AI深度学习

损 失 函 数 和 反 向 传 播

1 损失函数

- 常见的损失函数
- 回归问题：MSE
- 分类问题：交叉熵

2 神经网络训练的公式推导

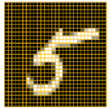
- 前向传播的公式推导
- 反向传播的公式推导
- 一些常见层的梯度推导
- 随机梯度下降
- 学习率衰减策略

1.损失函数

损失函数

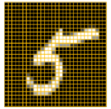
损失函数loss function
作用：衡量输出结果与真实标签的误差（损失）。
训练目标：最小化损失函数
在训练过程中，利用损失计算参数的梯度，更新参数。
同时损失函数值可以代表神经网络的拟合效果。

面试问题：什么是损失函数？为什么需要损失函数？



输入 → 网络参数1 → 输出结果y1 → L(y1,y')较大，预测效果差

真实标签y'



输入 → 网络参数2 → 输出结果y2 → L(y2,y')较小，预测效果好

损失函数

九曲阑干

常用的损失函数：

回归问题：MSE（均方误差mean squared error ）

分类问题：交叉熵（cross-entropy）

面试问题：一般情况下，分类问题和回归问题通常使用什么损失函数?

$$L(\hat{y}, y)$$

$\hat{y} = f(x;\theta)$ 预测值

y 真实值（现实）

损失函数

常用的损失函数：

回归问题：MSE（均方误差mean squared error ）

MSE适用于计算值的距离（欧式距离）

$$L(\hat{y}, y)$$

$\hat{y} = f(x;\theta)$ 预测值

y 真实值（现实）

$$L_{MSE} = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2$$

第 i 个样本的预测值

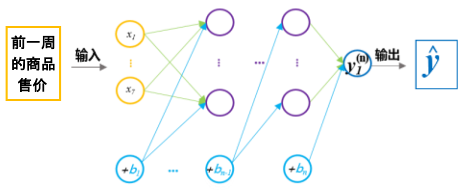
数据的循环（1~m）

当前batch中的样本数量

损失函数

常用的损失函数：

回归问题：MSE（均方误差mean squared error ）



处理商品售价样本（ 得到一个预测值 $\hat{y}$）

Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8
↓	↓	↓	↓	↓	↓	↓	↓
2.29	2.31	2.27	2.40	2.35	2.33	2.39	2.41

商品售价样本的label（ 真实值 y）

$$\hat{y} = 2.38$$

$$y = 2.41$$

MSE

$$E = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2$$

$$E = \frac{1}{1}\sum_{i=1}^{1}(\hat{y}_i - y_i)^2$$

$$= (\hat{y} - y)^2$$

$$= (2.38 - 2.41)^2$$

$$= 0.0009 \rightarrow \text{loss}$$

损失函数

常用的损失函数：

分类问题：交叉熵（cross-entropy）

交叉熵适用于计算分布的距离

对于离散分布的 p 和 q ,交叉熵误差为：

$$H(p,q) = -\sum_{j=1}^{T} p(x_j)\log q(x_j)$$

$p(x)$ 是样本 x 的真实值为第 j 类的概率

因为真实值即样本label已知，所以 $p(x)$ 为0或1；

$q(x)$ 是模型预测的样本 x 为第 j 类的概率

样本类别

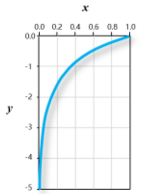
p 和 q 是两个单独的概率分布，$H(p)$ 是 p 的熵， $H(p,q)$ 是 p 相对于 q 的交叉熵（cross-entropy）。

$$L_{CE} = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{T} y_{ij}\log\hat{y}_{ij}$$

数据的类别

样本类别的循环（1~T）

样本数据的循环（1~m）



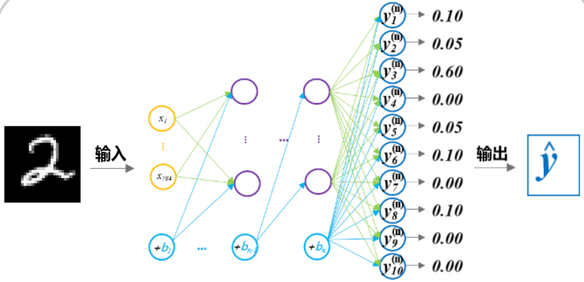
$y = \log x$

以 e 为底数的自然对数

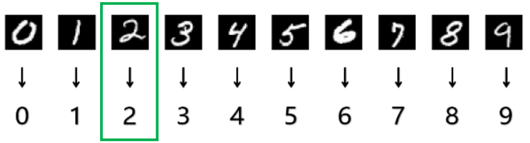
$\log_a x$：令 $a=e$，记作 $\ln x$

$x = e^y$

损失函数



处理单个手写数字样本（得到一个预测值$\hat{y}$）



手写数字样本的label（真实值y）

$$\hat{y} = \begin{bmatrix} 0.10 \\ 0.05 \\ \boxed{0.60} \\ 0.00 \\ 0.05 \\ 0.10 \\ 0.00 \\ 0.10 \\ 0.00 \\ 0.00 \end{bmatrix}$$

$$y = \begin{bmatrix} 0 \\ 0 \\ \boxed{1} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{array}{l} j=1 \\ j=2 \\ \boxed{j=3} \\ j=4 \\ j=5 \\ j=6 \\ j=7 \\ j=8 \\ j=9 \\ j=10 \end{array}$$

CEE

$$\begin{aligned} E &= -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{T} y_{ij}\log\hat{y}_{ij} \\ &= -\frac{1}{1}\sum_{i=1}^{1}\sum_{j=1}^{10} y_{ij}\log\hat{y}_{ij} \\ &= -y_3\log\hat{y}_3 \\ &= -1\log 0.60 \\ &= -(-0.5108256) \\ &\approx 0.51 \end{aligned}$$

$\rightarrow$ loss

损失函数

九曲阑干

CE和KL的关系

交叉熵（cross-entropy） $H(p,q) = -\sum_{i=1}^{n} p(x_i)\log(q(x_i))$

相对熵，也叫KL散度（Kullback-Leibler divergence） $D_{KL}(p||q) = -\sum_{i=1}^{n} p(x_i)\log(\frac{p(x_i)}{q(x_i)})$

