深度学习中的矩阵微积分学

Dezeming Family

Dezeming

Art. No 0
ISBN 000–00–0000–00–0
Edition 0.0

Cover design by Dezeming Family

Published by Dezeming
Printed in China

目录

前言及简介

DezemingFamily 系列书和小册子因为是电子书，所以可以很方便地进行修改和重新发布。如果您获得了 *DezemingFamily* 的系列书，可以从我们的网站 *[https://dezeming.top/]* 找到最新版。对书的内容建议和出现的错误欢迎在网站留言。

0.1 本书前言

矩阵微积分在深度学习中具有很重要的作用，尤其是涉及更一般化的张量和矩阵运算时。[1] 是一个很好的矩阵微积分学习教材（仅限于对向量微分，不包括对矩阵微分），但有些地方的描述和理解仍然需要仔细思考和分析。本文根据 [1] 来撰写，试图解释神经网络中所有你需要的矩阵演算，以便理解深层神经网络的训练。

这里对 [1] 进行一些初步介绍。[1] 的基础是微积分 1，提供链接帮助读者在需要时更新必要的数学知识。请注意，在开始学习训练并在实践中使用深度学习之前，您无需理解本材料；更确切地说，本教材面向的是那些已经熟悉神经网络基础知识的人，他们希望加深对基础数学的理解。如果你在某个时候遇到了困难，不要担心——只需返回并重读上一节，试着写下一些例子。如果你仍然感到困惑，可以在论坛 [2] 上询问。

本文并不是对 [1] 的直接翻译，所以如果想按照根据原文进行学习的话，建议直接去网站 [1] 进行学习。本文只是在 [1] 的基础上，删除了一些基础内容，并增加了一些更容易理解的例子和解释。

之所以名字叫“深度学习中的矩阵微积分学”而不是简单叫“矩阵微积分学”，是因为这里并没有涉及更进一步的矩阵微积分的内容——矩阵函数对矩阵求导，而这些内容在控制论中则是比较常见的。我们只关注与深度学习有关的部分，也就是矩阵-向量微积分。

1. 向量微积分学与偏导

本章简单介绍一下神经网络，然后介绍偏导和梯度方面的一点内容。

1.1 简介

即使不知道矩阵微积分，我们也可以很容易地通过现代深度学习库内置的自动微分，通过 PyTorch 库以最少量的微积分水平称为世界级的深度学习研究者。但是，如果你想真正了解这些库的情况，阅读讨论模型训练技术的最新进展，则需要了解矩阵微积分领域的很多内容。

对于一个线性结构：$z(\mathbf{x}) = \sum_i^n w_i x_i + b = \mathbf{w} \cdot \mathbf{x} + b$，函数 $z(\mathbf{x})$ 被叫做单位仿射函数 (unit's affine function)，之后，$z(\mathbf{x})$ 输入一个矫正器中（即 rectified linear unit，也就是 ReLU 函数：$max(0, z(\mathbf{x}))$），这种计算单元叫做人工神经元 (artificial neuron)。

神经网络由许多这样的单元组成，组织成多个神经元集合，称为层 (layers)。一层单位的激活成为下一层单位的输入。最后一层中一个或多个单元的激活称为网络输出。

我们需要训练的参数，其实就是 $\mathbf{w}$ 和 b，对损失函数求导，从而运用梯度下降算法。我们在实践中可以把求导当做一种函数映射 (map)。

1.2 偏导与向量微积分

对于一个多元函数，例如 $f(x,y)$，我们可以使用偏导来表示它的导数，把偏导组织成一个行向量 (horizontal vector) 的形式，我们把这个行向量称为 $f(x,y)$ 的梯度：

$$\nabla f(x,y) = \left[\frac{\partial f(x,y)}{\partial x}, \frac{\partial (x,y)}{\partial y}\right] \tag{1.2.1}$$

梯度属于向量微积分的一部分部分，它处理将 n 个标量参数映射到单个标量的函数 $a = f(x_1, x_2, ...)$。

本章的内容暂时就介绍这些，下一章我们同时考虑多个函数的导数，并推导出矩阵微积分的内容。

2. 矩阵微积分学

本章描述矩阵微积分学，从多个函数的微积分引入到雅克比矩阵。本章内容较为简单，都是一些基本定义，但需要对各种形式掌握熟练，否则就会在链式法则章节中迷失自我。

2.1 多个函数的微积分

我们现在对两个函数求偏导，注意这两个函数都有着同样的自变量 x 和 y：

$$a = f(x,y)$$
$$b = g(x,y)$$

把偏导的梯度向量排列，就能构成一个矩阵，其中梯度是矩阵的行：

$$\mathcal{J} = \begin{bmatrix} \nabla f(x,y) \\ \nabla g(x,y) \end{bmatrix} = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} & \frac{\partial f(x,y)}{\partial y} \\ \frac{\partial g(x,y)}{\partial x} & \frac{\partial g(x,y)}{\partial y} \end{bmatrix} \tag{2.1.1}$$

这个矩阵我们称为雅克比矩阵 (Jacobian matrix)。

我们这种表示方式叫做分子布局 (numerator layout)，有许多著作和软件会使用分母布局 (denominator layout)，其实这就是分子布局的矩阵转置：

$$\begin{bmatrix} \frac{\partial f(x,y)}{\partial x} & \frac{\partial f(x,y)}{\partial y} \\ \frac{\partial g(x,y)}{\partial x} & \frac{\partial g(x,y)}{\partial y} \end{bmatrix}^T = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} & \frac{\partial g(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} & \frac{\partial g(x,y)}{\partial y} \end{bmatrix} \tag{2.1.2}$$

2.2 雅克比的一般化

为了定义更一般形式上的雅克比矩阵，我们把多元参数定义为一个向量：

$$f(x,y,z) \Longrightarrow f(\mathbf{x})$$

小写的粗体 $\mathbf{x}$ 表示向量，一般体 x 为标量，x_i 表示向量 $\mathbf{x}$ 的第 i 个元素。我们可以定义向量的朝向 (orientation)，在默认情况下，设向量都是垂直的，比如 n 维向量的尺寸 (size) 是 $n \times 1$：

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{2.2.1}$$

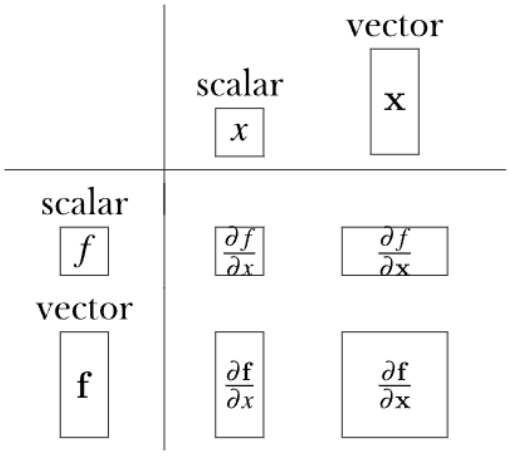
对于多个标量函数，可以把它们组合到一个向量里。令 $\mathbf{y} = \mathbf{f}(\mathbf{x})$ 是一个由 m 个多元标量函数构成的向量，把 $\mathbf{x}$ 作为输入。设 $n = |\mathbf{x}|$ 是 $\mathbf{x}$ 的基数 (cardinality)，每个 f_i 都返回一个标量值：

