解梯度

在学习复杂的表达式之前，先用一个简单的公式 $f(x, y) = xy$ ，计算其梯度。偏导如下：

$$f(x, y) = xy \quad \rightarrow \quad \frac{\partial f}{\partial x} = y \quad \frac{\partial f}{\partial y} = x$$

导数的含义是在一个点周围极其微小范围内公式的变化率，公式如下：

$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

注意上述公式左边的横线，它不是一个除号。 $\frac{d}{dx}$ 是一个操作符，它对公式 $f$ 进行了一个操作，返回了导函数。一个认识上述公式的方式是当 $h$ 非常小的时候，这个公式的图像会接近于一条直线，导数就是直线的斜率。换句话说，导数告诉我们在某个点的敏感程度（变化的快或慢）。比如，当 $x = 4, y = -3$ 时， $f(x, y) = -12$ 并且在 $x$ 处的导数为 $\frac{\partial f}{\partial x} = -3$ ，这表明如果我们稍稍增加一点 $x$ 的值，那么方程 $f$ 的值将会降低（因为负符号），并且以三倍于 $x$ 增加的值降低。我们也可以通过重构上述公式得到这个事实： $f(x+h) = f(x) + h \frac{df(x)}{dx}$ 。同样，因为公式对于 $y$ 的导数是4，所以这表示增加 $h$ 点 $y$ 值，那么公式 $f$ 的结果也会增加（因为导数是正的） $4h$ 。

如前所述， $\nabla f$ 是一个偏导数的向量： $\nabla f = [\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}] = [y, x]$

除了乘法，加法求导规则如下：

$$f(x, y) = x + y \quad \rightarrow \quad \frac{\partial f}{\partial x} = 1 \quad \frac{\partial f}{\partial y} = 1$$

从公式来看， $x, y$ 的导数都是1，这和 $x, y$ 值都没有关系。这很合理，因为增加 $x$ 或 $y$ 的值会直接增加 $f$ 的值，变化率独立于 $x$ 与 $y$ 具体的值（这与上面的乘法不同）。

最后一个max操作的导数：

$$f(x, y) = \max(x, y) \quad \rightarrow \quad \frac{\partial f}{\partial x} = \mathbb{1}(x \geq y) \quad \frac{\partial f}{\partial y} = \mathbb{1}(y \geq x)$$

从公式上看，当输入大于另一个输入时，响应输入的梯度为1。比如，当 $x = 4, y = 2$ 时， $\max$ 是4，因此这个方程对 $y$ 的值不敏感。也就是说，当我们给 $y$ 增加一点点值 $h$ 时，公式还是会输出4，因此对 $x$ 求导时，与 $y$ 无关， $f$ 方程式中的 $y$ 项应该为0，目前上述公式中只出现了 $x$ 项的处理。当然，如果我们大幅度增加 $y$ 值（超过2），那么 $f$ 的值将会变化、受影响，但是导数并不能处理当输入的数发生巨大变化的情况。导数只有当输入的值发生微小的变化时才有意义，正如定义中所述： $\lim_{h \rightarrow 0}$ 。

## 使用链式法则计算复合的公式的导数

让我们考虑一个复合公式： $f(x, y, z) = (x + y)z$ ，虽然这个方程也可以直接求导，但是我们使用一个有助于我们理解反向传播的计算方式。先将这个方程分解成两部分： $q = x + y$ 和 $f = qz$ 。对于这两部分，我们是分开计算的，比如 $f$ 是 $q$ 与 $z$ 的乘法 $\frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$ ； $q$ 是 $x$ 与 $y$ 的加法： $\frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$ ，并且我们对中间媒介 $q$ 的导数 $\frac{\partial f}{\partial q}$ 并不感兴趣，我们只关心 $x, y, z$ 对 $f$ 的导数感兴趣。链式法则就是告诉我们一种乘法，能过进行导数之间的计算： $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$ ，实际上在代码中也只需要将两个导数相乘即可：

