全连接神经网络（1）

叶山 中国地质大学（北京）

yes@cugb.edu.cn

本教案部分参考以下课程：斯坦福大学 CS 231、密歇根大学 EECS 442 & 498、威斯康辛大学麦迪逊分校 CS 766

总体结构

全连接神经网络总体结构

线性分类器

$$f = Wx + b$$

- x：图像的向量表示
- f：分数向量
- W：权值矩阵
- b：偏置向量

全连接神经网络（2层）

$$f = W_2\,\varphi(W_1 x + b_1) + b_2$$

- x：图像的向量表示
- f：分数向量
- W_1、W_2：第一层、第二层的权值矩阵
- b_1、b_2：第一层、第二层的偏置向量
- φ：非线性变换（激活函数），例如 $\varphi(W_1 x + b_1) = max(0, W_1 x + b_1)$

全连接神经网络总体结构

两层全连接神经网络：

$$f = W_2\,\varphi(W_1 x + b_1) + b_2$$

第一层
第二层

三层全连接神经网络：

$$f = W_3\varphi_2(W_2\,\varphi_1(W_1 x + b_1) + b_2) + b_3$$

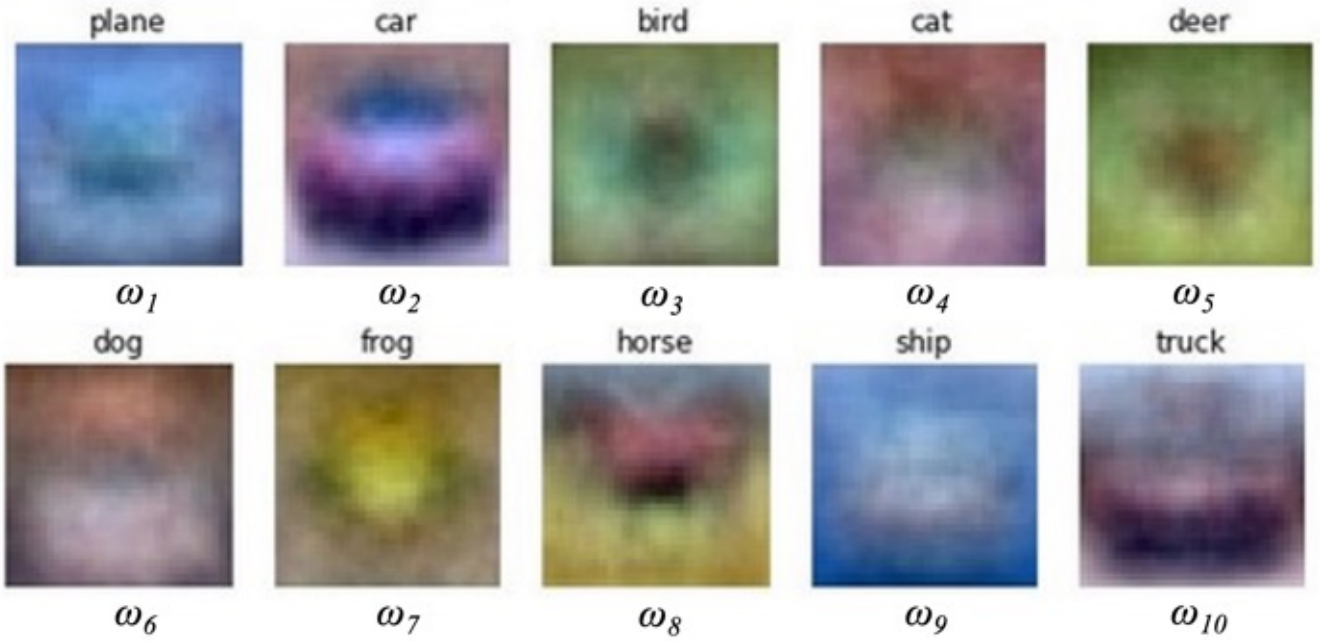
第一层
第二层
第三层

非线性操作（激活函数）
包括ReLU、Sigmoid、Tanh等

全连接神经网络的输入x到输出f的映射，需要通过级联多次变换实现。

权值模板

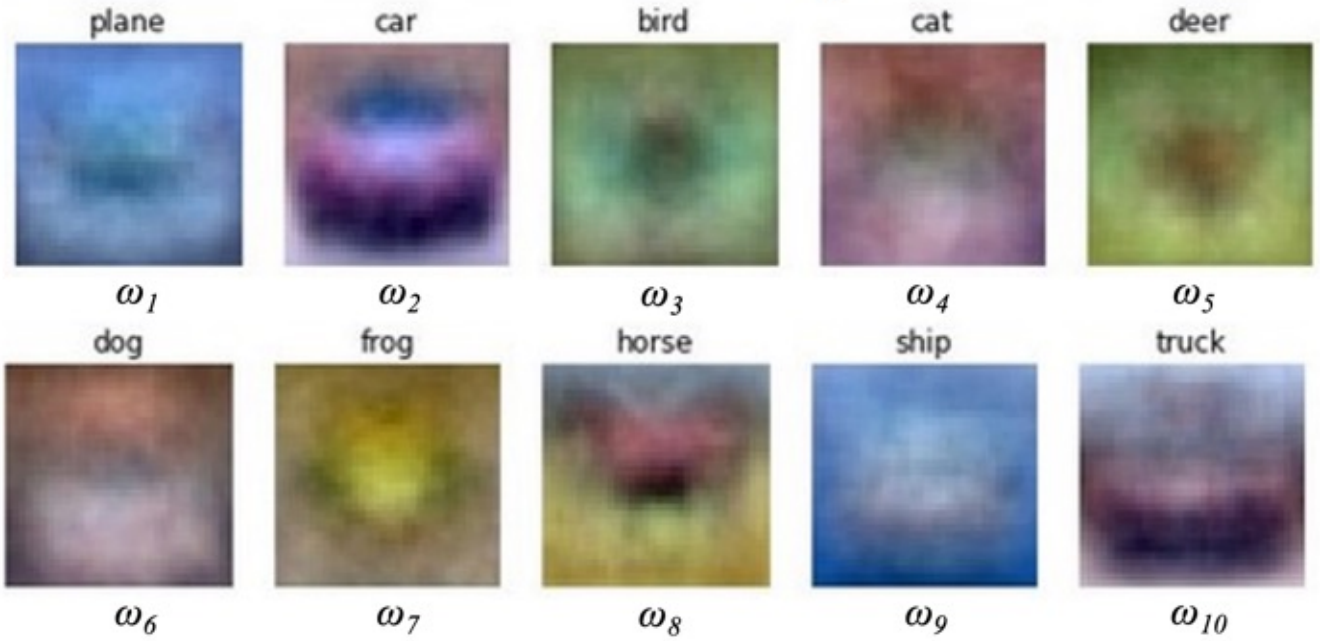
线性分类器：$f = Wx + b$



W可以看做模板，模板的个数由分类任务的类别个数决定。

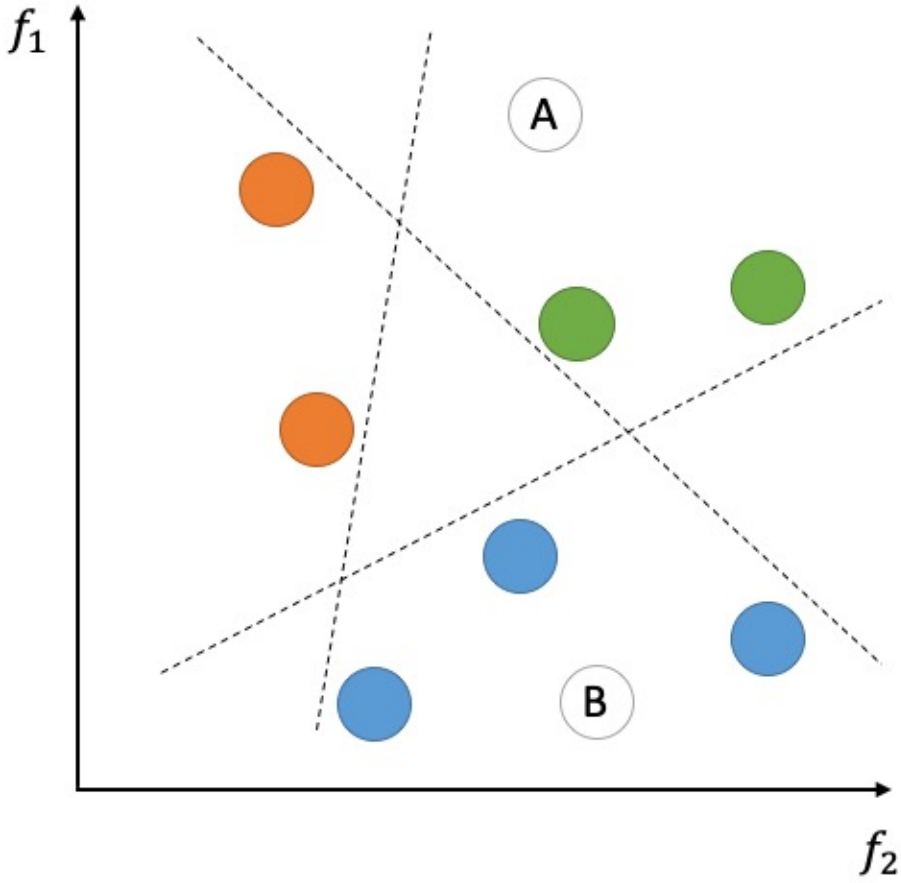
权值模板

两层全连接网络：$f = W_2\,\varphi(W_1 x + b_1) + b_2$



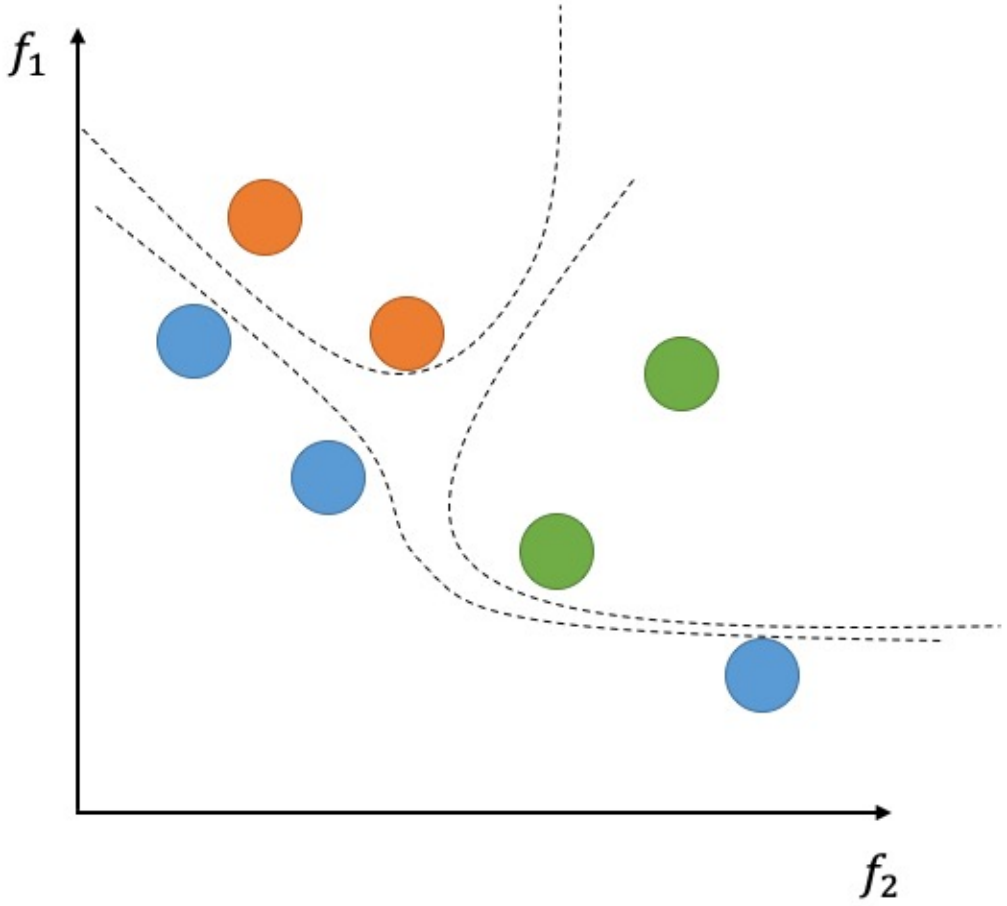
W_1也可以看做模板，模板的个数由人为指定，可以记录更加丰富的信息。行数自定，列数为输入图像x的像素个数。W_2融合多个模板的匹配结果，从而获得分数向量f。

