2. 神经网络的数学基础

WU Xiaokun 吴晓堃

xkun.wu [at] gmail

2021/03/15

本章内容

张量。张量运算。 基于梯度的优化。 反向传播算法与链式求导。 实践：二维仿射变换，基本激活函数，ReLU合成法构造一般函数。

重点：张量、张量运算、三种基本激活函数、深度学习层间运算的一般形式；

难点：张量运算的几何解释、基于梯度的优化、反向传播算法、ReLU合成法、一致逼近理论。

学习目标

- 理解张量的概念；
- 掌握张量运算，并理解其几何解释；
- 理解基于梯度的优化及随机梯度下降算法；
- 理解反向传播算法的理论基础：链式求导法；
- 掌握三种基本激活函数：ReLU，Sigmoid，Tanh；
- 了解ReLU合成法构造连续函数的方法。

概念

张量

数据的容器。0D、1D、2D张量又分别成为标量、向量、矩阵。

张量运算

数据不同表示之间的变换函数。

深度学习层间运算的一般形式

$$\underline{output} = activate(\underline{GT})$$

$$\underline{GT} = \mathbf{W} * \underline{input} + \mathbf{b}$$

二维仿射变换

二维仿射变换的一般形式

$$\underline{GT} = \mathbf{W} * \underline{input} + \mathbf{b}$$

基本二维仿射变换矩阵

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & e_x & 0 \\ e_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

深度学习的几何解释

如何想象高维空间?

首先研究低维空间，归纳出规律，然后将规律泛化到高维。

深度学习的几何解释

深度学习可以解释为高维空间中非常复杂的几何变换

- 一切数据都是张量，即几何空间中的点。
- 模型的每一层对数据点做一个几何变换；而模型本身是这些变换的合成。
- 注意：几何变换必须可微，意味着从输入到输出的几何变形必须平滑且连续。

模型的可探索空间（假设空间）需要足够大

只要模型的参数足够多，就能捕捉到原始数据中所有的映射关系。想象“Ω路径”。

深度学习与神经网络

深度学习的核心在于连续的几何空间操作

- 事物之间的映射关系在几何空间中有自然的度量函数：距离。
- 并且，从计算的角度来看，处理向量空间很高效。
- 注意：大脑是否通过几何空间来实现认知，则是另一个问题。

