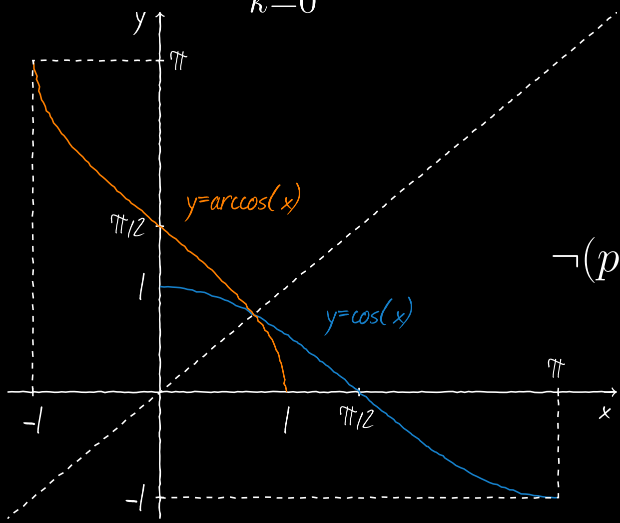
学习笔记

深度学习：从 MLP 到 GNN

李开运

version 1.1

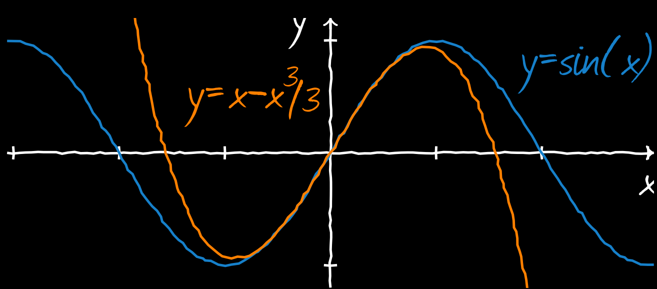
$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}$$



$$\zeta_k = |a|^{1/n} \mathrm{e}^{i(\arg(a)+2k\pi)/n}$$

$$\mathrm{e}^{i\pi} + 1 = 0$$

$$\neg(p \vee q) \equiv (\neg p) \wedge (\neg q)$$



$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Contents

1.1 多层感知机

1.2 反向传播

前向传递输入信号直至产生误差，反向传播误差信息更新权重矩阵。误差反向传播的目标是寻找一计算前馈神经网络的误差函数 $E(\boldsymbol{w})$ 的梯度的一种高效的方法。

许多实际应用中使用的误差函数，例如针对一组独立同分布的数据的最大似然方法定义的误差函数，由若干的求和式组成，每一项对应于训练集的一个数据点，即

$$E(\boldsymbol{w}) = \sum_{n=1}^{N} E_n(\boldsymbol{w}) \tag{1.1}$$

这里，我们要考虑的是计算 $\nabla E_n(\boldsymbol{w})$ 的问题。这可以使用顺序优化的方法计算，或者使用批处理方法在训练集上进行累加。

首先考虑一个简单的线性模型，其中输出 y_k 是输入变量 x_i 的线性组合，即，

$$y_k = \sum_i w_{ki} x_i \tag{1.2}$$

对于一个特定的输入模式 n，误差函数的形式为

$$E_n = \frac{1}{2} \sum_k (y_{nk} - t_{nk})^2 \tag{1.3}$$

其中 $y_{nk} = y_k(\boldsymbol{x}_n, \boldsymbol{w})$。这个误差函数关于一个权值 w_{ji} 的梯度为

$$\frac{\partial E_n}{\partial w_{ji}} = (y_{nj} - t_{nj}) x_{ni} \tag{1.4}$$

它可以表示为链接 w_{ji} 的输出端相关联的“误差信号” $y_{nj} - t_{nj}$ 和与链接的输入端相关联的变量 x_{nj} 的乘积。反向传播算法可以总结如下

1. 对于网络的一个输入向量 $\boldsymbol{x}_n$，使用下列进行正向传播，找到所有隐含单元和输出单元的激活。

$$a_j = \sum_i w_{ji} z_i \tag{1.5}$$

$$z_j = h(a_j) \tag{1.6}$$

   其中 z_i 是一个单元的激活，或者是输入。它向单元 j 发送一个链接，w_{ji} 是与这个链接关联的权值。

2. 计算所有输出单元的 δ_k

$$\delta_k = y_k - t_k \tag{1.7}$$

3. 获得网络中所有隐含单元的 δ_j

$$\begin{aligned} \delta_j \equiv \frac{\partial E_n}{\partial a_j} &= \sum_k \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial a_j} \\ &= h'(a_j) \sum_k w_{kj} \delta_k \end{aligned} \tag{1.8}$$