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{bmatrix} \tag{2.2.2}$$

我们考虑 $m = n$ 这种很常见的情况，雅克比矩阵就是这 $m \times n$ 的可能的偏导的集合（m 行 n 列），也可以认为是与 $\mathbf{x}$ 有关的 m 个梯度：

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \nabla f_1(\mathbf{x}) \\ \nabla f_2(\mathbf{x}) \\ \cdots \\ \nabla f_m(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial \mathbf{x}} f_1(\mathbf{x}) \\ \frac{\partial}{\partial \mathbf{x}} f_2(\mathbf{x}) \\ \cdots \\ \frac{\partial}{\partial \mathbf{x}} f_m(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x_1} f_1(\mathbf{x}) & \frac{\partial}{\partial x_2} f_1(\mathbf{x}) & \cdots & \frac{\partial}{\partial x_n} f_1(\mathbf{x}) \\ \frac{\partial}{\partial x_1} f_2(\mathbf{x}) & \frac{\partial}{\partial x_2} f_2(\mathbf{x}) & \cdots & \frac{\partial}{\partial x_n} f_2(\mathbf{x}) \\ & & \ddots & \\ \frac{\partial}{\partial x_1} f_m(\mathbf{x}) & \frac{\partial}{\partial x_2} f_m(\mathbf{x}) & \cdots & \frac{\partial}{\partial x_n} f_m(\mathbf{x}) \end{bmatrix} \tag{2.2.3}$$

对于 m 个方程的雅克比矩阵一共有 m 行。雅克比矩阵的形式一般可以参考如下：



对于 $f_i(\mathbf{x}) = x_i$ 的恒等函数 $\mathbf{f}(\mathbf{x}) = \mathbf{x}$，我们可以计算得到它的雅克比矩阵（这里的 m 等于 n）

是一个单位矩阵：

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial}{\partial x_1} f_1(\mathbf{x}) & \frac{\partial}{\partial x_2} f_1(\mathbf{x}) & \cdots & \frac{\partial}{\partial x_n} f_1(\mathbf{x}) \\ \frac{\partial}{\partial x_1} f_2(\mathbf{x}) & \frac{\partial}{\partial x_2} f_2(\mathbf{x}) & \cdots & \frac{\partial}{\partial x_n} f_2(\mathbf{x}) \\ & & \ddots & \\ \frac{\partial}{\partial x_1} f_m(\mathbf{x}) & \frac{\partial}{\partial x_2} f_m(\mathbf{x}) & \cdots & \frac{\partial}{\partial x_n} f_m(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x_1} x_1 & \frac{\partial}{\partial x_2} x_2 & \cdots & \frac{\partial}{\partial x_n} x_1 \\ \frac{\partial}{\partial x_1} x_2 & \frac{\partial}{\partial x_2} x_2 & \cdots & \frac{\partial}{\partial x_n} x_2 \\ & & \ddots & \\ \frac{\partial}{\partial x_1} x_n & \frac{\partial}{\partial x_2} x_n & \cdots & \frac{\partial}{\partial x_n} x_n \end{bmatrix} \tag{2.2.4}$$

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & 1 \end{bmatrix} = I \tag{2.2.5}$$

在继续之前，确保你可以推导出上面的每一步。如果被卡住，只需考虑矩阵中的每个元素，并应用一般的标量导数规则。这是一个普遍有用的技巧：将向量表达式简化为一组标量表达式，然后获取所有部分，最后将结果适当地组合成向量和矩阵。

我们在理解上面的计算时，一定要充分注意 $\mathbf{x}$ 是一个竖直的向量，$\mathbf{x}^T$ 才是水平的向量。

2.3 按元素的二元操作

向量中，按元素的二元操作 (Element-wise binary operations)，例如向量加减法，以及把向量与标量相乘。

所谓“按元素二元操作”，只是对每个向量的第一项应用一个运算，然后得到输出的第一项，之后对输入的第二项应用一个运算来得到输出的第二项，以此类推。这种操作中，要求的是输入和输出匹配，也就是说，输入为 n 元向量，输出也要是 n 元向量。深度学习中经常出现的例子有 $\max(\mathbf{w},\mathbf{x})$ 和 $\mathbf{w} > \mathbf{x}$（返回由 1 和 0 组成的向量）。

我们把“按元素二元操作”用符号进行一般化：

$$\mathbf{y} = \mathbf{f}(\mathbf{w}) \bigcirc \mathbf{g}(\mathbf{x}) \tag{2.3.1}$$

其中，$m = n = |\mathbf{y}| = |\mathbf{w}| = |\mathbf{x}|$。这里的 $\bigcirc$ 符号是元素运算符 (element-wise operator)，例如 +，而不是函数合成运算符 $\circ$（$(f \circ g)(x) = f(g(x))$）。

$$\mathbf{y} = \mathbf{f}(\mathbf{w}) \bigcirc \mathbf{g}(\mathbf{x}) = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x}) \\ f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x}) \end{bmatrix} \tag{2.3.2}$$

上式直接写为了 n 个式子，这是因为元素运算符给出 $m = n$ 大小的向量结果（输入匹配输出）。

对 $\mathbf{w}$ 求导，得到与 $\mathbf{w}$ 相关的雅克比矩阵：

$$\begin{aligned}\mathcal{J}_{\mathbf{w}} &= \frac{\partial \mathbf{y}}{\partial \mathbf{w}} \\ &= \begin{bmatrix} \frac{\partial}{\partial w_1}\Big(f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x})\Big) & \frac{\partial}{\partial w_2}\Big(f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x})\Big) & \cdots & \frac{\partial}{\partial w_n}\Big(f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x})\Big) \\ \frac{\partial}{\partial w_1}\Big(f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x})\Big) & \frac{\partial}{\partial w_2}\Big(f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x})\Big) & \cdots & \frac{\partial}{\partial w_n}\Big(f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x})\Big) \\ & & \ddots & \\ \frac{\partial}{\partial w_1}\Big(f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x})\Big) & \frac{\partial}{\partial w_2}\Big(f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x})\Big) & \cdots & \frac{\partial}{\partial w_n}\Big(f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x})\Big) \end{bmatrix}\end{aligned} \tag{2.3.3}$$