信息熵 $H(p) = -\sum_{i=1}^{n} p(x_i)\log(p(x_i))$

关系：相对熵KL = 交叉熵CE - 信息熵 $D_{KL}(p||q) = H(p,q) - H(p)$

在机器学习中，通常使用KL衡量两个分布的距离，用于分类问题的损失

由于p是标签固定，信息熵是常数，在深度学习中使用CE代替KL作为损失

面试问题：交叉熵或KL散度适用于计算什么样的距离？

面试问题：交叉熵和KL的关系是什么？

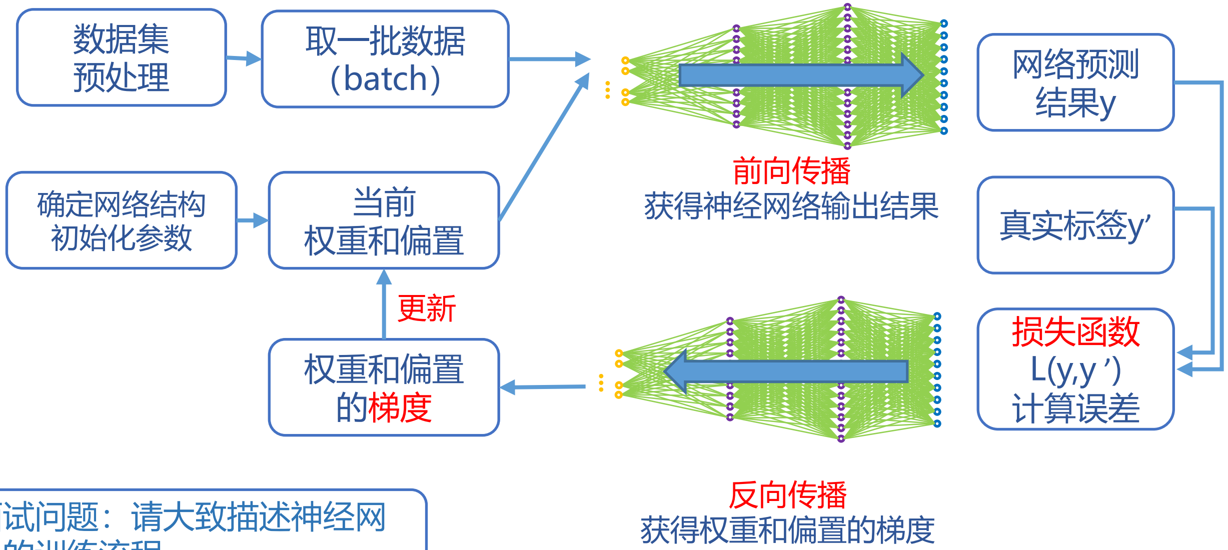
九曲阑干

2.神经网络训练的公式推导

前方高能！
有大量公式出没
建议自己动手推导

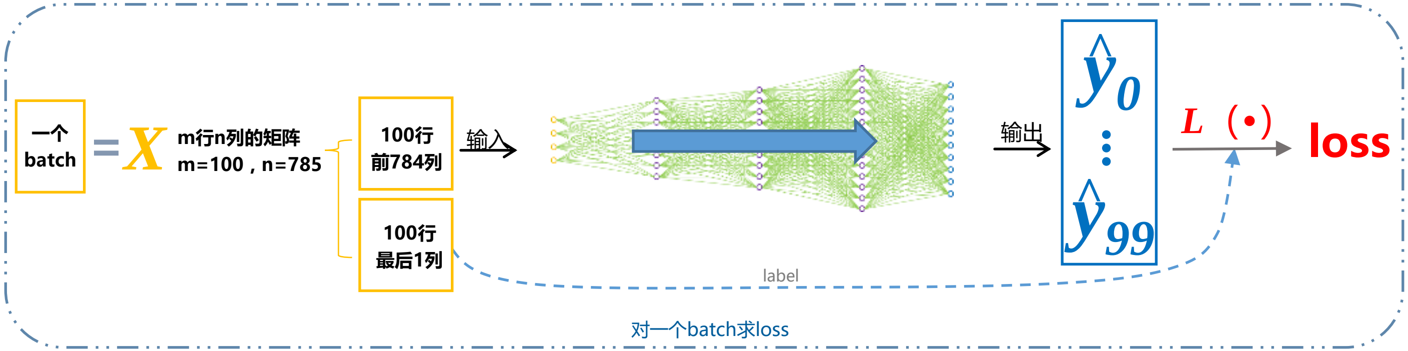
回顾：神经网络训练

①前向传播，计算结果，②计算损失，③反向传播，计算梯度，④更新参数



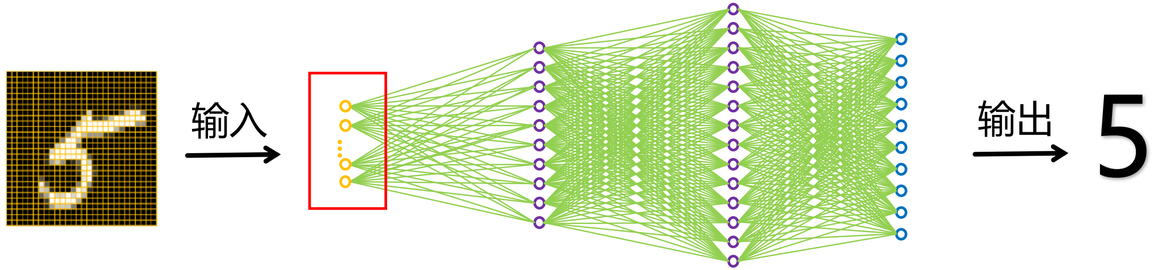
面试问题：请大致描述神经网络的训练流程

前向传播-求Loss



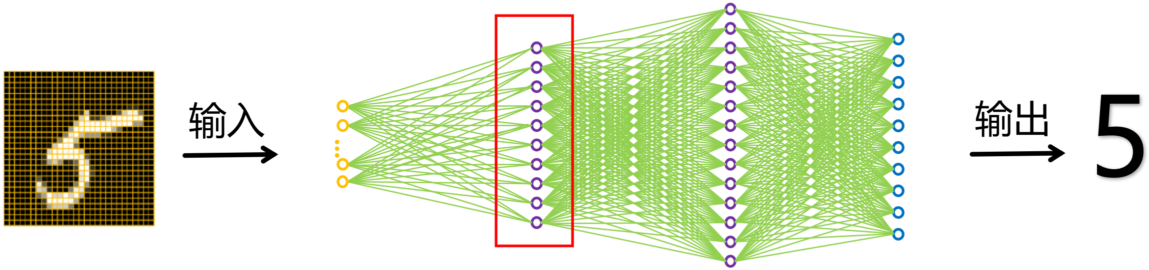
前向传播
方向：从前向后
计算：利用输入batch，计算输出结果和损失

前向传播的公式推导



输入图像x，标签y（为简化过程，省略ReLU）

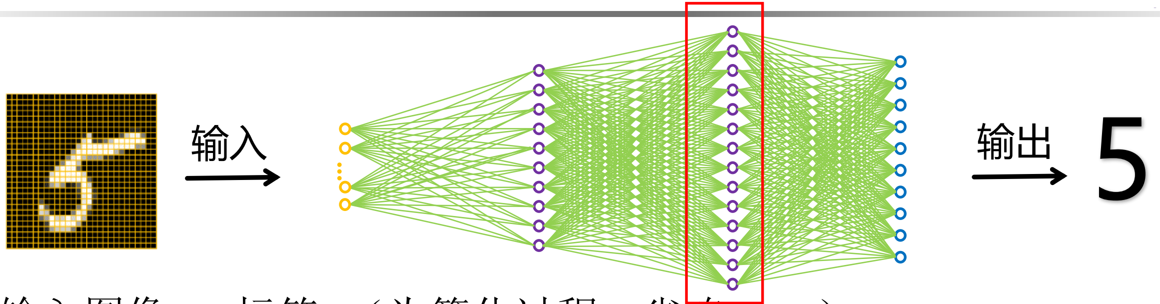
前向传播的公式推导



输入图像x，标签y（为简化过程，省略ReLU）

第一层输出$h^{(1)}$，参数为W_1, b_1，计算公式$h^{(1)} = W_1x + b_1$

前向传播的公式推导

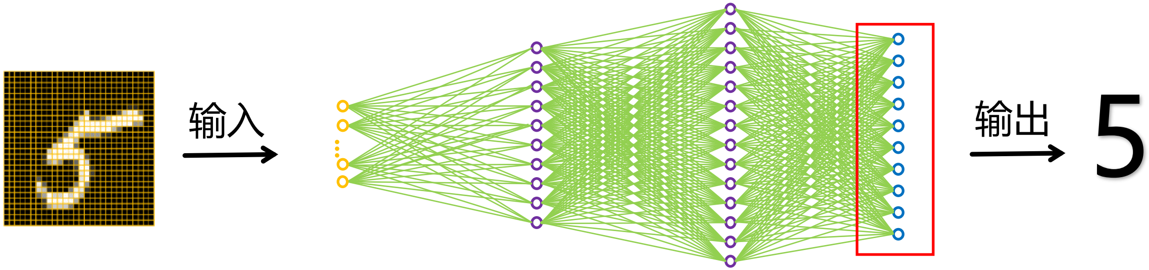


输入图像x，标签y（为简化过程，省略ReLU）

第一层输出$h^{(1)}$，参数为W_1, b_1，计算公式$h^{(1)} = W_1 x + b_1$

第二层输出$h^{(2)}$，参数为W_2, b_2，计算公式$h^{(2)} = W_2 h^{(1)} + b_2$

前向传播的公式推导



输入图像x，标签y（为简化过程，省略ReLU）

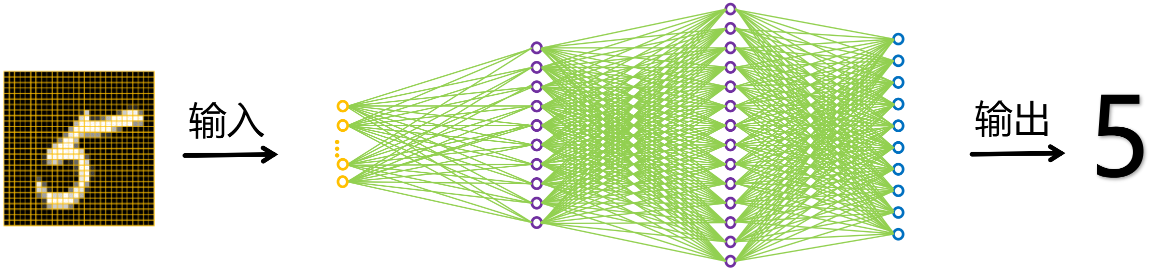
第一层输出$h^{(1)}$，参数为W_1, b_1，计算公式$h^{(1)} = W_1 x + b_1$