因此也可称为：分层表示学习、层级表示学习、深度可微模型、链式几何变换。

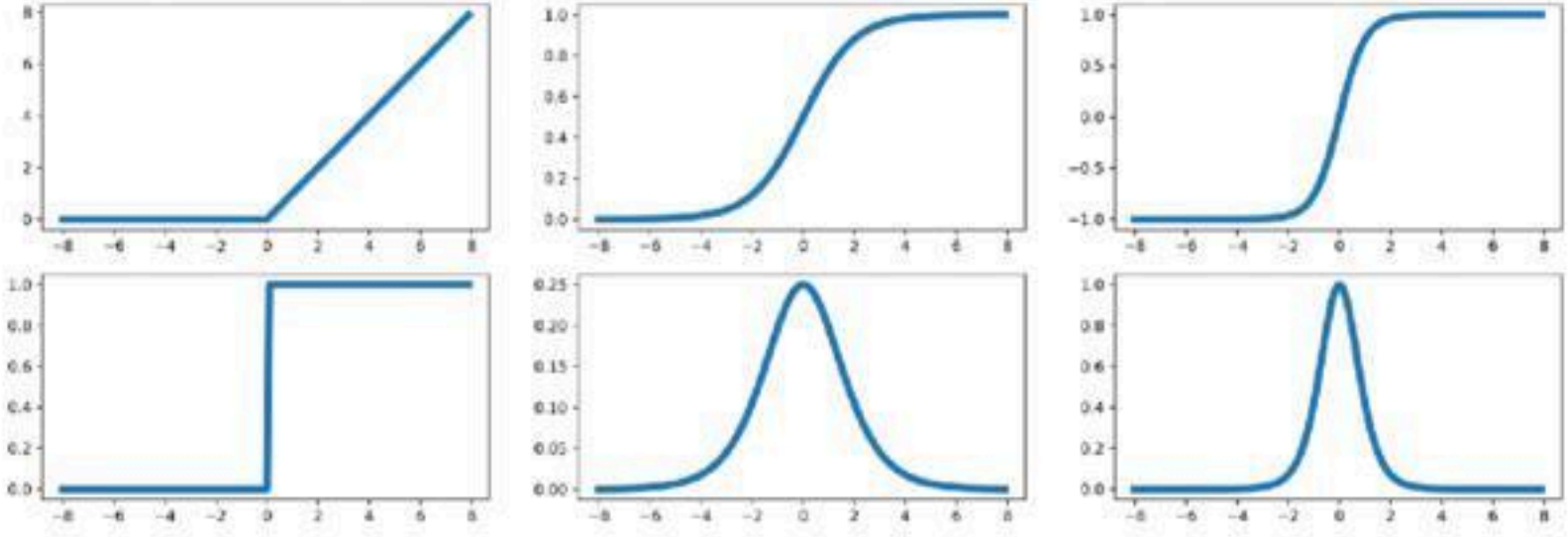
神经网络的名称纯粹是出于历史原因

- 神经网络最初来自于使用图对知识进行编码。
- 但它与神经或网络都没有关系，尤其是和大脑几乎没有任何关系。

激活函数

三种基本激活函数

ReLU，Sigmoid，Tanh。



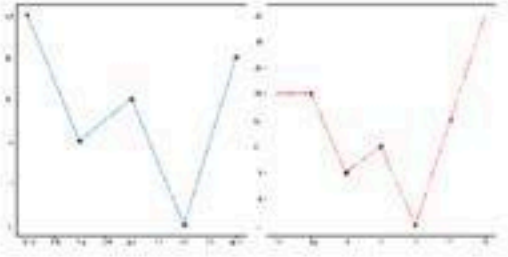
激活函数的作用

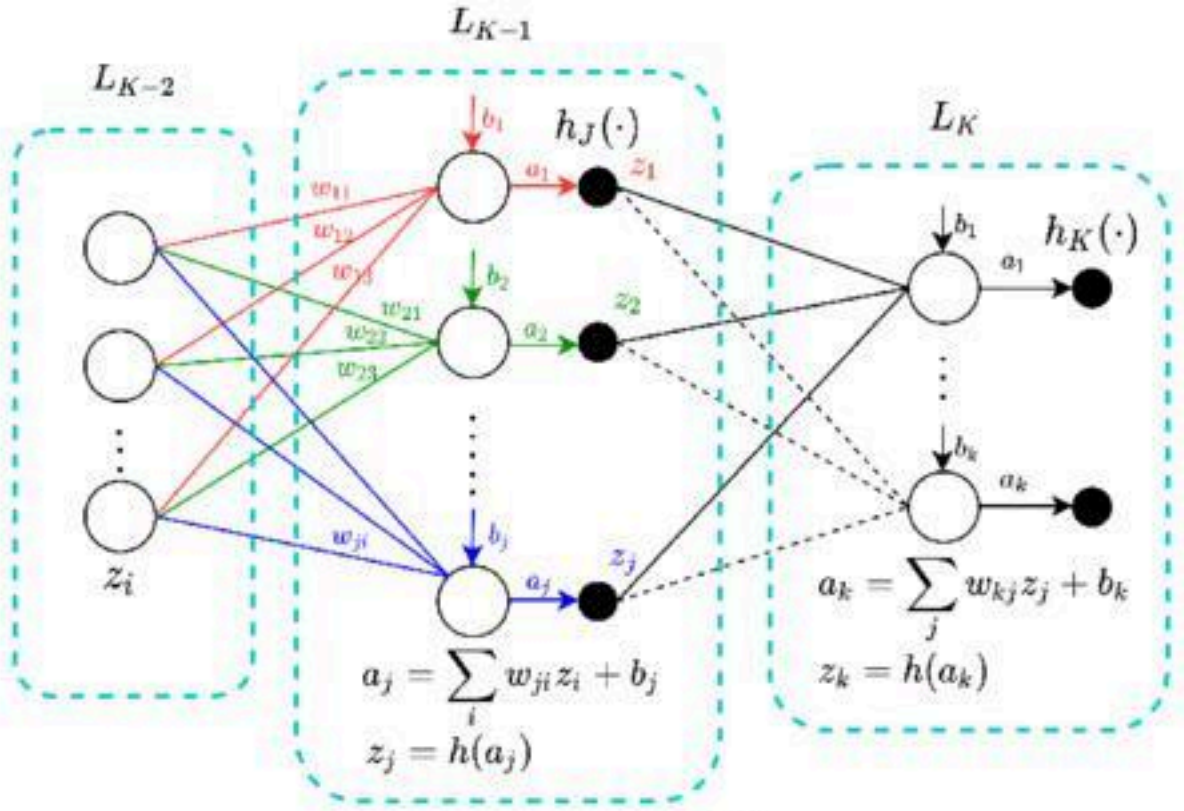
提供非线性。

ReLU合成法

ReLU 合成的一般形式

$$\sum relu(\mathbf{W} * input + \mathbf{b})$$





$$\mathbf{y} = L_K \circ L_{K-1} \circ L_{K-2} \cdots L_1(\mathbf{x})$$

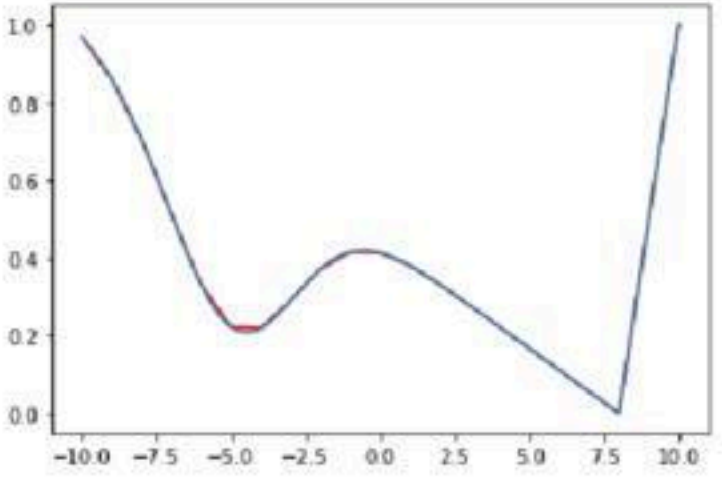