其中求和式的作用对象是所有向单元 j 发送链接的单元 k。注意，单元 k 可以包含其他的隐含单元和输出单元。

4. 计算导数

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i \tag{1.9}$$

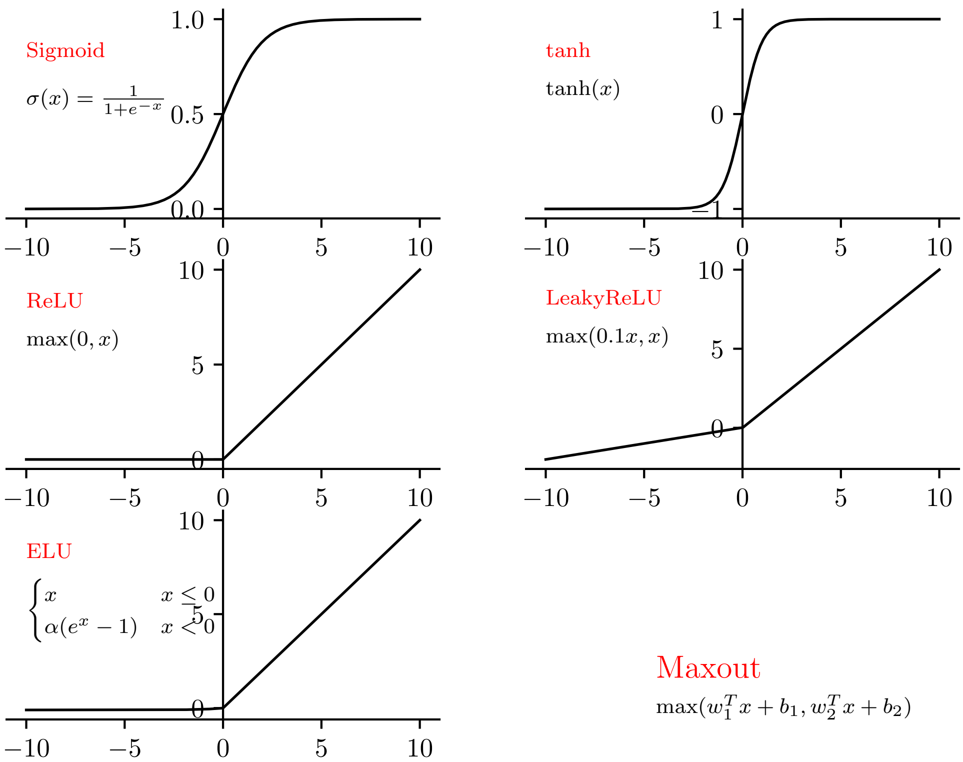
对于批处理方法，总误差函数 E 的导数可以通过下面的方式得到：对于训练集里的每个模式，重复上面的步骤，然后对所有的模式求和，即

$$\frac{\partial E}{\partial w_{ji}} = \sum_n \frac{\partial E_n}{\partial w_{ji}} \tag{1.10}$$

上面的推导中，我们隐式地假设网络中的每个隐含单元或输入单元相同的激活函数 $h(\cdot)$。

1.3 激活函数

激活函数的作用是为模型引入非线性，提高模型的表达能力。



1.4 损失函数

1.5 优化算法

1.6 过拟合

1.7 神经网络的扩展

1. 增加额外的处理层
2. 引入跨层链接
3. 稀疏网络

2.1 CNN

2.1.1 卷积

2.1.2 卷积层的前向传播

为简单起见，考虑单通道的情况

$$\boldsymbol{X} * \boldsymbol{K} = \boldsymbol{Y} \tag{2.1}$$

即

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{pmatrix} * \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix} = \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix} \tag{2.2}$$