第二层输出$h^{(2)}$，参数为W_2, b_2，计算公式$h^{(2)} = W_2 h^{(1)} + b_2$

第三层输出$\hat{y}$，参数为W_3, b_3，经过softmax，计算公式

$$h^{(3)} = W_3 h^{(2)} + b_3，\ \hat{y}_i = \frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

前向传播的公式推导



输入图像x，标签y（为简化过程，省略ReLU）

第一层输出$h^{(1)}$，参数为W_1, b_1，计算公式$h^{(1)} = W_1 x + b_1$

第二层输出$h^{(2)}$，参数为W_2, b_2，计算公式$h^{(2)} = W_2 h^{(1)} + b_2$

第三层输出$\hat{y}$，参数为W_3, b_3，经过softmax，计算公式

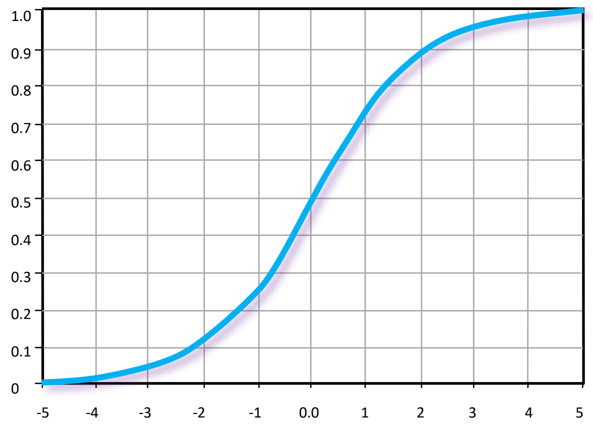
$$h^{(3)} = W_3 h^{(2)} + b_3，\hat{y}_i = \frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

交叉熵损失$L = -\sum_{i=1}^{T} y_i log\hat{y}_i$（T是类别数，T=10）

输出层-softmax函数

九曲阑干

用于**分类**的神经网络，**输出层**的每一个神经元都要经过softmax激活函数



面试问题：分类网络的输出层使用softmax的目的是什么？

函数表达式:

$$y_k = \frac{\exp(a_k)}{\sum_{i=1}^{n} \exp(a_i)}$$

exp(x)：是表示 e 的指数函数

n：输出层神经元的个数（即分类的所有类别数）

k：第k个神经元；

分子：输入信号 a_k 的指数函数；

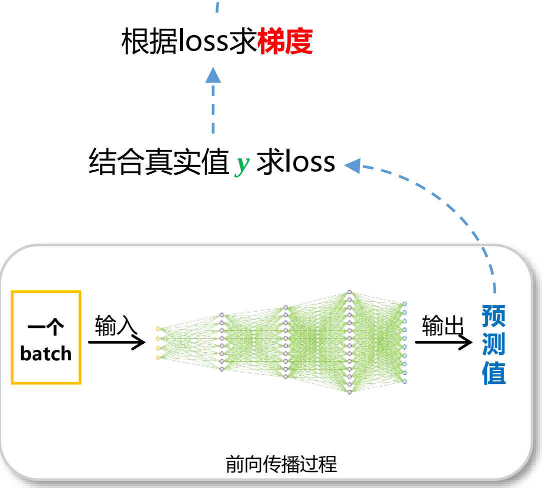
分母：所有输入信号的指数函数的和。

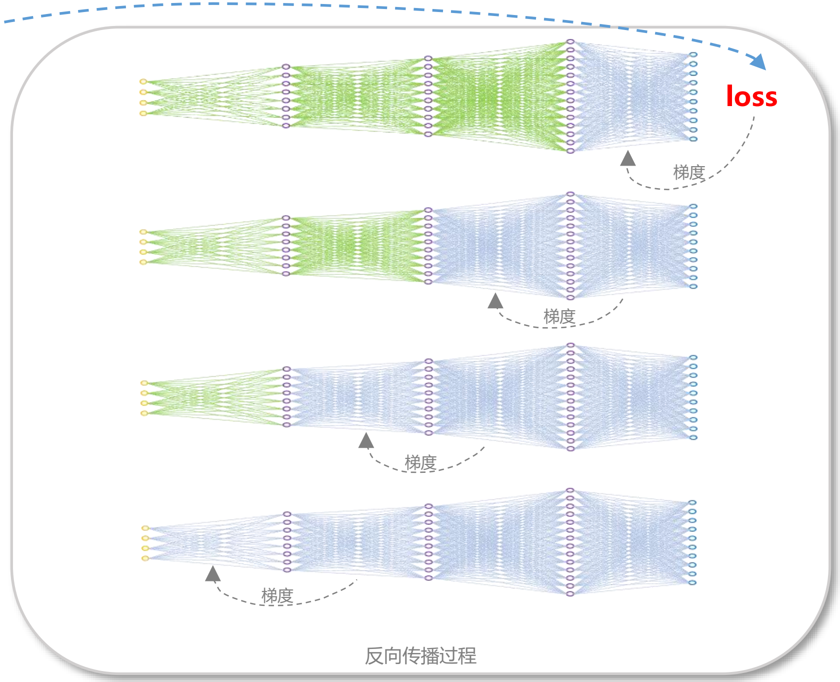
softmax将输出映射到0.0 到1.0之间的实数，输出和总是1。可以理解为“分类概率”

经过softmax计算后，概率值不会为负数。

反向传播

反向传播
方向：从后向前
计算：权重和偏置的梯度





反向传播：链式求导计算梯度

参数更新需要计算所有参数的梯度

如何计算神经网络所有参数的梯度?