以及对 $\mathbf{x}$ 求导，得到与 $\mathbf{x}$ 相关的雅克比矩阵：

$$\begin{aligned}\mathcal{J}_{\mathbf{x}} &= \frac{\partial \mathbf{y}}{\partial \mathbf{x}} \\ &= \begin{bmatrix} \frac{\partial}{\partial x_1}\Big(f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x})\Big) & \frac{\partial}{\partial x_2}\Big(f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x})\Big) & \cdots & \frac{\partial}{\partial x_n}\Big(f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x})\Big) \\ \frac{\partial}{\partial x_1}\Big(f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x})\Big) & \frac{\partial}{\partial x_2}\Big(f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x})\Big) & \cdots & \frac{\partial}{\partial x_n}\Big(f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x})\Big) \\ & & \ddots & \\ \frac{\partial}{\partial x_1}\Big(f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x})\Big) & \frac{\partial}{\partial x_2}\Big(f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x})\Big) & \cdots & \frac{\partial}{\partial x_n}\Big(f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x})\Big) \end{bmatrix}\end{aligned} \tag{2.3.4}$$

这是一个相当复杂的问题，但幸运的是，雅可比矩阵通常是一个对角矩阵：一个除了对角之外处处为零的矩阵。因为这大大简化了雅可比矩阵，所以让我们详细研究雅可比矩阵何时简化为 element-wise 操作的对角矩阵。

在对角雅克比中，所有非对角元素都是 0：

$$\frac{\partial}{\partial w_j}\Big(f_i(\mathbf{w}) \bigcirc g_i(\mathbf{x})\Big) = \mathbf{0} \quad j \neq i \tag{2.3.5}$$

这在什么条件下会发生呢？精确地说，当 f_i 和 g_i 相对于 w_j 来说是常量的时候：

$$\frac{\partial}{\partial w_j} f_i(\mathbf{w}) = \frac{\partial}{\partial w_j} g_i(\mathbf{x}) = \mathbf{0} \tag{2.3.6}$$

不管这些运算符是什么，如果它们的偏导都是 0，则运算结果就都会是 0：$\mathbf{0} \bigcirc \mathbf{0} = \mathbf{0}$。那么，当 f_i 和 g_i 都不是 w_j 的函数时，偏导就都是 0。对于 element-wise 运算，意味着 f_i 纯粹是 w_i 的函数，而且 g_i 也纯粹是 x_i 的函数，例如 $\mathbf{w} + \mathbf{x}$ 就是逐个 w_i 与 x_i 相加的结果。

因此，$f_i(\mathbf{w}) \bigcirc g_i(\mathbf{x})$ 退化为 $f_i(w_i) \bigcirc g_i(x_i)$，也就是说，目标也退化为了：

$$\frac{\partial}{\partial w_j} f_i(w_i) = \frac{\partial}{\partial w_j} g_i(x_i) = 0 \tag{2.3.7}$$

注意，写作 $f_i(w_i)$ 和 $g_i(x_i)$ 看起来与 f_i 和 g_i 这种向量函数是相违背的，按理来说我们应该写作 $\hat{f}_i(w_i) = f_i(\mathbf{w})$，但这可能会使方程看起来很混乱，所以就算了（反正程序员也会乐意接受重载函数）。我们把上述的约束条件叫做“按元素对角条件 (element-wise diagonal condition)”。在这种条件下，对角线雅克为：

$$\frac{\partial \mathbf{y}}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial}{\partial w_1}\Big(f_1(w_1) \bigcirc g_1(x_1)\Big) & & & \\ & \frac{\partial}{\partial w_2}\Big(f_2(w_2) \bigcirc g_2(x_2)\Big) & & 0 \\ & & \ddots & \\ 0 & & & \frac{\partial}{\partial w_n}\Big(f_n(w_n) \bigcirc g_n(x_n)\Big) \end{bmatrix} \tag{2.3.8}$$

或者简写为：

$$\frac{\partial \mathbf{y}}{\partial \mathbf{w}} = diag\Big(\frac{\partial}{\partial w_1}\Big(f_1(w_1) \bigcirc g_1(x_1)\Big), \frac{\partial}{\partial w_2}\Big(f_2(w_2) \bigcirc g_2(x_2)\Big), ..., \frac{\partial}{\partial w_n}\Big(f_n(w_n) \bigcirc g_n(x_n)\Big)\Big) \tag{2.3.9}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = diag\Big(\frac{\partial}{\partial x_1}\Big(f_1(w_1) \bigcirc g_1(x_1)\Big), \frac{\partial}{\partial x_2}\Big(f_2(w_2) \bigcirc g_2(x_2)\Big), ..., \frac{\partial}{\partial x_n}\Big(f_n(w_n) \bigcirc g_n(x_n)\Big)\Big) \tag{2.3.10}$$

由于我们一般只是用简单的向量运算，所以 $\mathbf{f}(\mathbf{w})$ 通常只是 $\mathbf{w}$，也就是说 $f_i(w_i) = w_i$，因此：

$$\mathbf{f}(\mathbf{w}) + \mathbf{g}(\mathbf{x}) \Longrightarrow y_i = f_i(\mathbf{w}) + g_i(\mathbf{x}) \Longrightarrow y_i = f_i(w_i) + g_i(x_i) \tag{2.3.11}$$

单个运算的偏导就是：

$$\frac{\partial}{\partial w_i}\Big(f_i(w_i) + g_i(x_i)\Big) = \frac{\partial}{\partial w_i}(w_i + x_i) = 1 + 0 = 1 \tag{2.3.12}$$

$$\frac{\partial}{\partial x_i}\Big(f_i(w_i) + g_i(x_i)\Big) = \frac{\partial}{\partial x_i}(w_i + x_i) = 0 + 1 = 1 \tag{2.3.13}$$

于是我们得到 $\frac{\partial(\mathbf{w}+\mathbf{x})}{\partial \mathbf{w}} = \frac{\partial(\mathbf{w}+\mathbf{x})}{\partial \mathbf{x}} = I$。我们可以依次导出一般的 element-wise 二元运算的雅克比：

表 2.1: element-wise 二元运算的雅克比

Op	与 $\mathbf{w}$ 有关的偏微分
$+$	$\frac{\partial(\mathbf{w}+\mathbf{x})}{\partial \mathbf{w}} = diag\Big(\cdots \frac{\partial(w_i+x_i)}{\partial w_i} \cdots\Big) = diag\Big(\overrightarrow{1}\Big) = I$
$-$	$\frac{\partial(\mathbf{w}-\mathbf{x})}{\partial \mathbf{w}} = diag\Big(\cdots \frac{\partial(w_i-x_i)}{\partial w_i} \cdots\Big) = diag\Big(\overrightarrow{1}\Big) = I$
$\otimes$	$\frac{\partial(\mathbf{w}\otimes\mathbf{x})}{\partial \mathbf{w}} = diag\Big(\cdots \frac{\partial(w_i\times x_i)}{\partial w_i} \cdots\Big) = diag\Big(\mathbf{x}\Big)$
$\oslash$	$\frac{\partial(\mathbf{w}\oslash\mathbf{x})}{\partial \mathbf{w}} = diag\Big(\cdots \frac{\partial(w_i/x_i)}{\partial w_i} \cdots\Big) = diag\Big(\cdots \frac{1}{x_i} \cdots\Big)$
Op	与 $\mathbf{x}$ 有关的偏微分
$+$	$\frac{\partial(\mathbf{w}+\mathbf{x})}{\partial \mathbf{x}} = diag\Big(\cdots \frac{\partial(w_i+x_i)}{\partial x_i} \cdots\Big) = diag\Big(\overrightarrow{1}\Big) = I$
$-$	$\frac{\partial(\mathbf{w}-\mathbf{x})}{\partial \mathbf{x}} = diag\Big(\cdots \frac{\partial(w_i-x_i)}{\partial x_i} \cdots\Big) = diag\Big(\overrightarrow{-1}\Big) = -I$
$\otimes$	$\frac{\partial(\mathbf{w}\otimes\mathbf{x})}{\partial \mathbf{x}} = diag\Big(\cdots \frac{\partial(w_i\times x_i)}{\partial x_i} \cdots\Big) = diag\Big(\mathbf{w}\Big)$
$\oslash$	$\frac{\partial(\mathbf{w}\oslash\mathbf{x})}{\partial \mathbf{x}} = diag\Big(\cdots \frac{\partial(w_i/x_i)}{\partial x_i} \cdots\Big) = diag\Big(\cdots \frac{-w_i}{x_i^2} \cdots\Big)$