这里

$$\begin{aligned} \begin{pmatrix} y_{11} \\ y_{12} \\ y_{21} \\ y_{22} \end{pmatrix} &= \begin{pmatrix} x_{11}k_{11} + x_{12}k_{12} + x_{21}k_{21} + x_{22}k_{22} \\ x_{12}k_{11} + x_{13}k_{12} + x_{22}k_{21} + x_{23}k_{22} \\ x_{21}k_{11} + x_{22}k_{12} + x_{31}k_{21} + x_{32}k_{22} \\ x_{22}k_{11} + x_{23}k_{12} + x_{32}k_{21} + x_{33}k_{22} \end{pmatrix} \\ &= \begin{pmatrix} x_{11} & x_{12} & x_{21} & x_{22} \\ x_{12} & x_{13} & x_{22} & x_{23} \\ x_{21} & x_{22} & x_{31} & x_{32} \\ x_{22} & x_{23} & x_{32} & x_{33} \end{pmatrix} \cdot \begin{pmatrix} k_{11} \\ k_{12} \\ k_{21} \\ k_{22} \end{pmatrix} \end{aligned} \tag{2.3}$$

所以,卷积运算最终转化为矩阵运算。需要对原始的 $\boldsymbol{X}, \boldsymbol{K}, \boldsymbol{Y}$ 进行变形操作,相应的记作 $\boldsymbol{XC}, \boldsymbol{KC}, \boldsymbol{YC}$。多通道的情况只需要在维度上将操作扩展即可。

2.1.3 卷积层的反向传播

分析 δ 误差反向传播过程可以简单的记忆为：如果神经网络 $l+1$ 层某个结点的 δ 误差要传到 l 层，我们就去找到前向传播时 $l+1$ 层的这个结点和第 l 层的哪些结点有关系，权重是多少，那么反向传播时，δ 误差就会乘上相同的权重传播回来。
为了书写方便，记

$$\begin{pmatrix} \delta_{11} \\ \delta_{12} \\ \delta_{21} \\ \delta_{22} \end{pmatrix} = \nabla \boldsymbol{YC} = \begin{pmatrix} \nabla y_{11} \\ \nabla y_{12} \\ \nabla y_{21} \\ \nabla y_{22} \end{pmatrix} \tag{2.4}$$

在反向传播中，δ 是从后面一层（一般是激活函数层或池化层）传过来的，是一个已知量，在此基础上求 $\nabla \boldsymbol{K}, \nabla \boldsymbol{X}$

1. 求 $\nabla \boldsymbol{K}$

$$\nabla \boldsymbol{KC} = \boldsymbol{XC}^T \cdot \nabla \boldsymbol{YC} \tag{2.5}$$

$\nabla \boldsymbol{KC}$ 只要 reshape 一下就可以得到 $\nabla \boldsymbol{K}$

2. 求 $\nabla \boldsymbol{X}$

根据反向传播公式，

$$\nabla \boldsymbol{X}\boldsymbol{C} = \nabla \boldsymbol{Y}\boldsymbol{C} \cdot \boldsymbol{K}\boldsymbol{C}^T \tag{2.6}$$

但是，从 $\nabla \boldsymbol{X}\boldsymbol{C}$ 还原到 $\nabla \boldsymbol{X}$ 不是一件容易的事，所以考虑新的计算方式。

根据前向传播

$$\begin{pmatrix} y_{11} \\ y_{12} \\ y_{21} \\ y_{22} \end{pmatrix} = \begin{pmatrix} x_{11}k_{11} + x_{12}k_{12} + x_{21}k_{21} + x_{22}k_{22} \\ x_{12}k_{11} + x_{13}k_{12} + x_{22}k_{21} + x_{23}k_{22} \\ x_{21}k_{11} + x_{22}k_{12} + x_{31}k_{21} + x_{32}k_{22} \\ x_{22}k_{11} + x_{23}k_{12} + x_{32}k_{21} + x_{33}k_{22} \end{pmatrix} \tag{2.7}$$