使用链式求导法则

$$y = f(x), L = g(y)$$

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y} \cdot \frac{\partial y}{\partial x}$$

反向传播：链式求导计算梯度

$$\hat{y}_i = \frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

$$L = -\sum_{i=1}^{T} y_i log \hat{y}_i$$

求：$\frac{\partial L}{\partial h^{(3)}} = ?$

答：$\frac{\partial L}{\partial h^{(3)}} = \hat{y} - y$

反向传播：链式求导计算梯度

$$\hat{y}_i = \frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

$$L = -\sum_{i=1}^{T} y_i log\hat{y}_i$$

求：$\frac{\partial L}{\partial h^{(3)}} = ?$

答：$\frac{\partial L}{\partial h^{(3)}} = \hat{y} - y$

推导过程：

由于y是one-hot，

假设真实类别是k，那么有

$$y_k = 1, y_{i(i \neq k)} = 0$$

则$L = -\sum_{i=1}^{T} y_i log\hat{y}_i = -log\hat{y}_k$

可知$\frac{\partial L}{\partial \hat{y}_k} = -\frac{1}{\hat{y}_k}$

下面求$\frac{\partial L}{\partial h_i^{(3)}} = ?$，

根据链式法则

$$\frac{\partial L}{\partial h_i^{(3)}} = \frac{\partial L}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial h_i^{(3)}}$$

已知$\hat{y}_k = \frac{e^{h_k^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$

反向传播：链式求导计算梯度

九曲阑干

$$\hat{y}_i=\frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T}e^{h_j^{(3)}}}$$

$$L=-\sum_{i=1}^{T}y_i log\hat{y}_i$$

求：$\frac{\partial L}{\partial h^{(3)}}=?$

答：$\frac{\partial L}{\partial h^{(3)}}=\hat{y}-y$

推导过程：

由于y是one-hot，

假设真实类别是k，那么有

$$y_k=1, y_{i(i\neq k)}=0$$

则$L=-\sum_{i=1}^{T}y_i log\hat{y}_i=-log\hat{y}_k$

可知$\frac{\partial L}{\partial \hat{y}_k}=-\frac{1}{\hat{y}_k}$

下面求$\frac{\partial L}{\partial h_i^{(3)}}=?$

根据链式法则

$$\frac{\partial L}{\partial h_i^{(3)}}=\frac{\partial L}{\partial \hat{y}_k}\frac{\partial \hat{y}_k}{\partial h_i^{(3)}}$$

已知$\hat{y}_k=\frac{e^{h_k^{(3)}}}{\sum_{j=1}^{T}e^{h_j^{(3)}}}$

分两种情况

①当$i=k$时，

$\hat{y}_k$的分子分母都有$h_k^{(3)}$

$$\frac{\partial \hat{y}_k}{\partial h_k^{(3)}}=\frac{e^{h_k^{(3)}}\left(\sum_{j=1}^{T}e^{h_j^{(3)}}\right)-e^{h_k^{(3)}}e^{h_k^{(3)}}}{\left(\sum_{j=1}^{T}e^{h_j^{(3)}}\right)^2}$$

$$=\frac{e^{h_k^{(3)}}}{\sum_{j=1}^{T}e^{h_j^{(3)}}}\cdot\frac{\sum_{j=1}^{T}e^{h_j^{(3)}}-e^{h_k^{(3)}}}{\sum_{j=1}^{T}e^{h_j^{(3)}}}$$

$$=\hat{y}_k(1-\hat{y}_k)$$

$$\frac{\partial L}{\partial h_i^{(3)}}=\frac{\partial L}{\partial \hat{y}_k}\frac{\partial \hat{y}_k}{\partial h_i^{(3)}}=-\frac{1}{\hat{y}_k}\cdot\hat{y}_k(1-\hat{y}_k)$$

可得$\frac{\partial L}{\partial h_i^{(3)}}=\hat{y}_k-1$

$$\left[\frac{u(x)}{v(x)}\right]' = \frac{u'(x)v(x) - u(x)v'(x)}{v^2(x)}. \quad 【v\,(x) \neq 0】$$

$$\hat{y}_k = \frac{e^{h_k^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

$$\frac{\partial \hat{y}_k}{\partial h_k^{(3)}} = \frac{e^{h_k^{(3)}}\left(\sum_{j=1}^{T} e^{h_j^{(3)}}\right) - e^{h_k^{(3)}} e^{h_k^{(3)}}}{\left(\sum_{j=1}^{T} e^{h_j^{(3)}}\right)^2} = \frac{e^{h_k^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}} \cdot \frac{\sum_{j=1}^{T} e^{h_j^{(3)}} - e^{h_k^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

反向传播：链式求导计算梯度

九曲阑干

$$\hat{y}_i = \frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

$$L = -\sum_{i=1}^{T} y_i log\hat{y}_i$$

求：$\frac{\partial L}{\partial h^{(3)}} = ?$

答：$\frac{\partial L}{\partial h^{(3)}} = \hat{y} - y$

推导过程：
由于y是one-hot，
假设真实类别是k，那么有

$$y_k = 1, y_{i(i\neq k)} = 0$$

则$L = -\sum_{i=1}^{T} y_i log\hat{y}_i = -log\hat{y}_k$

可知$\frac{\partial L}{\partial \hat{y}_k} = -\frac{1}{\hat{y}_k}$

下面求$\frac{\partial L}{\partial h_i^{(3)}} = ?$

根据链式法则

$$\frac{\partial L}{\partial h_i^{(3)}} = \frac{\partial L}{\partial \hat{y}_k}\frac{\partial \hat{y}_k}{\partial h_i^{(3)}}$$

已知$\hat{y}_k = \frac{e^{h_k^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$

分两种情况
②当$i \neq k$时，
$\hat{y}_k$的分母有$h_i^{(3)}$，分子没有

$$\frac{\partial \hat{y}_k}{\partial h_i^{(3)}} = \frac{-e^{h_k^{(3)}} e^{h_i^{(3)}}}{(\sum_{j=1}^{T} e^{h_j^{(3)}})^2}$$

$$= \frac{e^{h_k^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}} \cdot \frac{-e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

$$= \hat{y}_k(-\hat{y}_i)$$

$$\frac{\partial L}{\partial h_i^{(3)}} = \frac{\partial L}{\partial \hat{y}_k}\frac{\partial \hat{y}_k}{\partial h_i^{(3)}} = -\frac{1}{\hat{y}_k} \cdot \hat{y}_k(-\hat{y}_i)$$