在上面的公式中，$\otimes$ 和 $\oslash$ 分别表示 element-wise 乘法和除法，其中 $\otimes$ 又被叫做 Hadamard product(阿达马积)。

2.4 涉及标量展开的导数

当我们把一个标量与向量相加或者相乘的时候，我们其实是在把标量进行扩展：

$$\mathbf{y} = \mathbf{x} + z \Longrightarrow \mathbf{y} = \mathbf{x} + \overrightarrow{1}z \equiv \mathbf{f}(\mathbf{x}) + \mathbf{g}(z) \tag{2.4.1}$$

$$\mathbf{y} = \mathbf{x}z \Longrightarrow \mathbf{y} = \mathbf{x} \otimes \overrightarrow{1}z \equiv \mathbf{f}(\mathbf{x}) \otimes \mathbf{g}(z) \tag{2.4.2}$$

标量相加

标量相加，关于 $\mathbf{x}$ 的偏导是：

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = diag\Big(\cdots \frac{\partial}{\partial x_i}\Big(f_i(x_i) \bigcirc g_i(z)\Big)\cdots\Big) \tag{2.4.3}$$

$$\frac{\partial}{\partial x_i}\Big(f_i(x_i) + g_i(z)\Big) = \frac{\partial (x_i + z)}{\partial x_i} = 1 \tag{2.4.4}$$

关于 z 的偏导并不会构成一个矩阵，而只是构成一个向量，因为 z 只是一个标量：

$$\frac{\partial}{\partial z}\Big(f_i(x_i) + g_i(z)\Big) = \frac{\partial (x_i + z)}{\partial z} = 1 \tag{2.4.5}$$

于是我们得到：

$$\frac{\partial}{\partial \mathbf{x}}\Big(\mathbf{x} + z\Big) = I \tag{2.4.6}$$

$$\frac{\partial}{\partial z}\Big(\mathbf{x} + z\Big) = \overrightarrow{1} \tag{2.4.7}$$

标量相乘法

计算标量相乘也是如此，对 $\mathbf{x}$ 求导以后，得到：

$$\frac{\partial}{\partial x_i}\Big(f_i(x_i) \otimes g_i(z)\Big) = x_i \frac{\partial z}{\partial x_i} + z\frac{\partial x_i}{\partial x_i} = z \tag{2.4.8}$$

对 z 求导得到：

$$\frac{\partial}{\partial z}\Big(f_i(x_i) \otimes g_i(z)\Big) = x_i \frac{\partial z}{\partial z} + z\frac{\partial x_i}{\partial z} = x_i + 0 = x_i \tag{2.4.9}$$

所以，偏导可以表示为：

$$\frac{\partial}{\partial \mathbf{x}}\Big(\mathbf{x}z\Big) = diag(\overrightarrow{1}\, z) = Iz \tag{2.4.10}$$

$$\frac{\partial}{\partial z}\Big(\mathbf{x}z\Big) = \mathbf{x} \tag{2.4.11}$$

2.5 向量和 reduction

把一个向量中的元素相加是一个非常重要的操作，例如损失函数的计算。我们也可以用它来简化向量点乘的导数的计算，以及其他将向量化为标量的运算。

令 $y = sum(\mathbf{f}(\mathbf{x})) = \sum_{i=1}^{n} f_i(\mathbf{x})$，注意到这里我们的参数是 $\mathbf{x}$，这是要因为 sum 会用到全部的值。向量求和的梯度是：

$$\frac{\partial y}{\mathbf{x}} = \Big[\frac{\partial y}{\partial x_1}, \frac{\partial y}{\partial x_2}, \cdots, \frac{\partial y}{\partial x_n}\Big] \tag{2.5.1}$$

$$= \Big[\frac{\partial}{\partial x_1}\sum_i f_i(\mathbf{x}), \frac{\partial}{\partial x_2}\sum_i f_i(\mathbf{x}), \cdots, \frac{\partial}{\partial x_n}\sum_i f_i(\mathbf{x})\Big] \tag{2.5.2}$$

$$= \Big[\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_1}, \sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_2}, \cdots, \sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_n}\Big] \tag{2.5.3}$$

由于 $f_i(\mathbf{x}) = x_i$，梯度可以表示为：

$$\nabla y = \Big[\sum_i \frac{\partial x_i}{\partial x_1}, \sum_i \frac{\partial x_i}{\partial x_2}, \cdots, \sum_i \frac{\partial x_i}{\partial x_n}\Big] \tag{2.5.4}$$

$$= \Big[\frac{\partial x_1}{\partial x_1}, \frac{\partial x_2}{\partial x_2}, \cdots, \frac{\partial x_n}{\partial x_n}\Big] \tag{2.5.5}$$

$$= \Big[1, 1, \cdots, 1\Big] = \overrightarrow{1}^T \tag{2.5.6}$$

注意，我们得到了一个水平向量，而不是垂直向量，因此梯度就是 $\overrightarrow{1}^T$。我们必须要在计算中记住我们的向量是行向量还是列向量，否则在复杂的函数计算中就会变得很乱。

再举一个例子，令 $y = sum(\mathbf{x}z) = \sum_{i=1}^n x_i z$：

$$\frac{\partial y}{\partial \mathbf{x}} = \Big[\sum_i \frac{\partial}{\partial x_1} x_i z, \sum_i \frac{\partial}{\partial x_2} x_i z, ..., \sum_i \frac{\partial}{\partial x_n} x_i z\Big] \tag{2.5.7}$$

$$= \Big[z, z, ..., z\Big] \tag{2.5.8}$$

与标量 z 有关的导数是 1×1 的：

$$\frac{\partial y}{\partial z} = \frac{\partial}{\partial z} \sum_{i=1}^n x_i z \tag{2.5.9}$$

$$= \sum_i \frac{\partial}{\partial z} x_i z \tag{2.5.10}$$

$$= \sum_i x_i = sum(\mathbf{x}) \tag{2.5.11}$$

3. 链式法则

本章貌似是最难的一章（但其实仍然非常简单），本章描述矩阵微积分中的链式法则。

3.1 简述

我们目前还不能通过最简单的矩阵微积分法则来计算复杂函数的偏导，例如，对于嵌套表达式（比如 $sum(\mathbf{w}+\mathbf{x})$），我们需要研究如何实现向量链式法则。我们首先从单变量链式法则开始，引入一个重要的概念，叫做总导数 (total derivative)，使用它来定义单变量总导数链式法则，之后再研究向量链式法则。