线性可分



至少存在一个线性分界面，能把不同类别的样本完美地分开。

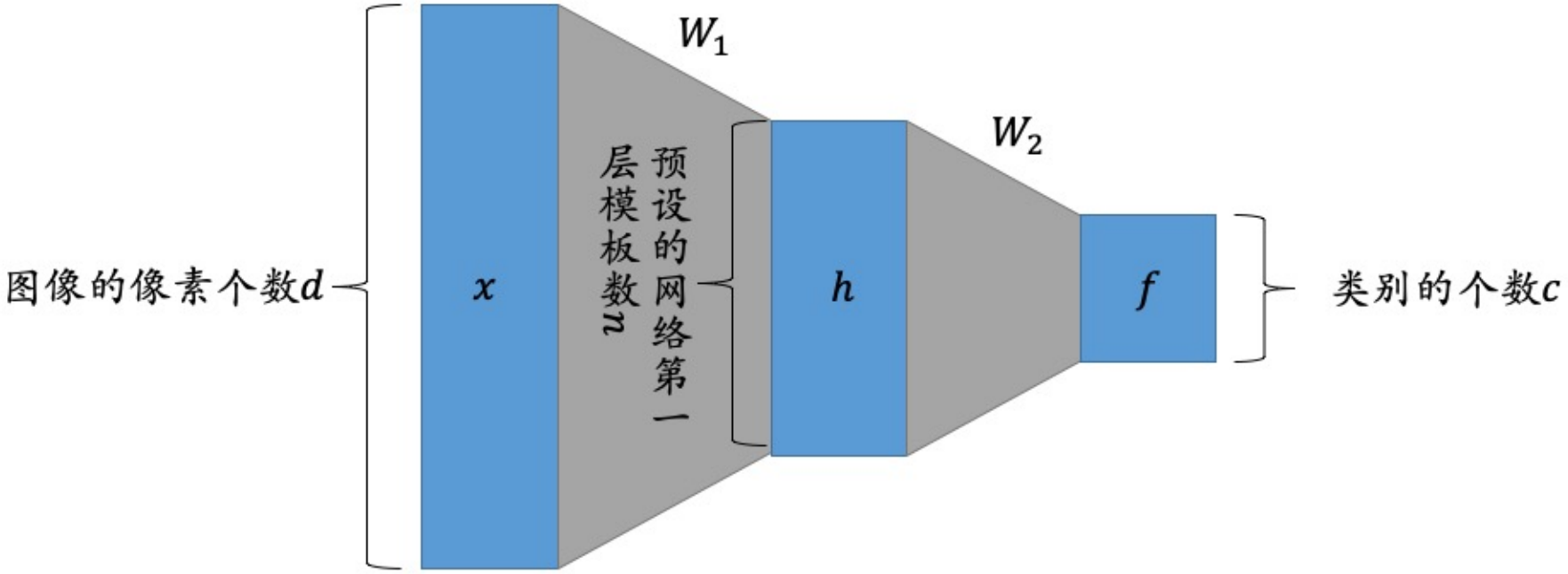
线性不可分



无法用线性分界面，把不同的类别完美地分开，需借助非线性操作来分类。

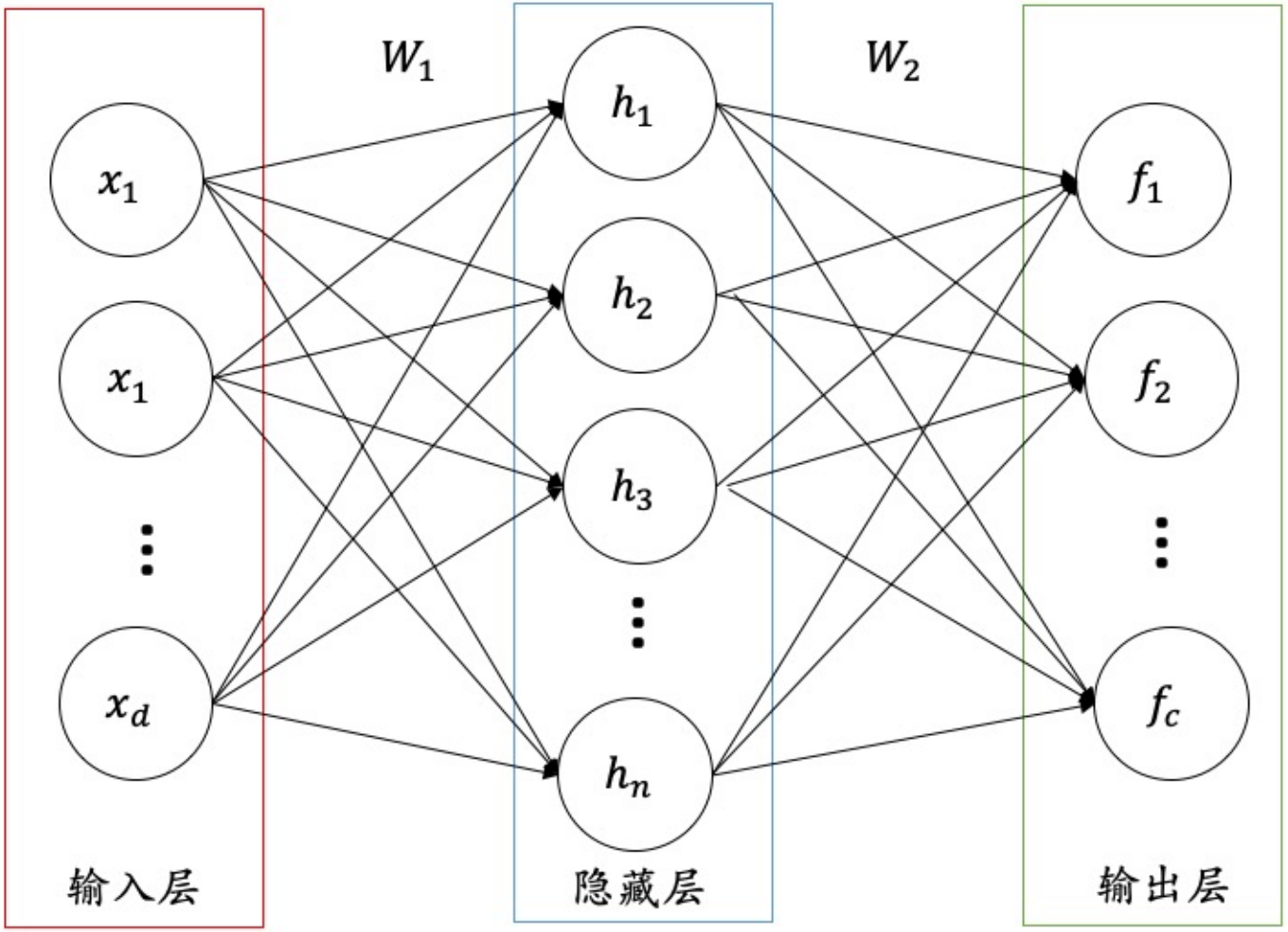
全连接神经网络的结构表示

两层全连接网络：$f = W_2\,\varphi(W_1 x + b_1) + b_2$

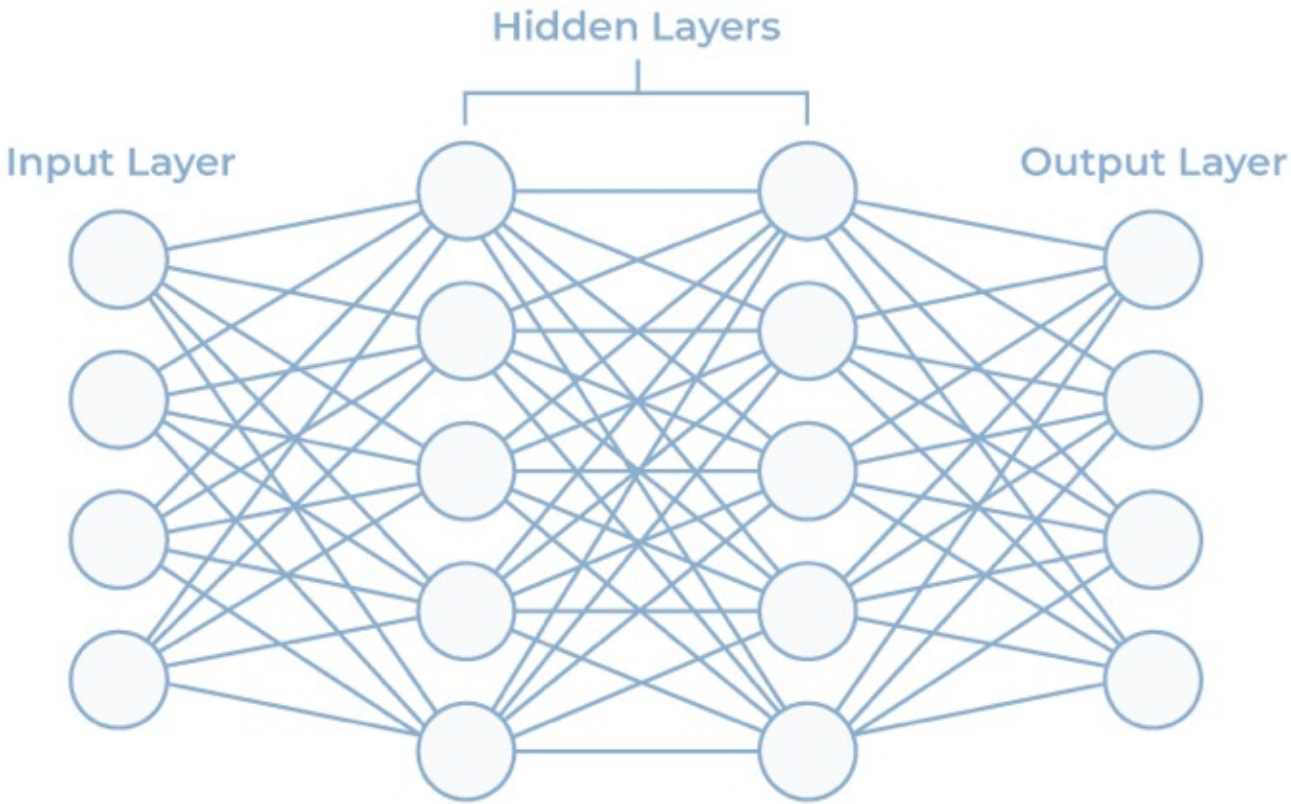


全连接神经网络的结构表示

两层全连接网络：$f = W_2\,\varphi(W_1 x + b_1) + b_2$



全连接神经网络的结构表示



3层全连接神经网络（描述层数时，通常不包括输入层）

激活函数

非线性操作

三层全链接网络:

$$f = W_3\varphi_2(W_2\,\varphi_1(W_1x + b_1) + b_2) + b_3$$

φ_1和φ_2为激活函数（activation function），目的是引入非线性操作，例如:

$$f = W_3 max(0, W_2\, max(0, W_1x + b_1) + b_2) + b_3$$

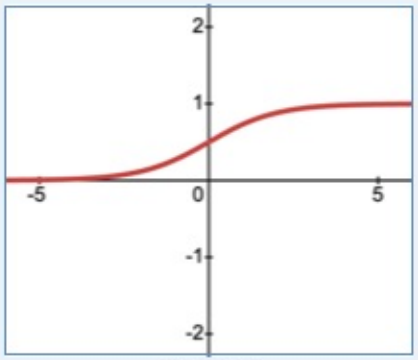
如果去掉激活函数，则变成:

$$f = W_3(W_2\ (W_1x + b_1) + b_2) + b_3$$

$$= W_1W_2W_3x + W_2W_3b_1 + W_3b_2 + b_3 = W'x + b'$$

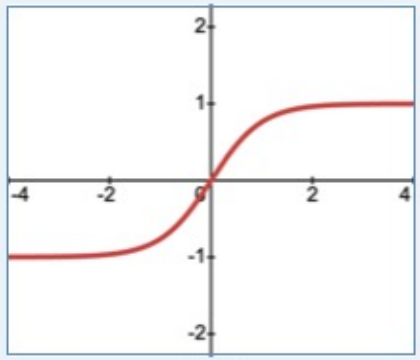
线性分类器

激活函数



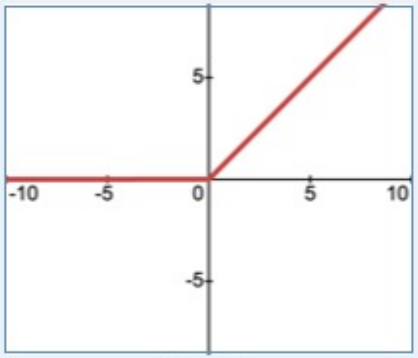
Sigmoid

$$y = \frac{1}{1 + e^{-x}}$$



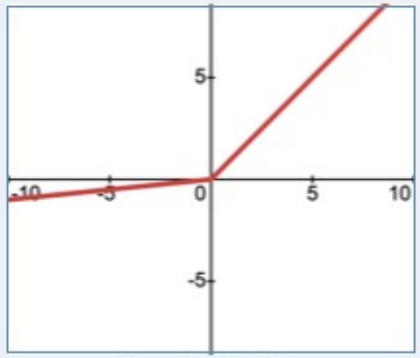
Tanh

$$y = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$



ReLU

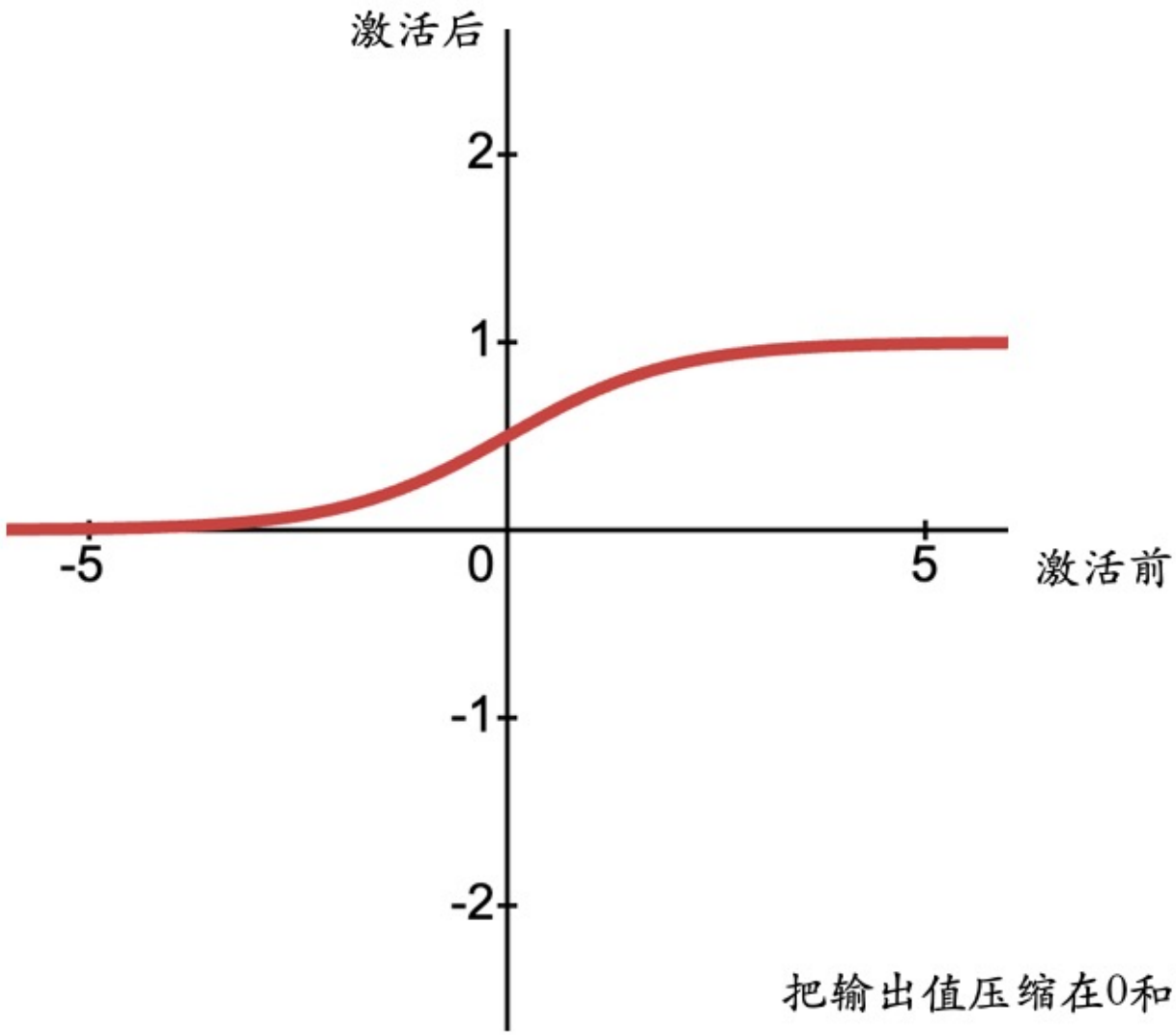
$$y = \max(0, x)$$



Leaky ReLU

$$y = \max(kx, x) \quad 0 < k < 1$$

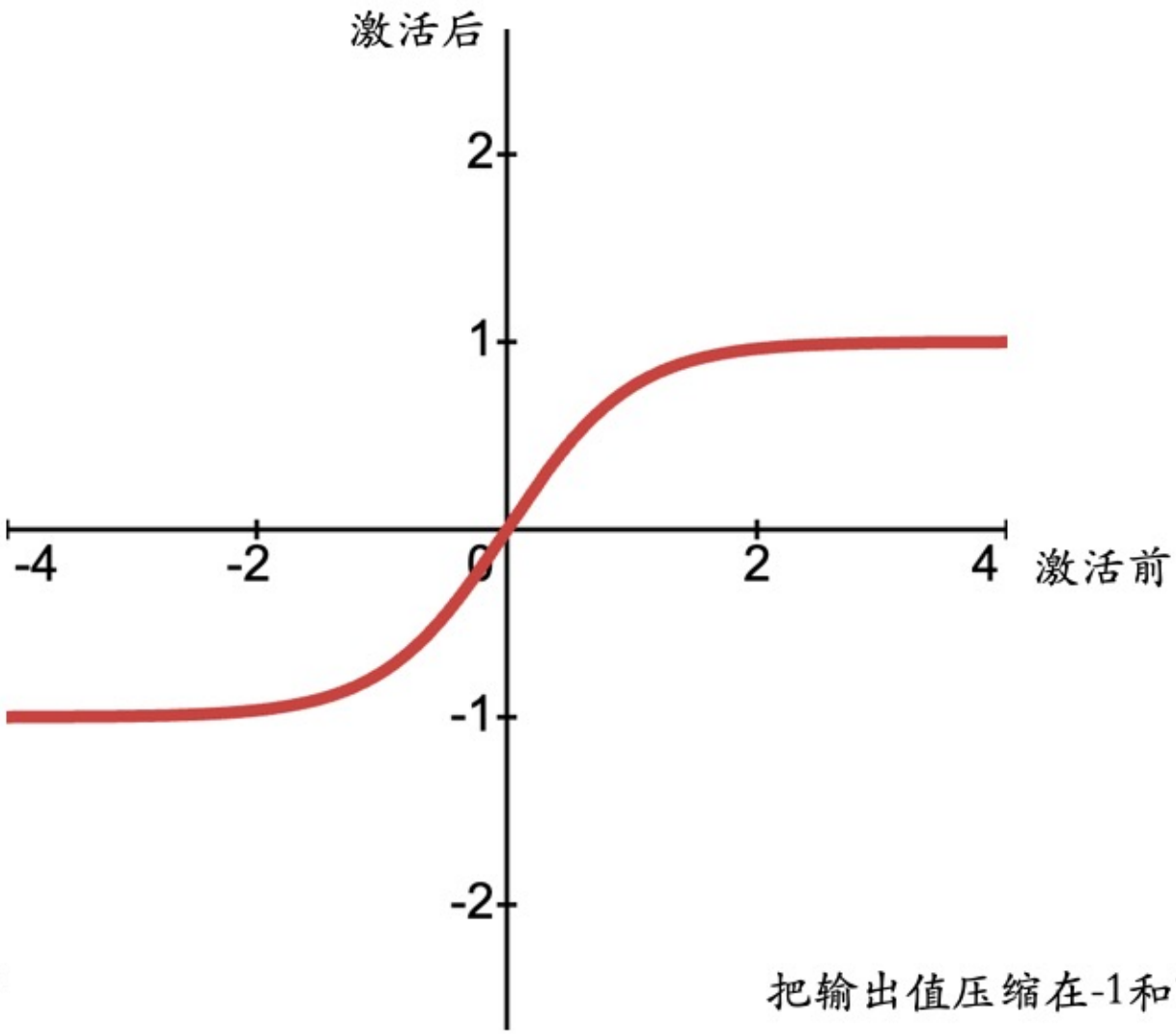
Sigmoid S状函数