可得$\frac{\partial L}{\partial h_i^{(3)}} = \hat{y}_i$

反向传播：链式求导计算梯度

九曲阑干

$$\hat{y}_i = \frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

$$L = \sum_{i=1}^{T} y_i log\hat{y}_i$$

求：$\frac{\partial L}{\partial h^{(3)}} = ?$

答：$\frac{\partial L}{\partial h^{(3)}} = \hat{y} - y$

推导过程：

由于y是one-hot，

假设真实类别是k，那么有

$$y_k = 1, y_{i(i \neq k)} = 0$$

则$L = \sum_{i=1}^{T} y_i log\hat{y}_i = log\hat{y}_k$

可知$\frac{\partial L}{\partial \hat{y}_k} = \frac{1}{\hat{y}_k}$

下面求$\frac{\partial L}{\partial h_i^{(3)}} = ?$，

根据链式法则

$$\frac{\partial L}{\partial h_i^{(3)}} = \frac{\partial L}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial h_i^{(3)}}$$

已知$\hat{y}_k = \frac{e^{h_k^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$

总结两种情况

①当$i = k$时，$y_k = 1$

$$\frac{\partial L}{\partial h_i^{(3)}} = \hat{y}_k - 1 = \hat{y}_k - y_k$$

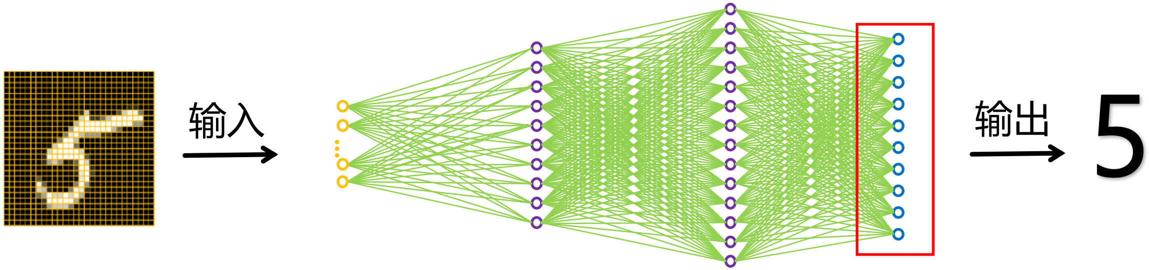
②当$i \neq k$时，$y_{i(i \neq k)} = 0$

$$\frac{\partial L}{\partial h_i^{(3)}} = \hat{y}_i = \hat{y}_i - y_i$$

合并可得

$$\frac{\partial L}{\partial h^{(3)}} = \hat{y} - y$$

反向传播的公式推导



输入图像x，标签y

第一层$h^{(1)} = W_1 x + b_1$

第二层$h^{(2)} = W_2 h^{(1)} + b_2$

第三层

$$h^{(3)} = W_3 h^{(2)} + b_3，\hat{y}_i = \frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

交叉熵损失$L = -\sum_{i=1}^{T} y_i log\hat{y}_i$

链式法则

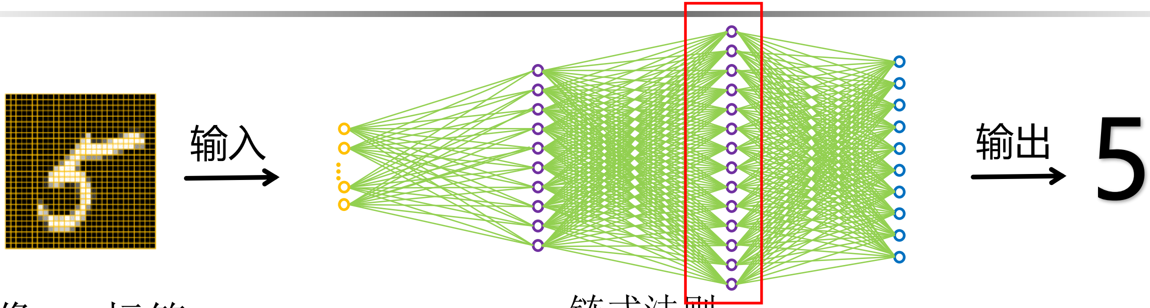
第三层$\frac{\partial L}{\partial h^{(3)}} = \hat{y} - y$

$$\frac{\partial L}{\partial W_3} = \frac{\partial L}{\partial h^{(3)}} \frac{\partial h^{(3)}}{\partial W_3} = \frac{\partial L}{\partial h^{(3)}} h^{(2)}$$

$$\frac{\partial L}{\partial b_3} = \frac{\partial L}{\partial h^{(3)}} \frac{\partial h^{(3)}}{\partial b_3} = \frac{\partial L}{\partial h^{(3)}}$$

$$\frac{\partial L}{\partial h^{(2)}} = \frac{\partial L}{\partial h^{(3)}} \frac{\partial h^{(3)}}{\partial h^{(2)}} = \frac{\partial L}{\partial h^{(3)}} W_3$$

反向传播的公式推导



输入图像x，标签y

第一层$h^{(1)} = W_1 x + b_1$

第二层$h^{(2)} = W_2 h^{(1)} + b_2$

第三层

$$h^{(3)} = W_3 h^{(2)} + b_3, \quad \hat{y}_i = \frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

交叉熵损失$L = -\sum_{i=1}^{T} y_i log \hat{y}_i$

链式法则

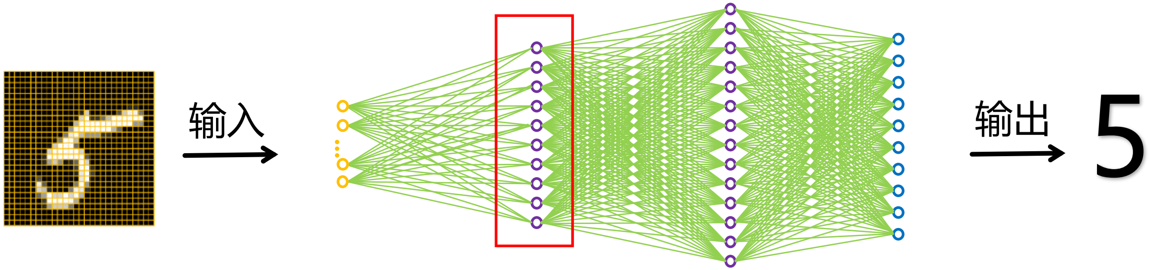
第二层$\frac{\partial L}{\partial h^{(2)}} = \frac{\partial L}{\partial h^{(3)}} W_3$

$$\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial h^{(2)}} \frac{\partial h^{(2)}}{\partial W_2} = \frac{\partial L}{\partial h^{(2)}} h^{(1)}$$

$$\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial h^{(2)}} \frac{\partial h^{(2)}}{\partial b_2} = \frac{\partial L}{\partial h^{(2)}}$$

$$\frac{\partial L}{\partial h^{(1)}} = \frac{\partial L}{\partial h^{(2)}} \frac{\partial h^{(2)}}{\partial h^{(1)}} = \frac{\partial L}{\partial h^{(2)}} W_2$$