链式法则是一个分而治之的策略（如 Quicksort），它将复杂的表达式分解为导数更容易计算的子表达式。它的强大之处来自这样一个事实：我们可以单独处理每个简单的子表达式，但仍然可以组合中间结果以获得正确的总体结果。

3.2 单变量链式法则简述

因为链式法则是微积分最基础的内容（虽然很多变形并没有那么直观和好理解），所以不会花太多篇幅去叙述。

对于嵌套函数，我们有两种写法：

$$y = f(g(x)) \equiv y = (f \circ g)(x) \tag{3.2.1}$$

求导为：

$$y' = f'(g(x))g'(x) \tag{3.2.2}$$

但是这种书写方法隐含了 $u = g(x)$ 这个事实，所以我们最好这么做：

$$u = g(x) \quad y = f(u) \tag{3.2.3}$$

$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dx} \tag{3.2.4}$$

有些情况下，例如 $y(x) = x + x^2$，可以看做：

$$u = x^2 \tag{3.2.5}$$

$$y(x,u) = x + u \tag{3.2.6}$$

$y(x,u)$ 意味着有多条路径从 x 到 y，为了处理这种情况，我们必须应用单变量总导数链式法则 (single-variable total-derivative)。

在自动微分中，一般会分为前向微分 (forward differentiation) 和反向微分 (backward differentiation)，从数据流 (dataflow) 的角度看，我们正在计算一个前向微分，因为它遵循正常的数据流方向，即先得到 $u = g(x)$，然后再得到 $y = f(u)$。

反向微分是我们根据结果是如何变化的来反推回去，得到方程参数 x 如何影响函数值。下面是两个定义：

$$Forward \quad differentiation:$$
$$\frac{dy}{dx} = \frac{du}{dx}\frac{dy}{du} \tag{3.2.7}$$
$$Backward \quad differentiation:$$
$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dx} \tag{3.2.8}$$

反向微分的例子

我们用一个例子来说明反向微分的工作情况，该例子就是 PyTorch 的工作原理。

首先定义一个运算流：

$$y = \ln(\sin(x^3)^2) \Longrightarrow \tag{3.2.9}$$

$$u_1 = f_1(x) = x^3 \tag{3.2.10}$$

$$u_2 = f_2(u_1) = \sin(u_1) \tag{3.2.11}$$

$$u_3 = f_3(u_2) = u_2^2 \tag{3.2.12}$$

$$u_4 = f_4(u_3) = \ln(u_3) = y \tag{3.2.13}$$

计算偏导：

$$\frac{d}{u_x}u_1 = \frac{d}{x}x^3 = 3x^2 \tag{3.2.14}$$

$$\frac{d}{u_1}u_2 = \frac{d}{u_1}\sin(u_1) = \cos(u_1) \tag{3.2.15}$$

$$\frac{d}{u_2}u_3 = \frac{d}{u_2}u_2^2 = 2u_2 \tag{3.2.16}$$

$$\frac{d}{u_3}u_4 = \frac{d}{u_3}\ln(u_3) = \frac{1}{u_3} \tag{3.2.17}$$

组合：

$$\frac{dy}{dx} = \frac{du_4}{dx} = \frac{du_4}{du_3}\frac{du_3}{du_2}\frac{du_2}{du_1}\frac{du_1}{du_x} \tag{3.2.18}$$

$$= \frac{6u_2x^2\cos(u_1)}{u_3} \tag{3.2.19}$$

然后将中间变量替换掉即可。

3.3 单变量总微分链式法则

单变量链式法则的问题在于所有函数都是单变量的，为了处理更复杂的情况，我们需要升级基本理论。

对于多变量函数：

$$u_1(x) = x^2 \tag{3.3.1}$$

$$u_2(x, u_1) = x + u_1 = y \tag{3.3.2}$$

我们可以看到，x 的一点变化给 y 带来的影响是：

$$\hat{y} = (x + \Delta x) + (x + \Delta x)^2 \tag{3.3.3}$$

因此为了得到 x 的变化对 y 的贡献，我们需要把所有的 x 的贡献都加到 y 上。与 x 有关的总导数，分为直接取决于 x 的部分和间接通过 $u_1(x)$ 的部分，也就是：

$$\frac{dy}{dx} = \frac{\partial f(x)}{\partial x} = \frac{\partial u_2(x, u_1)}{\partial x} \tag{3.3.4}$$

$$= \frac{\partial u_2}{\partial x}\frac{\partial x}{\partial x} + \frac{\partial u_2}{\partial u_1}\frac{\partial u_1}{\partial x} = \frac{\partial u_2}{\partial x} + \frac{\partial u_2}{\partial u_1}\frac{\partial u_1}{\partial x} \tag{3.3.5}$$

注意这里的 $\frac{\partial u_2}{\partial x}$ 的计算方式为：

$$\frac{\partial u_2}{\partial x} = \frac{\partial}{\partial x}(x + u_1) = 1 + 0 = 1 \tag{3.3.6}$$

由于分成了两部分，所以我们可以认为另一部分跟变量 x 无关。总导数假定所有变量都可能相互依赖，而偏导数假定除 x 以外的所有变量都是常数。

我们可以这么理解：当求与 x 有关的总导数时，其他的变量可能都与 x 有关，因此我们需要把它们的贡献都加起来。

我们来举一个乘法的例子：

$$u_1(x) = x^2 \tag{3.3.7}$$

$$u_2(x, u_1) = xu_1 \tag{3.3.8}$$

求导为：

$$\frac{\partial u_2(x, u_1)}{\partial x} = \frac{\partial x u_1}{\partial x} + \frac{\partial (x u_1)}{\partial u_1}\frac{\partial u_1}{\partial x} = u_1 + x \cdot 2x = 3x^2 \tag{3.3.9}$$

我们总结一下单变量总链式法则：

$$\frac{\partial f(u_1, u_2, \ldots, u_n)}{\partial x} = \sum_{i=1}^{n} \frac{\partial f}{\partial u_i}\frac{\partial u_i}{\partial x} \tag{3.3.10}$$

我们再进一步联想，可以把这个过程想象为一个点乘：

$$\frac{\partial f}{\partial \mathbf{u}} \cdot \frac{\partial \mathbf{u}}{\partial x} \tag{3.3.11}$$