$$y = \frac{1}{1 + e^{-x}}$$

把输出值压缩在0和1之间

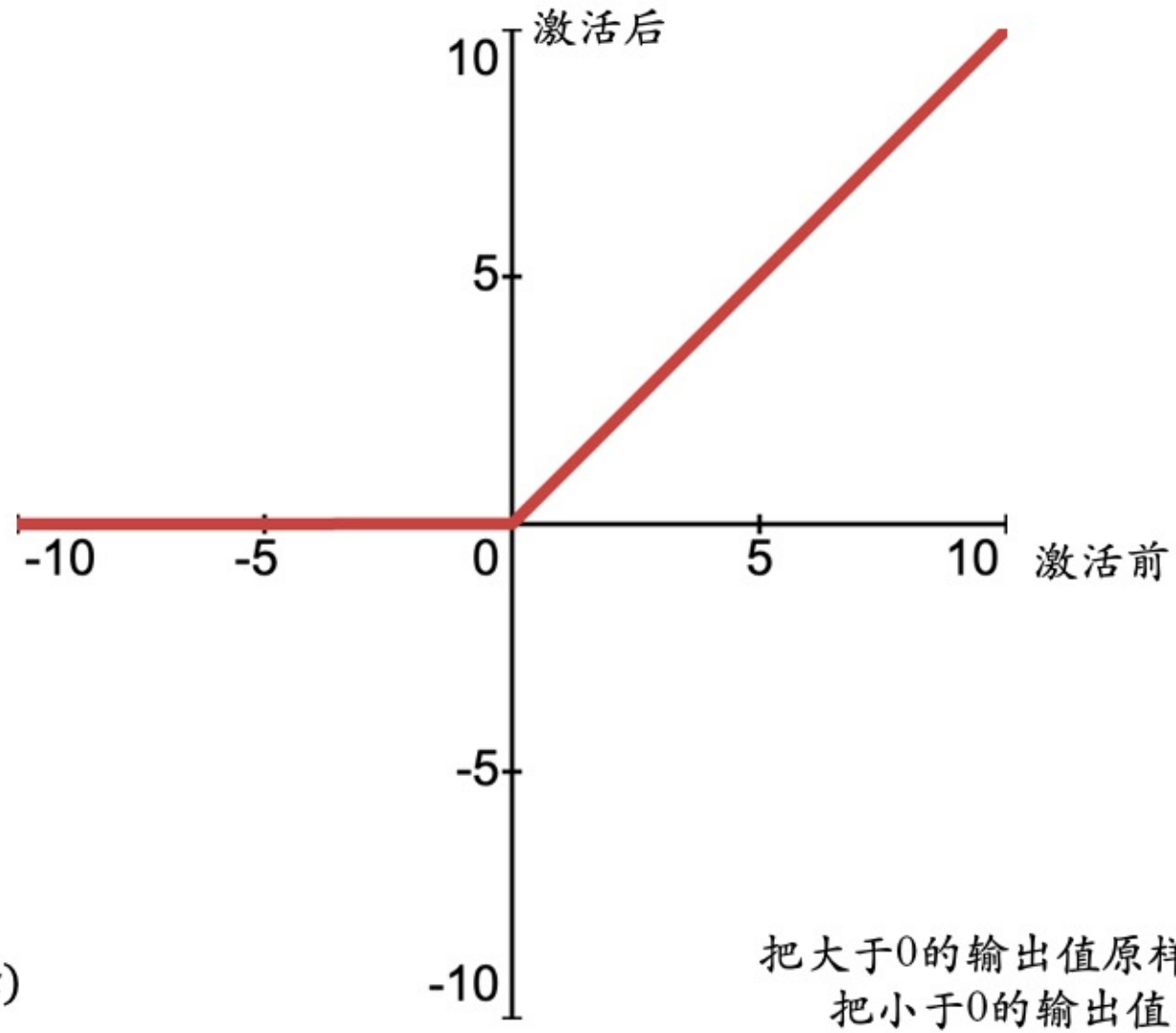
Tanh 双曲正切函数



$$y = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

把输出值压缩在-1和1之间

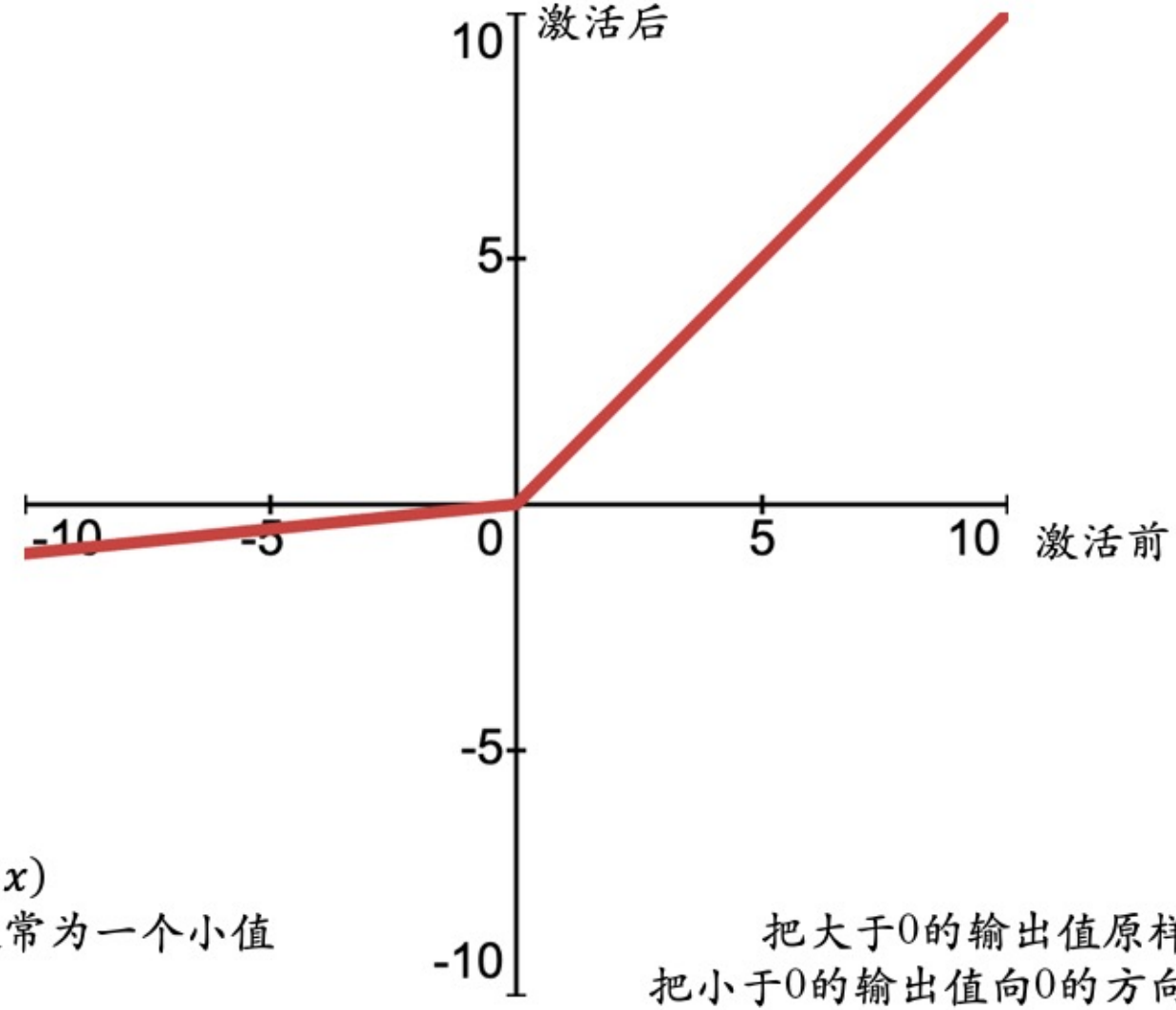
ReLU 线性整流函数



$y = \max(0, x)$

把大于0的输出值原样保留
把小于0的输出值变为0

Leaky ReLU 泄露线性整流函数



$y = \max(kx, x)$

$0 < k < 1$ 通常为一个小值

比如0.1

把大于0的输出值原样保留

把小于0的输出值向0的方向压缩

Softmax 归一化指数函数（输出层）

长度为类别数

f

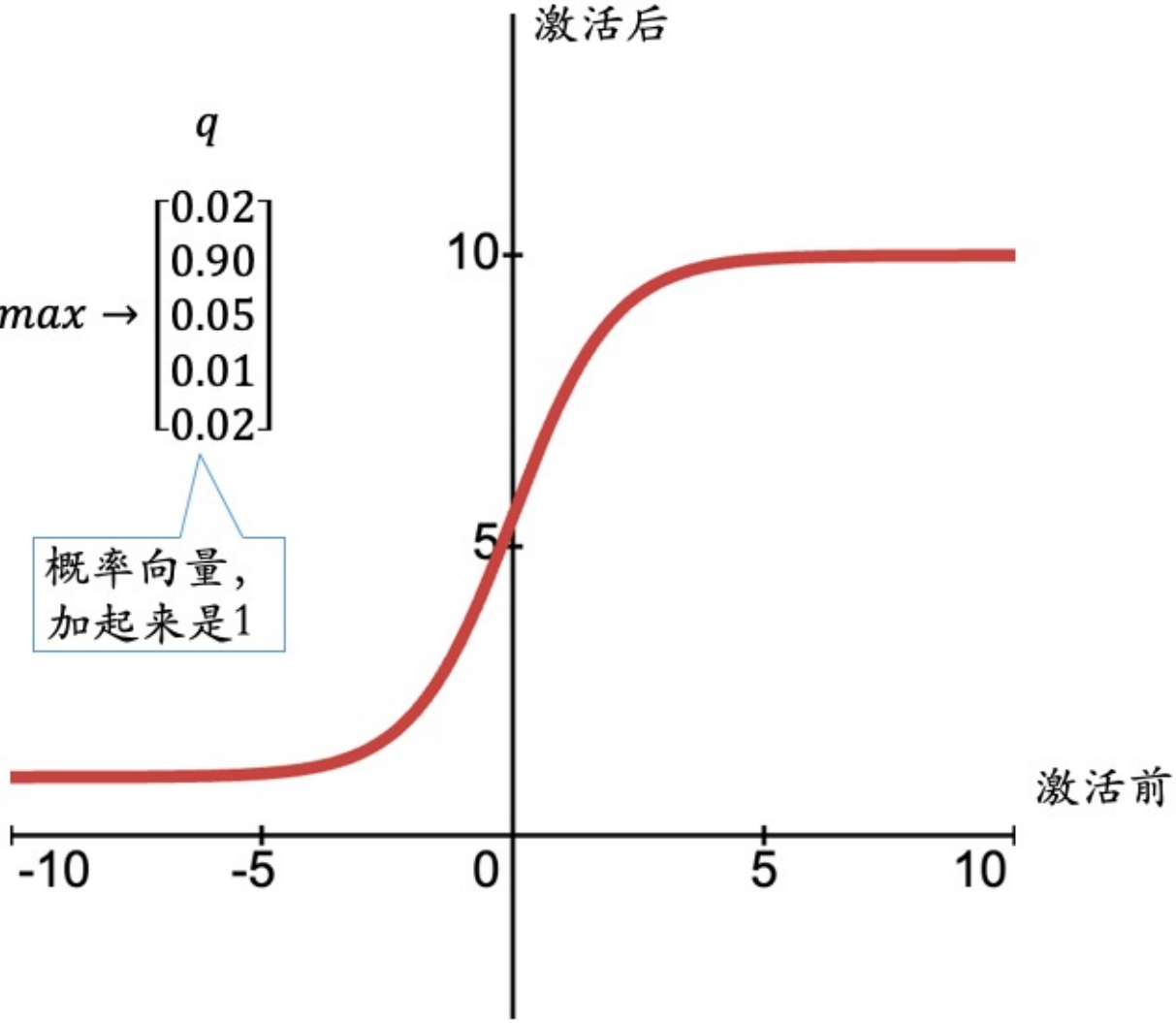