反向传播的公式推导



输入图像x，标签y

第一层$h^{(1)} = W_1 x + b_1$

第二层$h^{(2)} = W_2 h^{(1)} + b_2$

第三层

$$h^{(3)} = W_3 h^{(2)} + b_3，\hat{y}_i = \frac{e^{h_i^{(3)}}}{\sum_{j=1}^{T} e^{h_j^{(3)}}}$$

交叉熵损失$L = -\sum_{i=1}^{T} y_i log\hat{y}_i$

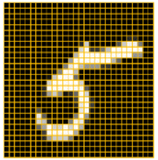
链式法则

第一层$\frac{\partial L}{\partial h^{(1)}} = \frac{\partial L}{\partial h^{(2)}} W_2$

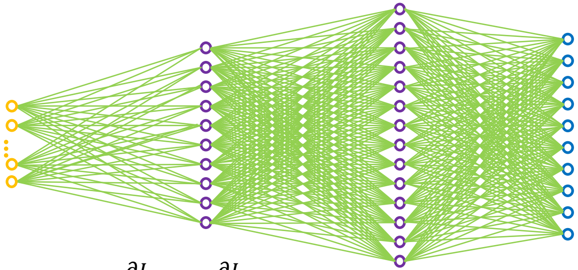
$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial h^{(1)}} \frac{\partial h^{(1)}}{\partial W_1} = \frac{\partial L}{\partial h^{(1)}} x$$

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial h^{(1)}} \frac{\partial h^{(1)}}{\partial b_1} = \frac{\partial L}{\partial h^{(1)}}$$

反向传播的公式推导



输入



输出 5

第三层 $\frac{\partial L}{\partial h^{(3)}} = \hat{y} - y$

$$\frac{\partial L}{\partial W_3} = \frac{\partial L}{\partial h^{(3)}} \frac{\partial h^{(3)}}{\partial W_3} = \frac{\partial L}{\partial h^{(3)}} h^{(2)}$$

$$\frac{\partial L}{\partial b_3} = \frac{\partial L}{\partial h^{(3)}} \frac{\partial h^{(3)}}{\partial b_3} = \frac{\partial L}{\partial h^{(3)}}$$

$$\frac{\partial L}{\partial h^{(2)}} = \frac{\partial L}{\partial h^{(3)}} \frac{\partial h^{(3)}}{\partial h^{(2)}} = \frac{\partial L}{\partial h^{(3)}} W_3$$

第二层 $\frac{\partial L}{\partial h^{(2)}} = \frac{\partial L}{\partial h^{(3)}} W_3$

$$\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial h^{(2)}} \frac{\partial h^{(2)}}{\partial W_2} = \frac{\partial L}{\partial h^{(2)}} h^{(1)}$$

$$\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial h^{(2)}} \frac{\partial h^{(2)}}{\partial b_2} = \frac{\partial L}{\partial h^{(2)}}$$

$$\frac{\partial L}{\partial h^{(1)}} = \frac{\partial L}{\partial h^{(2)}} \frac{\partial h^{(2)}}{\partial h^{(1)}} = \frac{\partial L}{\partial h^{(2)}} W_2$$

第一层 $\frac{\partial L}{\partial h^{(1)}} = \frac{\partial L}{\partial h^{(2)}} W_2$

$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial h^{(1)}} \frac{\partial h^{(1)}}{\partial W_1} = \frac{\partial L}{\partial h^{(1)}} x$$

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial h^{(1)}} \frac{\partial h^{(1)}}{\partial b_1} = \frac{\partial L}{\partial h^{(1)}}$$

总结：反向传播时每层需要计算：

①参数的梯度（用于更新参数）

②每层输出的梯度（用于反向传播时计算前一层）

面试问题：反向传播时需要计算哪些梯度?

一些常见层的梯度推导

九曲阑干

全连接层和卷积层（只包括线性运算）

输入x，参数包括权重W和偏置b，输出y

前向传播

$$y = Wx + b$$

反向传播（$\frac{\partial L}{\partial y}$会从后一层传过来）

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial W} = \frac{\partial L}{\partial y}x$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial b} = \frac{\partial L}{\partial y}$$

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial x} = \frac{\partial L}{\partial y}W$$

面试问题：全连接层的线性运算的前向传播和反向传播公式是什么？

一些常见层的梯度推导

九曲阑干

ReLU激活函数

输入x，输出y，无参数，逐元素操作

前向传播

$$y_i = \max(x_i, 0)$$

反向传播（$\frac{\partial L}{\partial y}$会从后一层传过来）

$$\frac{\partial L}{\partial x_i} = \frac{\partial L}{\partial y_i}\frac{\partial y_i}{\partial x_i} = \begin{cases} \frac{\partial L}{\partial y_i}, & x_i > 0 \\ 0, & x_i \le 0 \end{cases}$$