在我们开始研究向量链式法则之前，我们需要注意，网络上经常把这些内容叫做“多变量链式法则”，但多变量仅限于中间过程，其实总过程仍然是单变量的，而且导数和参数也都是单变量的，因此，为了和矩阵微积分的内容分开，我们这里一直都叫做“单变量全导数链式法则”。

3.4 向量链式法则

我们现在开始研究向量函数和向量变量，实际上，这种形式和单变量链式法则一样简单。

与标量有关的函数向量

首先，假设函数的输入变量都是标量，许多标量函数构成一个函数向量：

$$\mathbf{y} = \mathbf{f}(x) \tag{3.4.1}$$

例如：

$$\begin{bmatrix} y_1(x) \\ y_2(x) \end{bmatrix} = \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} = \begin{bmatrix} \ln(x^2) \\ \sin(3x) \end{bmatrix} \tag{3.4.2}$$

我们设置一个中间变量：

$$\begin{bmatrix} g_1(x) \\ g_2(x) \end{bmatrix} = \begin{bmatrix} x^2 \\ 3x \end{bmatrix} \tag{3.4.3}$$

计算 $\mathbf{y} = \mathbf{f}(\mathbf{g}(x))$：

$$\begin{bmatrix} f_1(\mathbf{g}) \\ f_2(\mathbf{g}) \end{bmatrix} = \begin{bmatrix} \ln(g_1) \\ \sin(g_2) \end{bmatrix} \tag{3.4.4}$$

在与标量 x 有关的导数需要被计算为：

$$\frac{\partial \mathbf{y}}{\partial x} = \begin{bmatrix} \frac{\partial f_1(\mathbf{g})}{\partial x} \\ \frac{\partial f_2(\mathbf{g})}{\partial x} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial g_1}\frac{\partial g_1}{\partial x} + \frac{\partial f_1}{\partial g_2}\frac{\partial g_2}{\partial x} \\ \frac{\partial f_2}{\partial g_1}\frac{\partial g_1}{\partial x} + \frac{\partial f_2}{\partial g_2}\frac{\partial g_2}{\partial x} \end{bmatrix} \tag{3.4.5}$$

$$= \begin{bmatrix} \frac{1}{g_1}2x + 0 \\ 0 + \cos(g_2)3 \end{bmatrix} = \begin{bmatrix} \frac{2}{x} \\ 3\cos(3x) \end{bmatrix} \tag{3.4.6}$$

上式之所以写成下面这样，是因为为了用标量函数来表现出向量函数的样子：

$$\frac{\partial \mathbf{y}}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial g_1}\frac{\partial g_1}{\partial x} + \frac{\partial f_1}{\partial g_2}\frac{\partial g_2}{\partial x} \\ \frac{\partial f_2}{\partial g_1}\frac{\partial g_1}{\partial x} + \frac{\partial f_2}{\partial g_2}\frac{\partial g_2}{\partial x} \end{bmatrix} \tag{3.4.7}$$

我们可以拆分为：

$$\begin{bmatrix} \frac{\partial f_1}{\partial g_1}\frac{\partial g_1}{\partial x} + \frac{\partial f_1}{\partial g_2}\frac{\partial g_2}{\partial x} \\ \frac{\partial f_2}{\partial g_1}\frac{\partial g_1}{\partial x} + \frac{\partial f_2}{\partial g_2}\frac{\partial g_2}{\partial x} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial g_1} & \frac{\partial f_1}{\partial g_2} \\ \frac{\partial f_2}{\partial g_1} & \frac{\partial f_2}{\partial g_2} \end{bmatrix} \begin{bmatrix} \frac{\partial g_1}{\partial x} \\ \frac{\partial g_2}{\partial x} \end{bmatrix} = \frac{\partial \mathbf{f}}{\partial \mathbf{g}}\frac{\partial \mathbf{g}}{\partial x} \tag{3.4.8}$$

也就是说，这个雅克比是其他两个雅克比的乘积。

当我们把 x 替换为向量 $\mathbf{x}$ 时，意味着 $\frac{\partial \mathbf{g}}{\partial \mathbf{x}}$ 和 $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$ 都不再是向量，而都会变为矩阵。完整的链式法则表示为：

$$\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{g}(\mathbf{x})) = \frac{\partial \mathbf{f}}{\partial \mathbf{g}}\frac{\partial \mathbf{g}}{\partial \mathbf{x}} \tag{3.4.9}$$

注意矩阵相乘不符合交换律。

完全形式是：

$$\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{g}(\mathbf{x})) = \begin{bmatrix} \frac{\partial f_1}{\partial g_1} & \frac{\partial f_1}{\partial g_2} & \cdots & \frac{\partial f_1}{\partial g_k} \\ \frac{\partial f_2}{\partial g_1} & \frac{\partial f_2}{\partial g_2} & \cdots & \frac{\partial f_2}{\partial g_k} \\ & & \cdots & \\ \frac{\partial f_m}{\partial g_1} & \frac{\partial f_m}{\partial g_2} & \cdots & \frac{\partial f_m}{\partial g_k} \end{bmatrix} \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ & & \cdots & \\ \frac{\partial g_k}{\partial x_1} & \frac{\partial g_k}{\partial x_2} & \cdots & \frac{\partial g_k}{\partial x_n} \end{bmatrix} \tag{3.4.10}$$

其中，$|\mathbf{f}| = m$，$|\mathbf{x}| = n$，$|\mathbf{g}| = k$，得到的雅克比矩阵是 $m \times n$ 的。

很多时候，雅克比矩阵都是方阵（$m = n$），且非对角线元素都是 0。神经网络是向量的函数，而不是函数的向量。例如在神经仿射函数 (neuron affine function，参考第一章) 中，其实是计算 $sum(\mathbf{w} \otimes \mathbf{x})$，激活函数是 $max(0, \mathbf{x})$，我们会在下一章来描述它们的导数。

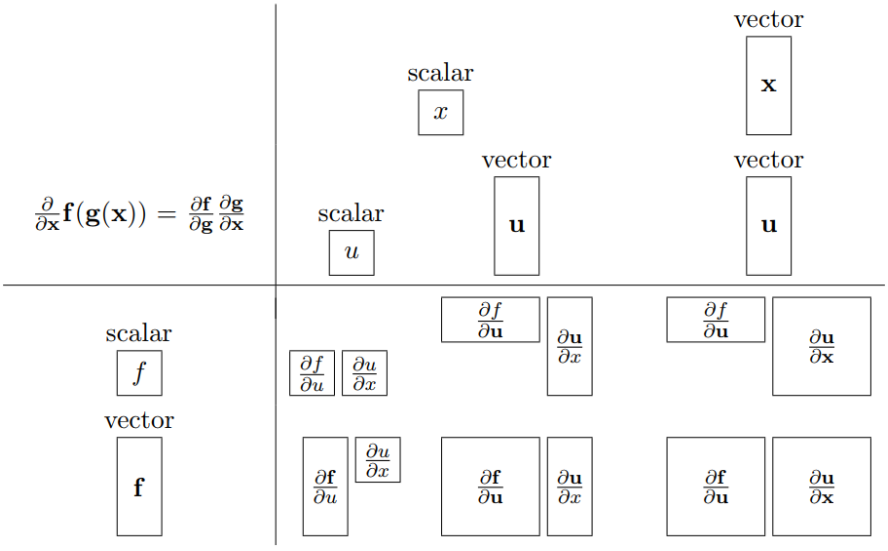
当 f_i 完全是 g_i 的函数，并且 g_i 完全是 x_i 的函数时，雅克比矩阵就是：