一致逼近原理

Universal approximation theorem（非常有用的废话）

In approximation theory, both and networks are known to at an .

构造法：

- 构造线性函数；
- 构造分段线性函数；
- 构造离散化的任意函数。



梯度下降法

优化目标

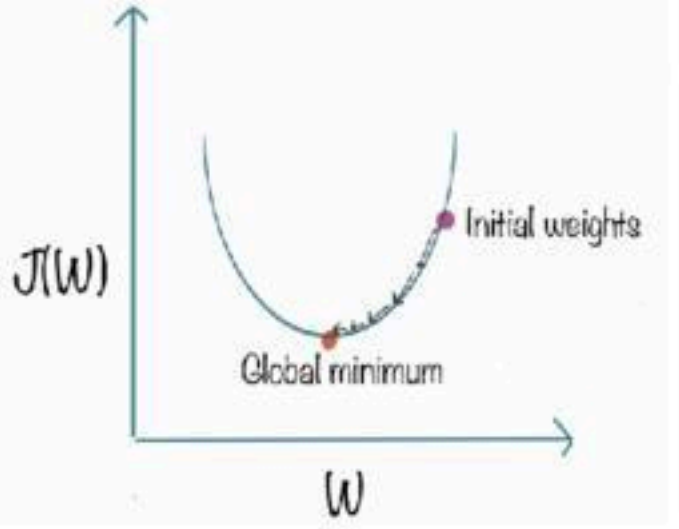
Objective: $\arg\min_{W,b} J$,

with: $J = \|\overline{y} - y\|, \overline{y} = \sum relu(\mathbf{W} * x + \mathbf{b})$

梯度下降更新

$W_1 = W_0 - \nabla J * s$

- 步长（学习率）要选取合适；
- 带动量解决收敛速度、局部极值。



反向传播算法

链式求导

将链式法则应用于神经网络梯度值的计算，得到的算法叫作反向传播算法。