面试问题：ReLU的前向传播和反向传播公式是什么？

一些常见层的梯度推导

Softmax+交叉熵

输入h，softmax输出$\hat{y}$，真实标签y，无参数

前向传播

softmax：$\hat{y}_i = \frac{e^{h_i}}{\sum_{j=1}^{T} e^{h_j}}$

交叉熵损失 $L = -\sum_{i=1}^{T} y_i log\hat{y}_i$ （T是类别数）

反向传播

$$\frac{\partial L}{\partial h} = \hat{y} - y$$

面试问题：softmax和交叉熵损失的的前向传播和反向传播公式是什么？

一些常见层的梯度推导

九曲阑干

池化层

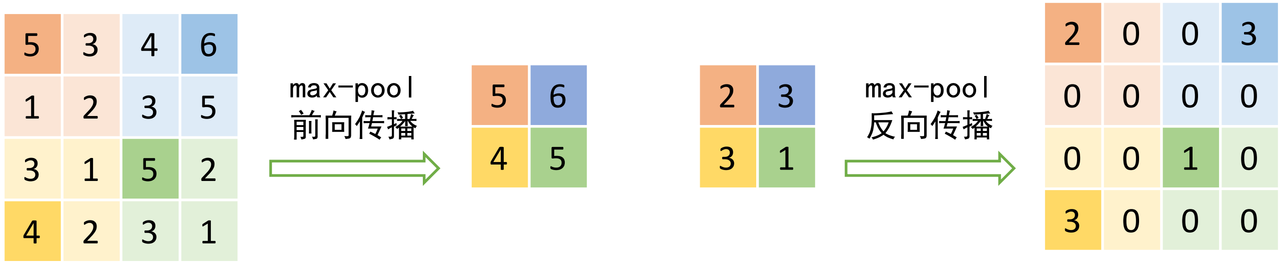
输入x，输出y，无参数，逐元素操作

面试问题：最大池化层的前向传播和反向传播是如何计算的？

最大池化

前向传播：对池化区域内取最大值

反向传播：将$\frac{\partial L}{\partial y}$赋值到最大值对应位置，其余梯度为0



一些常见层的梯度推导

九曲阑干

池化层

输入x，输出y，无参数，逐元素操作

> 面试问题：平均池化层的前向传播和反向传播是如何计算的？

平均池化

前向传播：对池化区域内取平均值

反向传播：将$\frac{\partial L}{\partial y}$平均赋值到池化区域

5	3	4	6
2	2	4	2
1	2	5	2
4	1	4	1

ave-pool
前向传播 →

3	4
2	3

4	8
12	4

ave-pool
反向传播 →

1	1	2	2
1	1	2	2
3	3	1	1
3	3	1	1

随机梯度下降

训练目标：最小化损失函数

随机梯度下降基本思路：

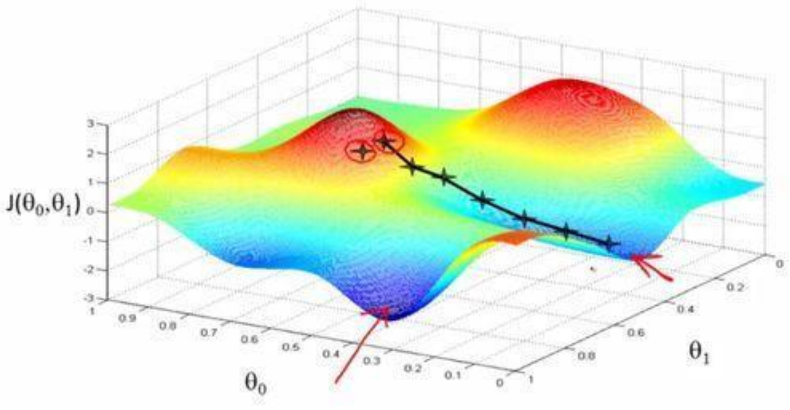
✓ 根据损失，利用链式求导法则计算梯度，

✓ 利用梯度更新参数，

✓ 参数更新后的损失会减小

✓ 不断向着损失更小的方向优化

梯度的反方向是损失函数下降最快的方向



随机梯度下降

问题：为什么不使用求解析解的方式，而要使用随机梯度下降？

一般的求解析解方式（如最小二乘法）：

①准备数据（温度和冰激凌的价格）

②列出自变量和因变量关系（线性模型）

②列出损失函数（如MSE）

③计算参数的偏导（梯度）

④计算梯度为0时的参数结果

	销量
25°	110
27°	115
31°	155
33°	160
35°	180

i	x	y
1	25	110
2	27	115
3	31	155
4	33	160
5	35	180

随机梯度下降

问题：为什么不使用求解析解的方式，而要使用随机梯度下降?

一般的求解析解方式（如最小二乘法）：
①准备数据（温度和冰激凌的价格）
②列出自变量和因变量关系（线性模型）
②列出损失函数（如MSE）
③计算参数的偏导（梯度）
④计算梯度为0时的参数结果

$$f(x) = ax + b$$

$$\epsilon = \sum(f(x_i) - y_i)^2 = \sum(ax_i + b - y_i)^2$$

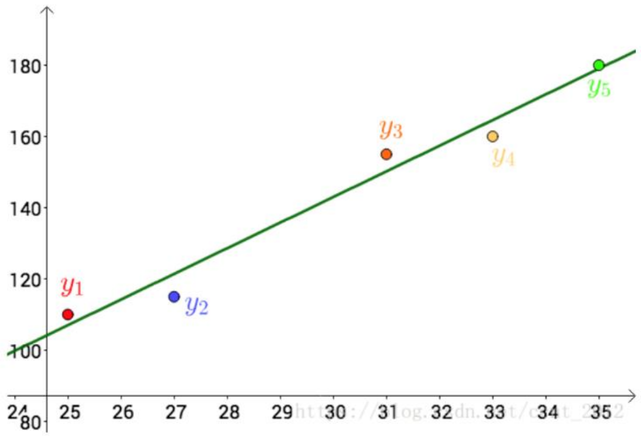
$$\begin{cases} \frac{\partial}{\partial a}\epsilon = 2\sum(ax_i + b - y_i)x_i = 0 \\ \frac{\partial}{\partial b}\epsilon = 2\sum(ax_i + b - y_i) = 0 \end{cases}$$

随机梯度下降

问题：为什么不使用求解析解的方式，而要使用随机梯度下降？

一般的求解析解方式（如最小二乘法）：
①准备数据（温度和冰激凌的价格）
②列出自变量和因变量关系（线性模型）
②列出损失函数（如MSE）
③计算参数的偏导（梯度）
④计算梯度为0时的参数结果

$$\begin{cases} a \approx 7.2 \\ b \approx -73 \end{cases}$$

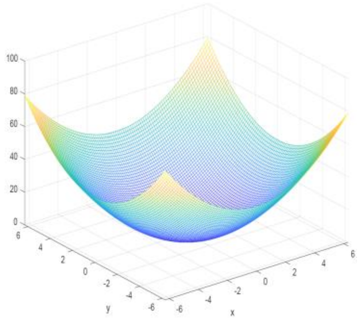


随机梯度下降

问题：为什么不使用求解析解的方式，而要使用随机梯度下降？

二次函数：只有一个极值点（梯度为0的点）

$$\epsilon = \sum (f(x_i) - y_i)^2 = \sum (ax_i + b - y_i)^2$$



随机梯度下降

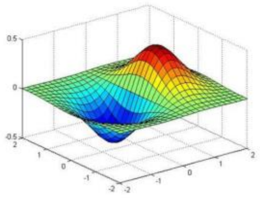
问题：为什么不使用求解析解的方式，而要使用随机梯度下降？

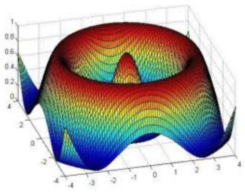
深度神经网络：

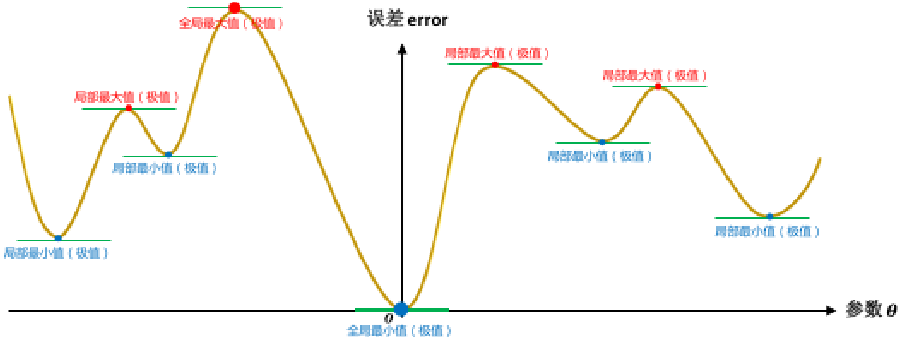
①深度神经网络的不是凸优化问题，优化曲面中有很多局部极小值，直接求梯度为0的点不一定是最优的

②求解析解计算复杂度高（可能需要算大矩阵的逆）

③使用随机梯度下降可以跳出差的局部极小值点，跳到较好的局部极小值点







随机梯度下降

问题：为什么使用随机梯度下降，即每次要取一个batch？

梯度下降：使用全部数据集计算梯度
随机梯度下降：
（注意：最原始的随机梯度下降指仅使用一个样本计算梯度）
目前深度学习中的随机梯度下降指mini-batch梯度下降，即使用一个batch的样本计算梯度
1、使用所有数据计算梯度，计算量太大，为了减少计算量使用一个batch
2、使用batch计算梯度是对整个数据集计算梯度的无偏估计，梯度的误差不大，可以保证收敛