$$\frac{\partial \mathbf{f}}{\partial \mathbf{g}} = diag(\frac{\partial f_i}{\partial g_i}) \tag{3.4.11}$$

$$\frac{\partial \mathbf{g}}{\partial \mathbf{x}} = diag(\frac{\partial g_i}{\partial x_i}) \tag{3.4.12}$$

$$\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{g}(\mathbf{x})) = diag(\frac{\partial f_i}{\partial g_i}\frac{\partial g_i}{\partial x_i}) \tag{3.4.13}$$

下表总结了当前遇到的所有类型的函数雅克比矩阵：



注意，小方块表示的是标量，长条表示的是向量，大方块表示的是矩阵。

3.5 示例

我们分别针对复杂的情况和简单的情况举两个例子来加深对上面的过程的理解，如果大家有兴趣也可以自己构建一些例子，毕竟光看公式可能很难看懂所有细节。

示例一：复杂情况

设输入向量为 $\mathbf{x} = [x_1, x_2]^T$，第一层函数为：

$$\mathbf{g}(\mathbf{x}) = \begin{bmatrix} g_1(x_1, x_2) \\ g_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_1 + 2 \cdot x_2 \\ x_1 \cdot x_2 \end{bmatrix} \tag{3.5.1}$$

第二层函数为：

$$\mathbf{f}(\mathbf{g}) = \begin{bmatrix} f_1(g_1, g_2) \\ f_2(g_1, g_2) \end{bmatrix} = \begin{bmatrix} g_1 + g_2 \\ g_1 - g_2 \end{bmatrix} \tag{3.5.2}$$

总的函数为：

$$\mathbf{y} = \mathbf{f}(\mathbf{g}(\mathbf{x})) = \begin{bmatrix} x_1 + 2x_2 + x_1 x_2 \\ x_1 + 2x_2 - x_1 x_2 \end{bmatrix} \tag{3.5.3}$$

偏导为：

$$\frac{\partial}{\partial \mathbf{x}} \mathbf{f}(\mathbf{g}(\mathbf{x})) = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_2} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 1 + x_2 & 2 + x_1 \\ 1 - x_2 & 2 - x_1 \end{bmatrix} \tag{3.5.4}$$

现在我们应用一下链式法则：

$$\frac{\partial \mathbf{f}}{\partial \mathbf{g}} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{3.5.5}$$

$$\frac{\partial \mathbf{g}}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 2 \\ x_2 & x_1 \end{bmatrix} \tag{3.5.6}$$

$$\frac{\partial \mathbf{f}}{\partial \mathbf{g}} \frac{\partial \mathbf{g}}{\partial \mathbf{x}} = \begin{bmatrix} 1 + x_2 & 2 + x_1 \\ 1 - x_2 & 2 - x_1 \end{bmatrix} \tag{3.5.7}$$

简单示例

现在看一个简单示例：

$$\mathbf{g}(\mathbf{x}) = \begin{bmatrix} g_1(x_1, x_2) \\ g_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_1 \\ 2x_2 \end{bmatrix} \tag{3.5.8}$$

$$\mathbf{f}(\mathbf{g}) = \begin{bmatrix} f_1(g_1, g_2) \\ f_2(g_1, g_2) \end{bmatrix} = \begin{bmatrix} 2g_1 \\ g_2^2 \end{bmatrix} \tag{3.5.9}$$

总的函数为：

$$\mathbf{y} = \mathbf{f}(\mathbf{g}(\mathbf{x})) = \begin{bmatrix} 2x_1 \\ 4x_2^2 \end{bmatrix} \tag{3.5.10}$$

得到的雅克比矩阵为：

$$\frac{\partial}{\partial \mathbf{x}} \mathbf{f}(\mathbf{g}(\mathbf{x})) = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_2} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 8x_2 \end{bmatrix} \tag{3.5.11}$$

得到一个对角阵。

大家可以自行来验证一下链式法则。

4. 梯度

本节描述神经网络中的梯度。

4.1 神经元激活的梯度

对于神经元激活：

$$activation(\mathbf{x}) = max(0, \mathbf{w} \cdot \mathbf{x} + b) \tag{4.1.1}$$

这里表示全连接权重核 rectified 线性单元激活。不过，其他仿射函数，例如卷积和其他激活函数也遵循相似的逻辑。

我们关注于计算：

$$\frac{\partial}{\partial \mathbf{w}}(\mathbf{w} \cdot \mathbf{x} + b) \tag{4.1.2}$$

$$\frac{\partial}{\partial b}(\mathbf{w} \cdot \mathbf{x} + b) \tag{4.1.3}$$

所谓点乘，就是 element-wise 相乘再求和：

$$\sum_i^n (w_i x_i) = sum(\mathbf{w} \otimes \mathbf{x}) \tag{4.1.4}$$

$$\mathbf{w} \cdot \mathbf{x} = \mathbf{w}^T \mathbf{x} \tag{4.1.5}$$

根据链式法则，先写成数据流：

$$\mathbf{u} = \mathbf{w} \otimes \mathbf{x} \tag{4.1.6}$$

$$y = sum(\mathbf{u}) \tag{4.1.7}$$

偏导计算为：

$$\frac{\partial \mathbf{u}}{\partial \mathbf{w}} = \frac{\partial}{\partial \mathbf{w}}(\mathbf{w} \otimes \mathbf{x}) = diag(\mathbf{x}) \tag{4.1.8}$$

$$\frac{\partial y}{\partial \mathbf{u}} = \frac{\partial}{\partial \mathbf{u}} sum(\mathbf{u}) = \overrightarrow{1}^T \tag{4.1.9}$$

$$\frac{\partial y}{\partial \mathbf{w}} = \frac{\partial y}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \mathbf{w}} = \overrightarrow{1}^T diag(\mathbf{x}) = \mathbf{x}^T \tag{4.1.10}$$

我们来检查一下链式法则：

$$y = \mathbf{w} \cdot \mathbf{x} = \sum_i^n (w_i x_i) \tag{4.1.11}$$

$$\frac{\partial y}{\partial w_j} = \frac{\partial}{\partial w_j} \sum_i (w_i x_i) = x_j \tag{4.1.12}$$

$$\frac{\partial y}{\partial \mathbf{w}} = [x_1, ..., x_n] = \mathbf{x}^T \tag{4.1.13}$$