$$\begin{bmatrix}1.3\\5.1\\2.2\\0.7\\1.1\end{bmatrix} \rightarrow softmax \rightarrow \begin{bmatrix}0.02\\0.90\\0.05\\0.01\\0.02\end{bmatrix}$$

q

网络的原始输出向量

概率向量，加起来是1

$$y = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

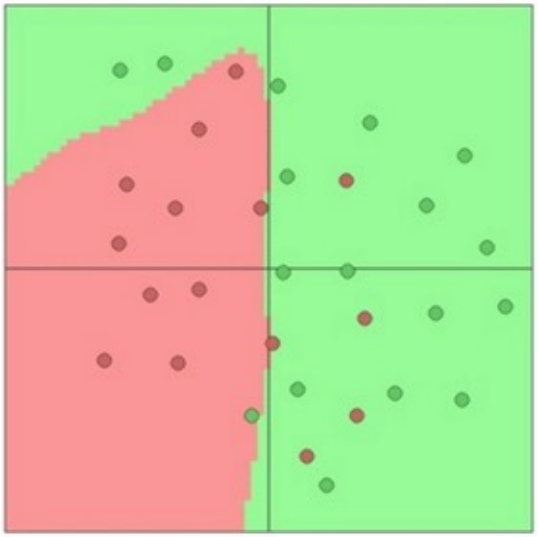


网络的设置及损失计算

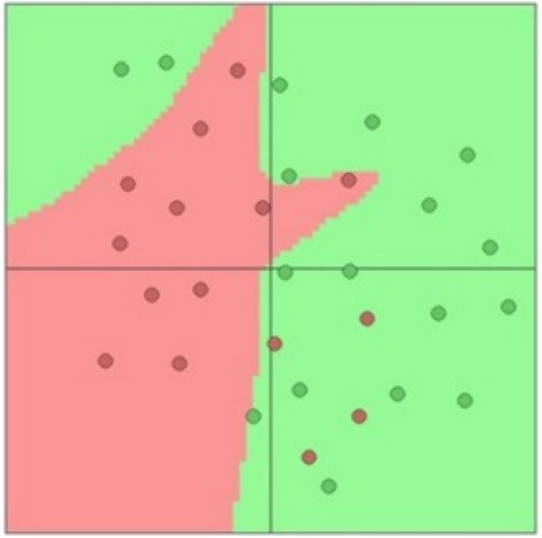
网络结构设计

两层全连接网络

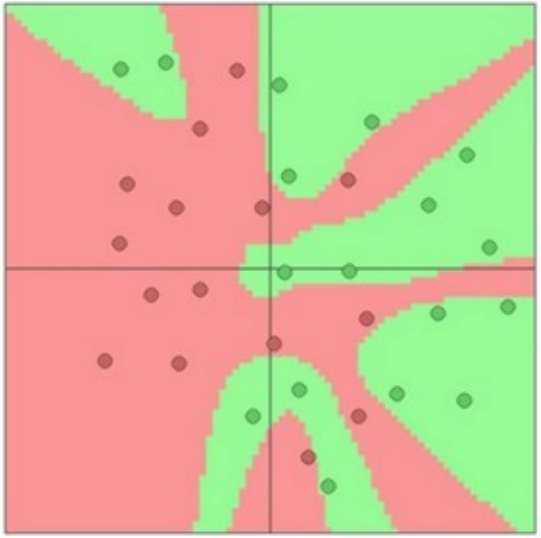
隐藏层3个神经元



隐藏层6个神经元



隐藏层20个神经元



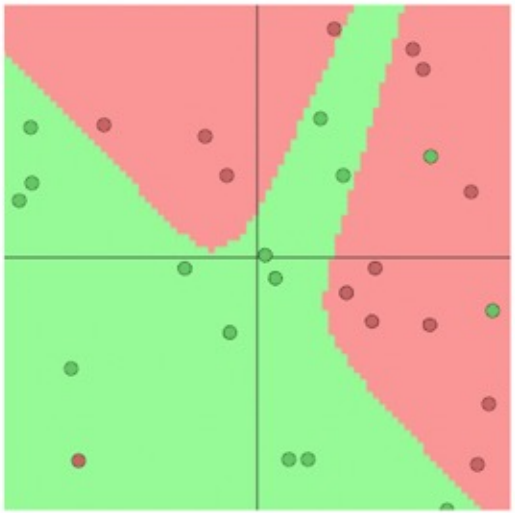
隐藏层神经元越多（越宽），
非线性分类能力越强。

自定义层数、神经元、激活函数、正则项，并可视化分界面：
https://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