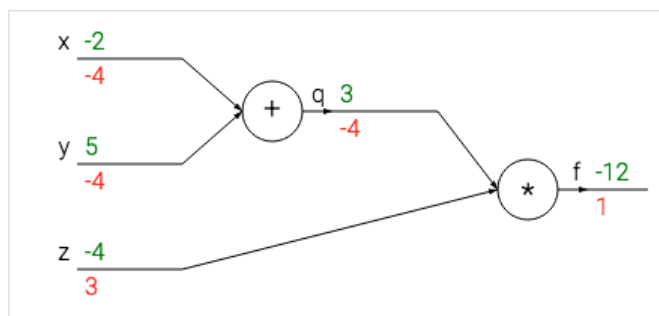
```

1
2  # 样例输入
3  x = -2; y = 5; z = -4
4
5  #先正向传播
6  q = x + y  #q=3
7  f = q * z  #f=-12
8
9  # 反向传播的第一步: f = q * z
10 dfdz = q # df/dz = q, 对于z的导数是 3
11 dfdq = z # df/dq = z, 对于q的导数 -4
12
13 # 再反向传播通过: q = x + y
14 dfdx = 1.0 * dfdq # dq/dx = 1. 这里进行的相乘就是链式法则
15 dfdy = 1.0 * dfdq # dq/dy = 1
16

```

最后我们剩下变量： $[dfdx, dfdy, dfdz]$ ，这三个数分别告诉了我们 $f$ 对 $x, y, z$ 的敏感程度。这是一个反向传播的简单例子。为了简洁，我们在表达式中删除 $df$ 部分。比如 $dfdq$ 用 $dq$ 表达，反正我们都知道这是针对公式 $f$ 的导数。

用电路图表示下路过程：



左边是带有实数的电路图表示着复合公式，正向传播计算结果（绿色表示），反向传播是从最后到开始反复使用链式法则计算出对各个输入导数（红色表示）。导数可以认为是逆着电路方向传播计算的得出的。

## 理解反向传播

反向传播是一个局部的过程。每一个门（有+、-、\*、/符号）都能得到一些输入和产出两个东西：根据输入的计算值和输入的梯度。两者都可以独立计算，并且不依赖整个电路图的其他元素。然而，当向前传递进行完成之后，再向后传递的过程中，会计算每一个输出值对于最终输出值的梯度，并且利用链式规则用门处的梯度（此时是门对于最终结果的梯度）乘以门处每一个输入的梯度，这样就能得到每一个输入对于最终输出的梯度，也就是 $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$ 的过程。

这个额外的乘法操作（对于每一个输入）利用链式法则将每一个感觉没用的门转为了神经网络等复杂电路中的一个齿轮。

让我们再利用例子回顾一遍，‘加’门接受参数 $[-2, 5]$ 产出值3，因为是加法操作，所以输入对这个门的输出的梯度为1。剩下的操作计算出最终结果-12。在反向传播的过程中，链式法则会不断应用。在加门处（它是后面乘法门的输入），它对于最终结果的梯度是-4。然后我们接续计算加

门的输入对于最终结果的输入，也就是  $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$  的过程，就计算完成了。可以看出x、y对于最终输出的梯度是符合要求的。当希望最终的输出值更高时，那我们就希望加门处的值更低(因为z那里有个负符号)，并且我们假设就是4，而刚刚我们计算出来了:x,y的梯度都是负数，也就是说当x,y减少时，会使得加门处的值减少，接着最终输出的值会增加，这过程符合预期。

在反向传播中，门处梯度的正负号表明了输入和最终结果的关系。为了使最终结果增加，我们可以清楚的看到应该减少还是增加门处的值。  
(这段话感觉很奇怪，感觉不好翻译)

(未完，可能不续了)

#cs231n


◀ 【译】cs231n第三课：优化 - 随机梯度下降

【译】第一章：MySQL的结构与历史（HP-MySQL 3rd） ▶

0条评论

还没有评论，沙发等你来抢

社交帐号登录: [微信](#) [微博](#) [QQ](#) [人人](#) [更多»](#)



说点什么吧...

发布

zhangyang's blog正在使用多说

© 2017 ♥ Zhang Yang

由 [Hexo](#) 强力驱动 | 主题 - [NexT.Pisces](#)