最后，计算完整的线性结构：

$$\frac{\partial y}{\partial \mathbf{w}} = \frac{\partial}{\partial \mathbf{w}} \mathbf{w} \cdot \mathbf{x} + \frac{\partial}{\partial \mathbf{w}} b = \mathbf{x}^T + \overrightarrow{0}^T = \mathbf{x}^T \tag{4.1.14}$$

$$\frac{\partial y}{\partial b} = \frac{\partial}{\partial b} \mathbf{w} \cdot \mathbf{x} + \frac{\partial}{\partial b} b = 0 + 1 = 1 \tag{4.1.15}$$

现在再看激活结构：

$$max(0, \mathbf{x}) = \begin{bmatrix} max(0, x_1) \\ max(0, x_2) \\ ... \\ max(0, x_n) \end{bmatrix} \tag{4.1.16}$$

$$\frac{\partial}{\partial x} max(0, z) = \begin{cases} 0 & z \leq 0 \\ \frac{dz}{dz} = 1 & z > 0 \end{cases} \tag{4.1.17}$$

所以得到：

$$\frac{\partial}{\partial \mathbf{x}} max(0, \mathbf{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} max(0, x_1) \\ \frac{\partial}{\partial x_2} max(0, x_2) \\ ... \\ \frac{\partial}{\partial x_n} max(0, x_n) \end{bmatrix} \tag{4.1.18}$$

所以把总的过程写为一个数据流：

$$z(\mathbf{w}, b, \mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b \tag{4.1.19}$$

$$activation(z) = max(0, z) \tag{4.1.20}$$

链式法则计算如下：

$$\frac{\partial activation}{\partial \mathbf{w}} = \frac{\partial activation}{\partial z}\frac{\partial z}{\partial \mathbf{w}} = \begin{cases} 0\mathbf{x}^T = \overrightarrow{0}^T & \mathbf{w}\cdot\mathbf{x}+b \le 0 \\ 1\mathbf{x}^T = \mathbf{x}^T & \mathbf{w}\cdot\mathbf{x}+b > 0 \end{cases} \tag{4.1.21}$$

$$\frac{\partial activation}{\partial b} = \begin{cases} 0\frac{\partial z}{\partial b} = 0 & \mathbf{w}\cdot\mathbf{x}+b \le 0 \\ 1\frac{\partial z}{\partial b} = 1 & \mathbf{w}\cdot\mathbf{x}+b > 0 \end{cases} \tag{4.1.22}$$

4.2 损失函数的梯度

设有一堆向量样本：

$$X = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N]^T \tag{4.2.1}$$

$$N = |X| \tag{4.2.2}$$

定义损失函数：

$$C(\mathbf{w}, b, X, \mathbf{y}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - activation(\mathbf{x}_i))^2 \tag{4.2.3}$$

数据流表示如下：

$$u(\mathbf{w}, b, \mathbf{x}) = max(0, \mathbf{w}\cdot\mathbf{x}+b) \tag{4.2.4}$$

$$v(y, u) = y - u \tag{4.2.5}$$

$$C(v) = \frac{1}{N}\sum_{i=1}^{N} v^2 \tag{4.2.6}$$

4.2.1 损失函数对权重的梯度

易知：

$$\frac{\partial v(y,u)}{\partial \mathbf{w}} = \frac{\partial}{\partial \mathbf{w}}(y-u) = \overrightarrow{0}^T - \frac{\partial u}{\partial \mathbf{w}} = \begin{cases} \overrightarrow{0}^T & \mathbf{w}\cdot\mathbf{x}+b \le 0 \\ -\mathbf{x}^T & \mathbf{w}\cdot\mathbf{x}+b > 0 \end{cases} \tag{4.2.7}$$

之后，经过一系列计算，得到：

$$\frac{\partial C(v)}{\partial \mathbf{w}} = \frac{\partial}{\partial \mathbf{w}}\frac{1}{N}\sum_{i=1}^{N} v^2 \tag{4.2.8}$$

$$= \begin{cases} \overrightarrow{0}^T & \mathbf{w}\cdot\mathbf{x}+b \le 0 \\ \frac{2}{N}\sum_{i=1}^{N}(\mathbf{w}\cdot\mathbf{x}_i + b - y_i)\mathbf{x}_i^T & \mathbf{w}\cdot\mathbf{x}+b > 0 \end{cases} \tag{4.2.9}$$

设每一个样本的误差为：

$$e_i = \mathbf{w}\cdot\mathbf{x}_i + b - y_i \tag{4.2.10}$$

就可以写为：

$$\frac{\partial C}{\partial \mathbf{w}} = \frac{2}{N}\sum_{i=1}^{N} e_i \mathbf{x}_i^T \tag{4.2.11}$$

更新参数的方法：

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \frac{\partial C}{\partial \mathbf{w}} \tag{4.2.12}$$

4.2.2 损失函数对偏置的梯度

易知：

$$\frac{\partial v(y,u)}{\partial b} = \frac{\partial}{\partial b}(y-u) = 0 - \frac{\partial u}{\partial b} = \begin{cases} 0 & \mathbf{w}\cdot\mathbf{x}+b \le 0 \\ -1 & \mathbf{w}\cdot\mathbf{x}+b > 0 \end{cases} \tag{4.2.13}$$

可以得到偏微分：

$$\frac{\partial C(v)}{\partial b} = \frac{\partial}{\partial b}\frac{1}{N}\sum_{i=1}^{N} v^2 \tag{4.2.14}$$

$$= \begin{cases} 0 & \mathbf{w}\cdot\mathbf{x}+b \le 0 \\ \frac{2}{N}\sum_{i=1}^{N}(\mathbf{w}\cdot\mathbf{x}_i + b - y_i) & \mathbf{w}\cdot\mathbf{x}+b > 0 \end{cases} \tag{4.2.15}$$

设每一个样本的误差为：

$$e_i = \mathbf{w}\cdot\mathbf{x}_i + b - y_i \tag{4.2.16}$$

这样可以得到：

$$\frac{\partial C}{\partial b} = \frac{2}{N}\sum_{i=1}^{N} e_i \tag{4.2.17}$$

以及更新参数：

$$b_{t+1} = b_t - \eta \frac{\partial C}{\partial b} \tag{4.2.18}$$

4.3 总结

借助链式法则，我们就可以很容易地得到神经网络的导数形式——矩阵微积分的雅克比矩阵。

本文的内容难度较低，且不涉及更为复杂的矩阵函数对矩阵求导，所以例子更加直观。如果想继续学习更深入的内容，公式上可以参考《矩阵分析与应用-张贤达》（个人认为张贤达的书里面有错误，在《Jocabian 雅克比矩阵与应用》一文中我有提到，而且原理证明讲得也比较少）或者 [3] 的内容。

Bibliography

[1] https://explained.ai/matrix-calculus/index.html

[2] https://forums.fast.ai/c/theory

[3] Jan R. Magnus, Heinz NeudeckerMatrix. Differential Calculus with Applications in Statistics and Econometrics. 1999. Wiley