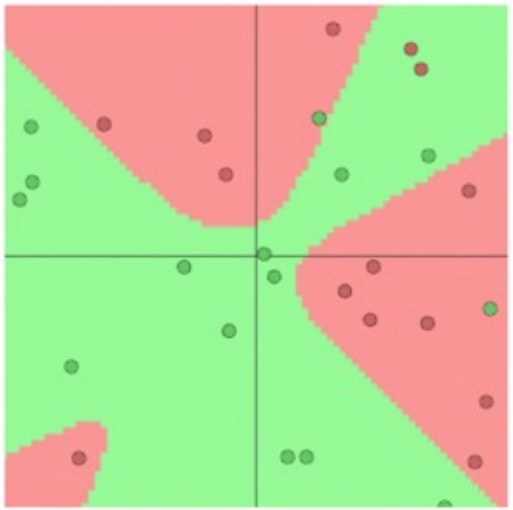
网络结构设计

每个隐藏层6个神经元

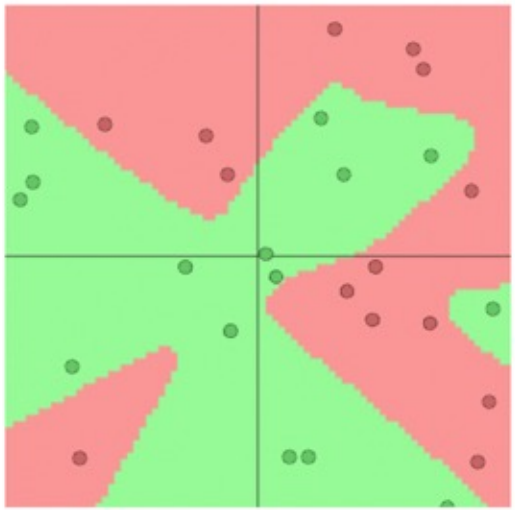
2层全连接网络



3层全连接网络



4层全连接网络



神经网络的层数越多（越深），
非线性分类能力越强。

自定义层数、神经元、激活函数、正则项，并可视化分界面：
https://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

损失函数的问题

在介绍线性分类器时，我们介绍过多类支持向量机损失（multiclass SVM loss/hinge loss）：

$$L_i = \sum_{j \neq y_i} \max(0, s_{ij} - s_{y_i} + 1)$$

但是，支持向量机（线性分类器的一种）和神经网络输出层的softmax（非线性）是两种完全不同的分类算法，多类支持向量机损失通常不会与softmax混用。

在softmax框架下，损失函数通常用交叉熵损失（cross entropy loss）。

交叉熵损失的本质

$f = W_2\,\varphi(W_1 x + b_1) + b_2$

0.6
-2.3
1.9

$q = softmax(f)$

鲨鱼分数最高
分类错误

0.21	海星
0.01	龙虾
0.78	鲨鱼

分类结果
(概率分布向量)

0.21	海星
0.01	龙虾
0.78	鲨鱼

q

差距

损失函数L

真实标签
(ground truth)

1	海星
0	龙虾
0	鲨鱼

p

熵的意义

熵

entropy

- $H(p) = -\sum_x p(x)\log p(x)$
- 体现有效信息量的多少

相对熵

Kullback – Leibler divergence

- $KL(p||q) = -\sum_x p(x)\log \frac{q(x)}{p(x)}$
- 两个分布之间的不相似程度，也叫KL散度

交叉熵

cross entropy

- $H(p,q) = -\sum_x p(x)\log q(x)$
- 两个概率分布间的差异

信息熵

熵：原本来自热力学领域，用于描述系统的混乱程度。

- 从微观角度看，熵是系统无序程度的度量，也就是说，熵越大，系统越混乱、越无序；反之，系统越有序，熵就越小。
- 从宏观角度看，熵与系统中热量的传递和转换有关，特别是与自发过程的方向性紧密相连。在一个孤立系统中，如果没有外部能量的输入，其总混乱程度（即熵）只可能增加，不可能自发减少，这就是熵增原理。

扩展到信息论领域，熵表示信息的不确定性或随机性。熵越大，表示信息的不确定性越大。

信息熵

足球比赛
极弱队vs极强队

p	
≈0	胜(w)
≈0	平(d)
≈1	负(l)

$$H(p) = -\sum_x p(x)logp\,(x)$$
$$= -[p(w)logp(w) + p(d)logp(d) + p(l)logp(l)]$$
$$\approx -[0 + 0 + 1\times \log(1)] = 0$$

熵为0，结果没有任何悬念（信息不确定性极小）

足球比赛
两队势均力敌

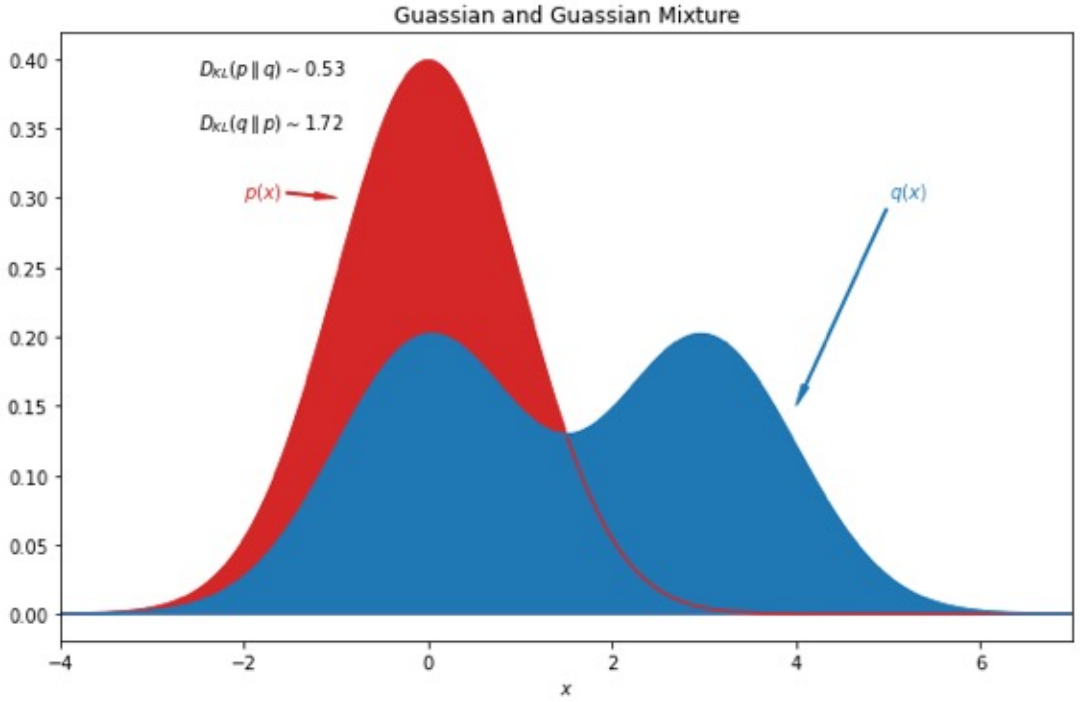
p	
1/3	胜(w)
1/3	平(d)
1/3	负(l)

$$H(p) = -\sum_x p(x)logp\,(x)$$
$$= -[p(w)logp(w) + p(d)logp(d) + p(l)logp(l)]$$
$$= -[\frac{1}{3}\times\log(\frac{1}{3}) + \frac{1}{3}\times\log(\frac{1}{3}) + \frac{1}{3}\times\log(\frac{1}{3})] \approx 0.477$$

熵为0.477，悬念达到最大

相对熵（KL散度）

Kullback–Leibler divergence：衡量p和q两个分布的不相似程度



$$KL(p||q) = -\sum_x p(x) log \frac{q(x)}{p(x)}$$

通常：$KL(p||q) \neq KL(q||p)$

交叉熵

$$H(p,q) = -\sum_x p(x)\log q(x)$$

$$= -\sum_x p(x)\log q(x) - \sum_x p(x)\log p(x) + \sum_x p(x)\log p(x)$$

$$= -\sum_x p(x)\log p(x) - \sum_x p(x)\log q(x) + \sum_x p(x)\log p(x)$$

$$= -\sum_x p(x)\log p(x) - \left(\sum_x p(x)\log q(x) - \sum_x p(x)\log p(x)\right)$$

$$= -\sum_x p(x)\log p(x) - \sum_x p(x)\log \frac{q(x)}{p(x)}$$

$$= H(p) + KL(p||q)$$

交叉熵＝熵＋相对熵

交叉熵损失

分类结果
（概率分布向量）

0.21	海星
0.01	龙虾
0.78	鲨鱼

q

交叉熵损失

真实标签
(ground truth)

1	海星
0	龙虾
0	鲨鱼

p

$$H(p,q) = H(p) + KL(p||q)$$

真实标签为one hot形式，因此 $H(p) = 0$

$$H(p,q) = KL(p||q)$$

分类器场景下，可以用交叉熵来衡量分类结果和真实标签这两个概率分布向量之间的差距。

交叉熵损失

分类结果
（概率分布向量）

0.21	海星
0.01	龙虾
0.78	鲨鱼

q

交叉熵损失

真实标签
(ground truth)

1	海星
0	龙虾
0	鲨鱼

p

$$H(p,q) = -\sum_x p(x) log q(x)$$

$$L = -\big(p(\text{海星}) \times log q(\text{海星}) + p(\text{龙虾}) \times log q(\text{龙虾}) + p(\text{鲨鱼}) \times log q(\text{鲨鱼})\big)$$
$$= -(1 \times \log(0.21) + 0 \times \log(0.01) + 0 \times \log(0.78))$$
$$= -(1 \times \log(0.21)) \approx 0.67$$