面试问题：为什么训练神经网络使用随机梯度下降，不用求解析解的方式直接计算？

面试问题：为什么随机梯度下降中计算梯度只使用一个batch？这样为什么可以保证收敛？

回顾：参数更新

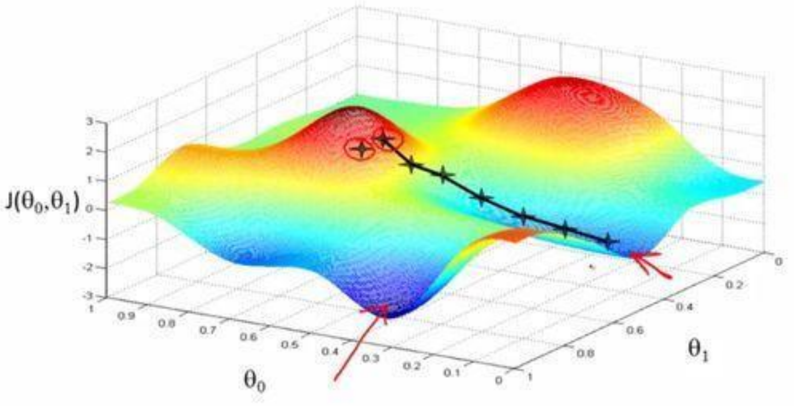
学习率learning rate：参数更新的步长，用η表示

梯度的反方向是损失函数下降最快的方向

参数更新公式:

$$W' \leftarrow W - \eta \frac{\partial L}{\partial W}$$

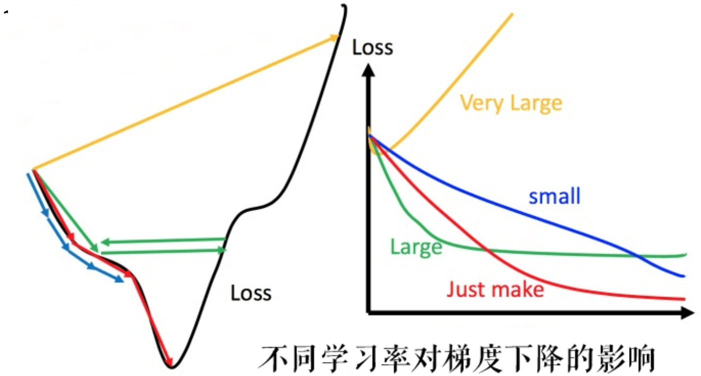
$$b' \leftarrow b - \eta \frac{\partial L}{\partial b}$$



学习率对训练的影响

九曲阑干

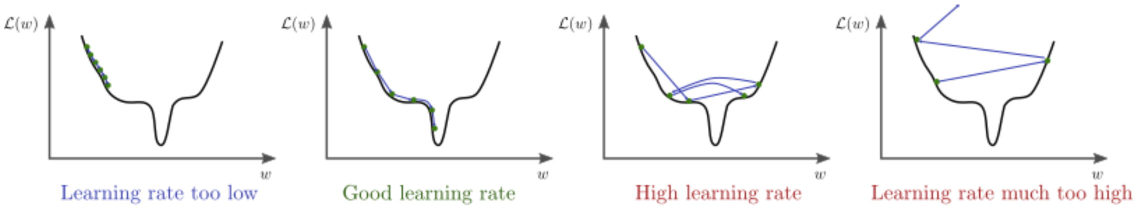
情况一：学习率合适，收敛速度快，效果好
情况二：学习率过小，收敛速度慢
情况三：学习率较大，开始收敛速度快，到一定程度loss不下降（在极小值附近摆动），效果差
情况四：学习率过大，无法收敛



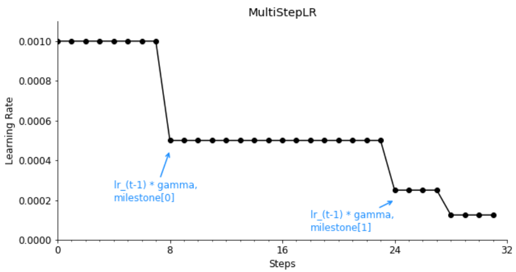
不同学习率对梯度下降的影响

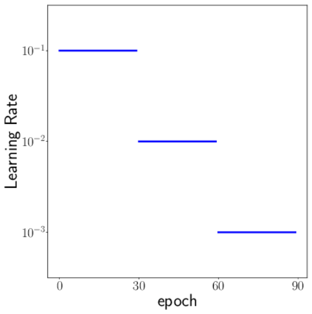
lumingdong.cn

面试问题：介绍学习率过大过小情况下对训练速度和结果的影响

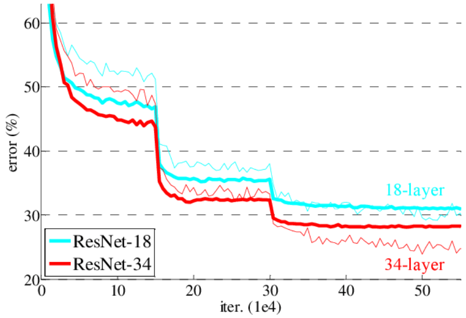


学习率衰减策略





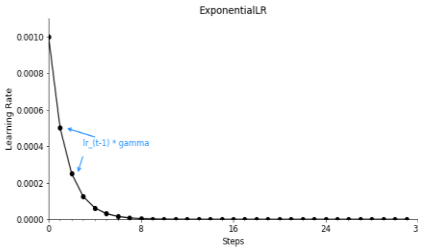
(a) Learning rate decay strategy



(b) Figure taken from He et al. (2016)

最常用：分段常数，设定在T1, T2, T3时学习率调整为β1，β2，β3，
逐渐变小，学习率变小时loss会阶梯状下降

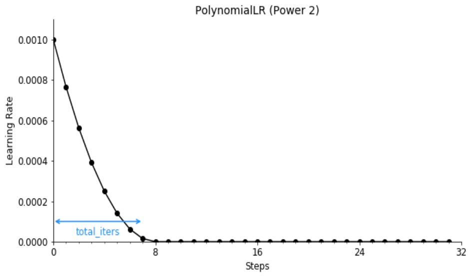
学习率衰减策略



指数衰减策略

exponential decay

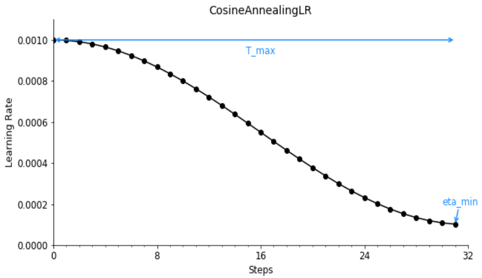
$$\eta_t = \eta_0 \beta^t$$



多项式衰减策略

polynomial decay

$$\eta_t = \eta_0 (\beta t + 1)^{-\alpha}$$



余弦退火衰减策略

CosineAnnealing

学习率按余弦的四分之一曲线衰减

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})\left(1 + \cos\left(\frac{T_{cur}}{T_{max}}\pi\right)\right)$$