可以计算每个 x_{ij} 的导数

$$\begin{aligned} \begin{pmatrix} \nabla x_{11} \\ \nabla x_{12} \\ \nabla x_{13} \\ \nabla x_{21} \\ \nabla x_{22} \\ \nabla x_{23} \\ \nabla x_{31} \\ \nabla x_{32} \\ \nabla x_{33} \end{pmatrix} &= \begin{pmatrix} k_{22}0 + k_{21}0 + K_{12}0 + k_{11}\delta_{11} \\ k_{22}0 + k_{21}0 + K_{12}\delta_{11} + k_{11}\delta_{12} \\ k_{22}0 + k_{21}0 + K_{12}\delta_{12} + k_{11}0 \\ k_{22}0 + k_{21}\delta_{11} + K_{12}0 + k_{11}\delta_{21} \\ k_{22}\delta_{11} + k_{21}\delta_{12} + K_{12}\delta_{21} + k_{11}\delta_{22} \\ k_{22}\delta_{12} + k_{21}0 + K_{12}\delta_{22} + k_{11}0 \\ k_{22}0 + k_{21}0 + K_{12}\delta_{21} + k_{11}0 \\ k_{22}\delta_{21} + k_{21}\delta_{22} + K_{12}0 + k_{11}0 \\ k_{22}\delta_{22} + k_{21}0 + K_{12}0 + k_{11}0 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 0 & 0 & \delta_{11} \\ 0 & 0 & \delta_{11} & \delta_{12} \\ 0 & 0 & \delta_{12} & 0 \\ 0 & \delta_{11} & 0 & \delta_{21} \\ \delta_{11} & \delta_{12} & \delta_{21} & \delta_{22} \\ \delta_{12} & 0 & \delta_{22} & 0 \\ 0 & \delta_{21} & 0 & 0 \\ \delta_{21} & \delta_{22} & 0 & 0 \\ \delta_{22} & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} k_{11} \\ k_{12} \\ k_{21} \\ k_{22} \end{pmatrix} \end{aligned} \tag{2.8}$$

设上面三个矩阵分别为 $\nabla \boldsymbol{X}', \nabla \boldsymbol{Y}', \nabla \boldsymbol{K}'$，即 $\nabla \boldsymbol{X}' = \nabla \boldsymbol{Y}' \cdot \nabla \boldsymbol{K}'$。从而可见

$$\nabla \boldsymbol{X} = \begin{pmatrix} \nabla x_{11} & \nabla x_{12} & \nabla x_{13} \\ \nabla x_{21} & \nabla x_{22} & \nabla x_{23} \\ \nabla x_{31} & \nabla x_{32} & \nabla x_{33} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \delta_{11} & \delta_{12} & 0 \\ 0 & \delta_{21} & \delta_{22} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} * \begin{pmatrix} k_{22} & k_{21} \\ k_{12} & k_{11} \end{pmatrix} \tag{2.9}$$

这是一个卷积运算。不同的是对 $\nabla \boldsymbol{Y}$ 进行卷积，从后向前卷积，有的文章称为逆向卷积。

2.1.4 池化层及池化层的反向传播

2.2 ImageNet

2.3 GoogleNet

2.4 ResNet

2.5 FCN

3.1 RNN

3.1.1 RNN 前向传播

简单起见，我们考虑一个无偏差项的循环神经网络，且激活函数为恒等映射（$\phi(x)=x$）。设时间步 t 的输入为单样本 $\boldsymbol{x}_t \in \mathbb{R}^d$，标签为 y_t，那么隐藏状态 $\boldsymbol{h}_t \in \mathbb{R}^h$ 的计算表达式为

$$\boldsymbol{h}_t = \boldsymbol{W}_{hx}\boldsymbol{x}_t + \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1},$$

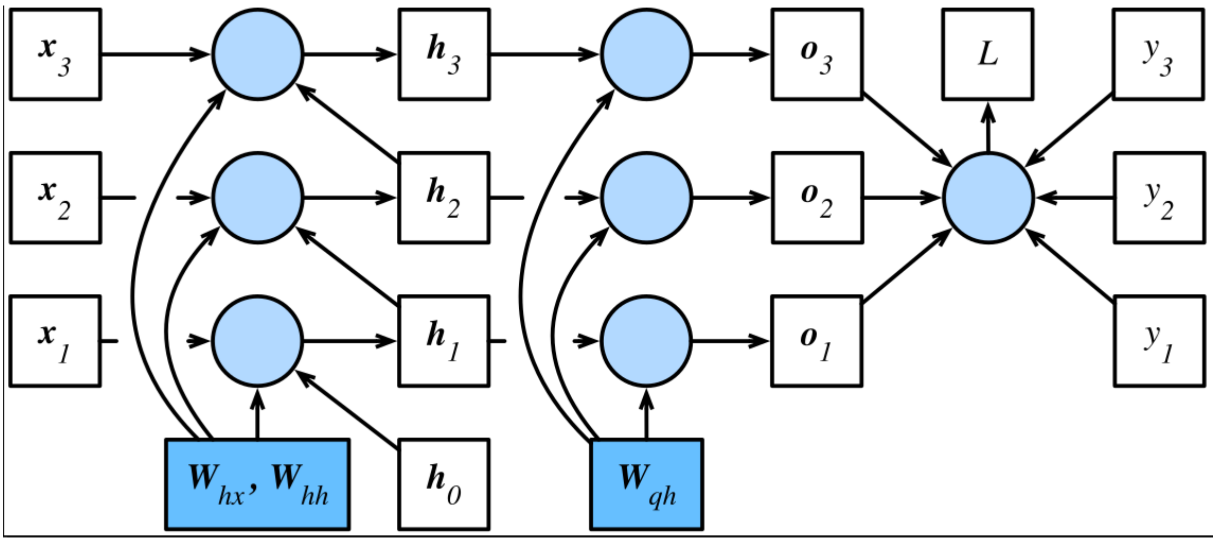
其中 $\boldsymbol{W}_{hx} \in \mathbb{R}^{h\times d}$ 和 $\boldsymbol{W}_{hh} \in \mathbb{R}^{h\times h}$ 是隐藏层权重参数。设输出层权重参数 $\boldsymbol{W}_{qh} \in \mathbb{R}^{q\times h}$，时间步 t 的输出层变量 $\boldsymbol{o}_t \in \mathbb{R}^q$ 计算为