损失对比

分类器

f

0.6	海星
-2.3	龙虾
1.9	鲨鱼

多类支持向量机损失

$L_i = \sum_{j \neq y_i} \max(0, s_{ij} - s_{y_i} + 1)$

$= \max(0, -2.3 - 0.6 + 1) + \max(0, 1.9 - 0.6 + 1)$

$= \max(0, -1.9) + \max(0, 2.3) = 2.3$

softmax

1	海星
0	龙虾
0	鲨鱼

p

0.21	海星
0.01	龙虾
0.78	鲨鱼

q

交叉熵损失

$L_i = -\sum_x p(x) \log q(x)$

$= -(1 \times \log(0.21) + 0 \times \log(0.01) + 0 \times \log(0.78))$

$= -(1 \times \log(0.21)) \approx 0.67$

损失对比

模型	海星	龙虾	鲨鱼
A	10	-2	3
B	10	9	9
C	10	-100	-100

softmax

模型	海星	龙虾	鲨鱼
A	0.99	6.14×10^{-6}	2.26×10^{-6}
B	0.58	0.21	0.21
C	≈1	≈0	≈0

三个模型的损失都大于0

交叉熵损失

模型A

$-\log(0.99) = 0.0043$

模型B

$-\log(0.58) = 0.237$

模型C

$-\log(\approx 1) = 1.69 \times 10^{-48}$

模型B的“置信度”不够高

三个模型的损失都为0

多类SVM损失

模型A

$\max(0, (-2) - 10 + 1) + \max(0, -3 - 10 + 1) = 0$

模型B

$\max(0, 9 - 10 + 1) + \max(0, 9 - 10 + 1) = 0$

模型C

$\max(0, (-100) - 10 + 1) + \max(0, (-100) - 10 + 1) = 0$

无法判断ABC的“置信度”

损失对比

模型	海星	龙虾	鲨鱼
A	10	-2	3
B	10	9	9
C	10	-100	-100

softmax

模型	海星	龙虾	鲨鱼
A	0.99	6.14×10^{-6}	2.26×10^{-6}
B	0.58	0.21	0.21
C	≈1	≈0	≈0

模型B还不够好，需要继续训练！

交叉熵损失

模型A

$$-\log(0.99) = 0.0043$$

模型B

$$-\log(0.58) = 0.237$$

模型C

$$-\log(\approx 1) = 1.69\times10^{-48}$$

模型B的“置信度”不够高

三个模型都达到要求，可以停止训练了！

多类SVM损失

模型A

$$\max(0,(-2)-10+1)+\max(0,-3-10+1)=0$$

模型B

$$\max(0,9-10+1)+\max(0,9-10+1)=0$$

模型C

$$\max(0,(-100)-10+1)+\max(0,(-100)-10+1)=0$$

无法判断ABC的“置信度”

前向传播和反向传播

计算图

计算图（computation graph）是一种用于描述数学计算的图形模型。

计算图是一种有向无环图，用于表达输入、输出以及中间变量之间的计算关系。

在计算图中，节点代表数学运算（“运算门”单元），边代表运算结果之间的依赖关系。

作用和构建

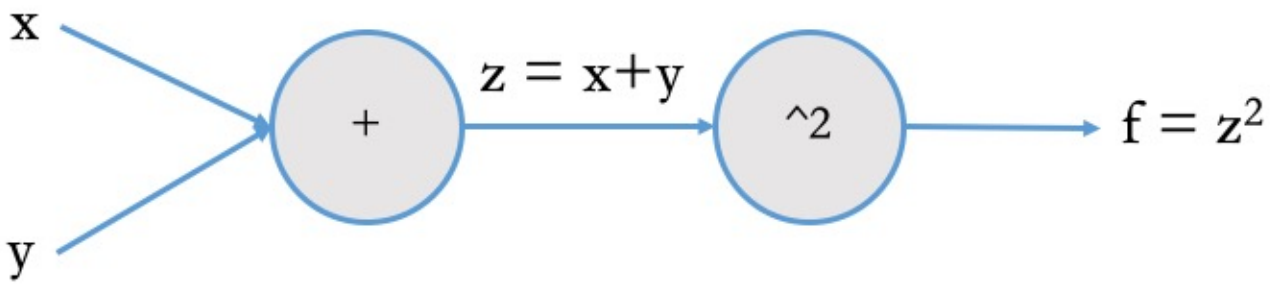
计算图在神经网络训练过程中起着关键作用，用于组织和可视化复杂的计算过程。

通过计算图，可以更有效地进行梯度计算和参数更新，从而提高神经网络的训练效率。它常用于神经网络的前向传播和反向传播算法中。

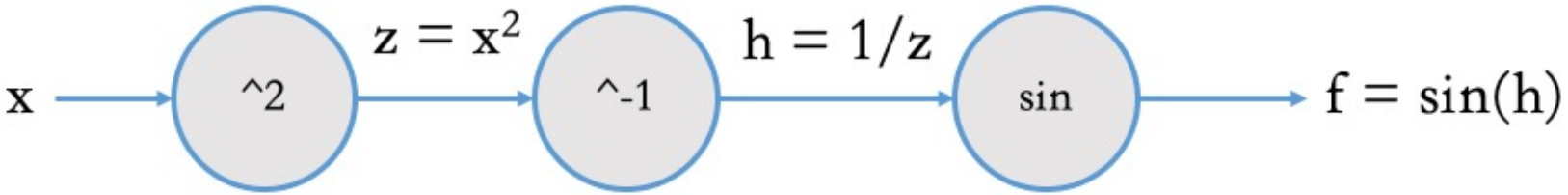
在构建计算图时，需要注意颗粒度的问题。如果函数过于复杂，可以将多个节点组合成一个大的节点，以简化计算过程。