$$\boldsymbol{o}_t = \boldsymbol{W}_{qh}\boldsymbol{h}_t.$$

设时间步 t 的损失为 $\ell(\boldsymbol{o}_t, y_t)$。时间步数为 T 的损失函数 L 定义为

$$L = \frac{1}{T}\sum_{t=1}^{T}\ell(\boldsymbol{o}_t, y_t).$$

我们将 L 称为有关给定时间步的数据样本的目标函数。



3.1.2 通过时间反向传播

模型的参数是 $\boldsymbol{W}_{hx}$, $\boldsymbol{W}_{hh}$ 和 $\boldsymbol{W}_{qh}$。链式法则的运算符用 prod 表示。

1. $\partial L/\partial \boldsymbol{W}_{qh}$

$$\frac{\partial L}{\partial \boldsymbol{W}_{qh}} = \sum_{t=1}^{T}\text{prod}\left(\frac{\partial L}{\partial \boldsymbol{o}_t}, \frac{\partial \boldsymbol{o}_t}{\partial \boldsymbol{W}_{qh}}\right) = \sum_{t=1}^{T}\frac{\partial L}{\partial \boldsymbol{o}_t}\boldsymbol{h}_t^\top. \tag{3.1}$$

2. $\partial L/\partial \boldsymbol{W}_{hx}$

$$\frac{\partial L}{\partial \boldsymbol{W}_{hx}} = \sum_{t=1}^{T}\text{prod}\left(\frac{\partial L}{\partial \boldsymbol{h}_t}, \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{W}_{hx}}\right) = \sum_{t=1}^{T}\frac{\partial L}{\partial \boldsymbol{h}_t}\boldsymbol{x}_t^\top \tag{3.2}$$

3. $\partial L/\partial \boldsymbol{W}_{hh}$

$$\frac{\partial L}{\partial \boldsymbol{W}_{hh}} = \sum_{t=1}^{T} \text{prod}\left(\frac{\partial L}{\partial \boldsymbol{h}_t}, \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{W}_{hh}}\right) = \sum_{t=1}^{T} \frac{\partial L}{\partial \boldsymbol{h}_t} \boldsymbol{h}_{t-1}^{\top}. \tag{3.3}$$

其中，目标函数有关各时间步输出层变量的梯度 $\partial L/\partial \boldsymbol{o}_t \in \mathbb{R}^q$ 很容易计算：

$$\frac{\partial L}{\partial \boldsymbol{o}_t} = \frac{\partial \ell(\boldsymbol{o}_t, y_t)}{T \cdot \partial \boldsymbol{o}_t}. \tag{3.4}$$

目标函数有关时间步隐藏状态的梯度 $\partial L/\partial \boldsymbol{h}_t \in \mathbb{R}^h$。依据链式法则，我们得到

$$\frac{\partial L}{\partial \boldsymbol{h}_t} = \sum_{i=t}^{T} \left(\boldsymbol{W}_{hh}^{\top}\right)^{T-i} \boldsymbol{W}_{qh}^{\top} \frac{\partial L}{\partial \boldsymbol{o}_{T+t-i}}. \tag{3.5}$$

由上式中的指数项可见，当时间步数 T 较大或者时间步 t 较小时，目标函数有关隐藏状态的梯度较容易出现衰减和爆炸。这也会影响其他包含 $\partial L/\partial \boldsymbol{h}_t$ 项的梯度。

3.2 LSTM

3.3 GRU

3.4 Transform

3.5 Bert

4.1 GNN

4.2 GCN