计算图

函数$f = (x + y)^2$的计算图

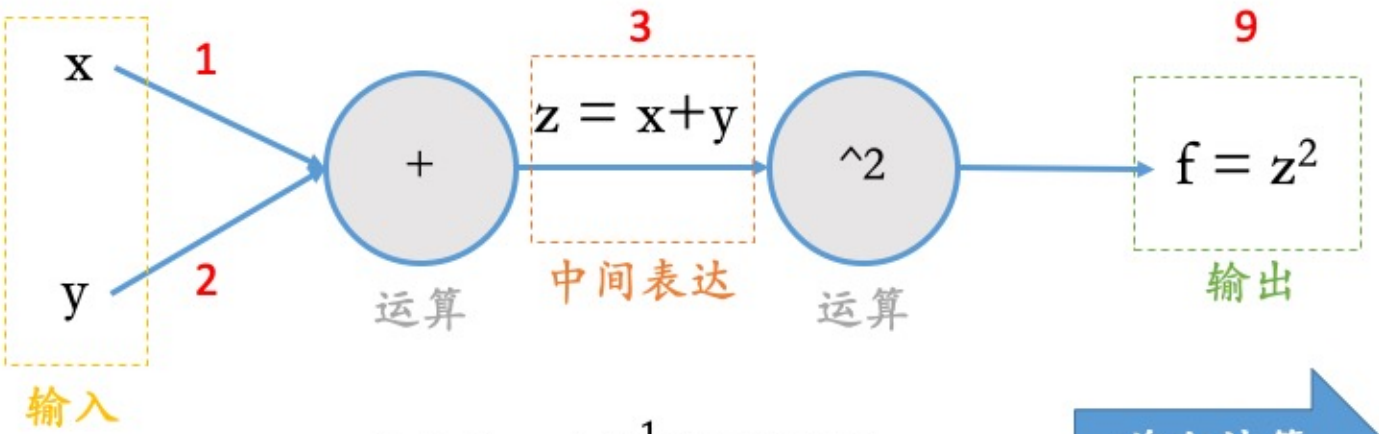


函数$f = \sin(\frac{1}{x^2})$的计算图



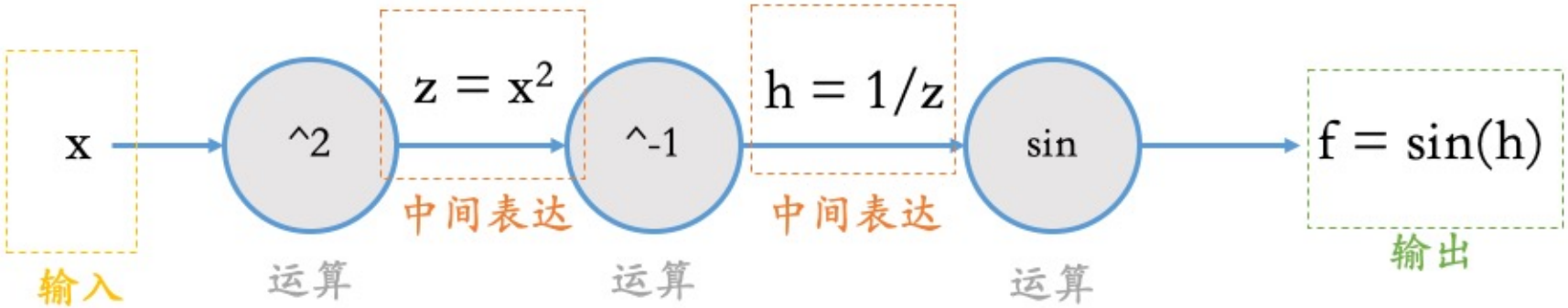
前向传播

函数$f=(x+y)^2$的计算图



函数$f=\sin(\frac{1}{x^2})$的计算图

前向计算

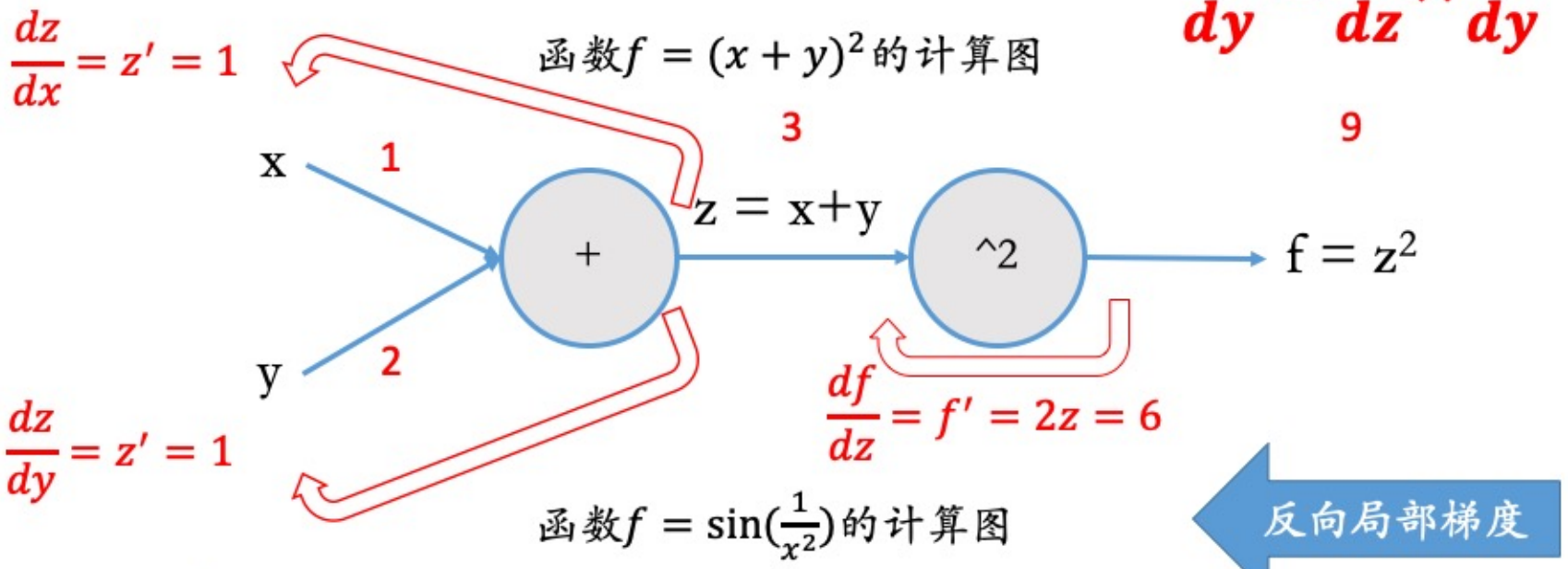


反向传播

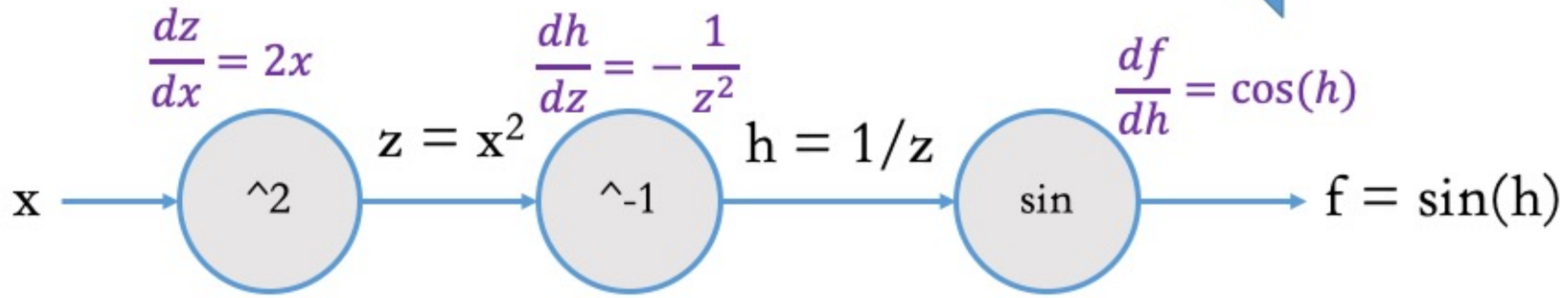
链式法则

$$\frac{df}{dx}=\frac{df}{dz}\times\frac{dz}{dx}$$

$$\frac{df}{dy}=\frac{df}{dz}\times\frac{dz}{dy}$$



反向局部梯度

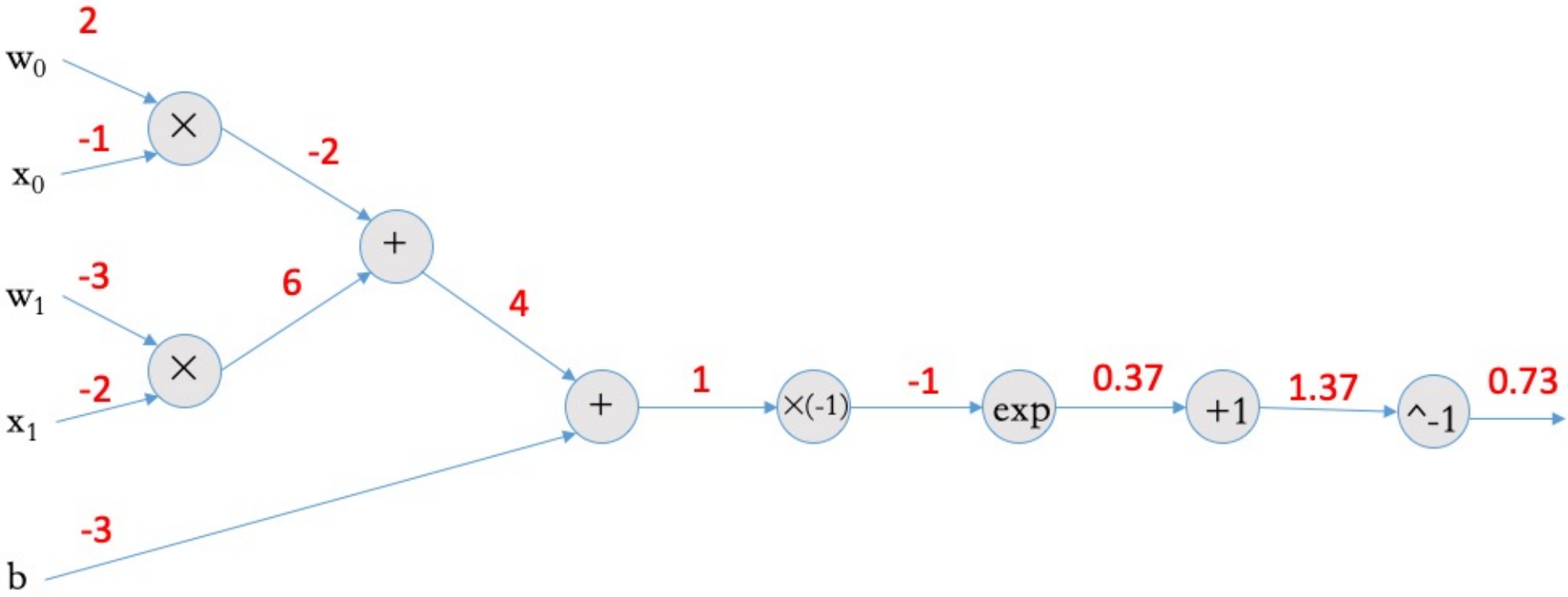


$$\frac{df}{dx}=\frac{df}{dh}\times\frac{dh}{dz}\times\frac{dz}{dx}$$

前向传播和反向传播：举例

前向计算

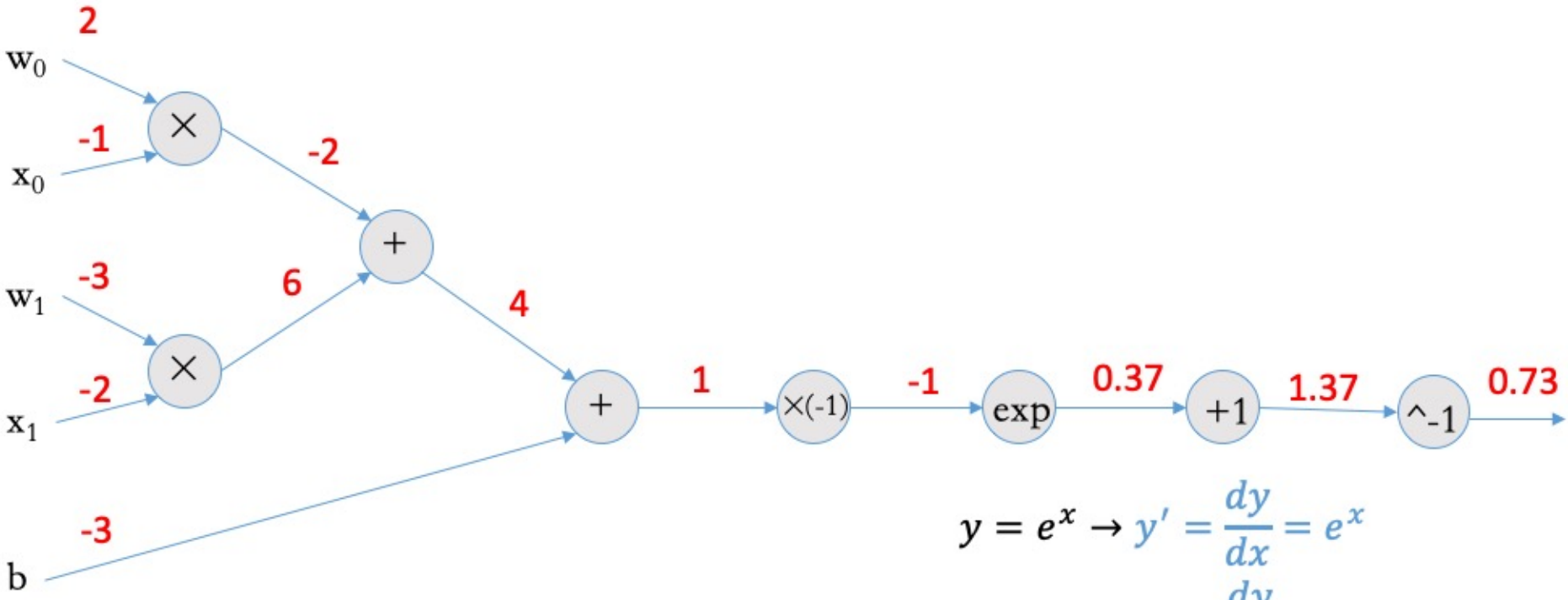
$$f(w,x,b) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$



前向传播和反向传播：举例

反向计算

$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$



常用导数

$$y=e^x \rightarrow y'=\frac{dy}{dx}=e^x$$

$$y=kx \rightarrow y'=\frac{dy}{dx}=k$$

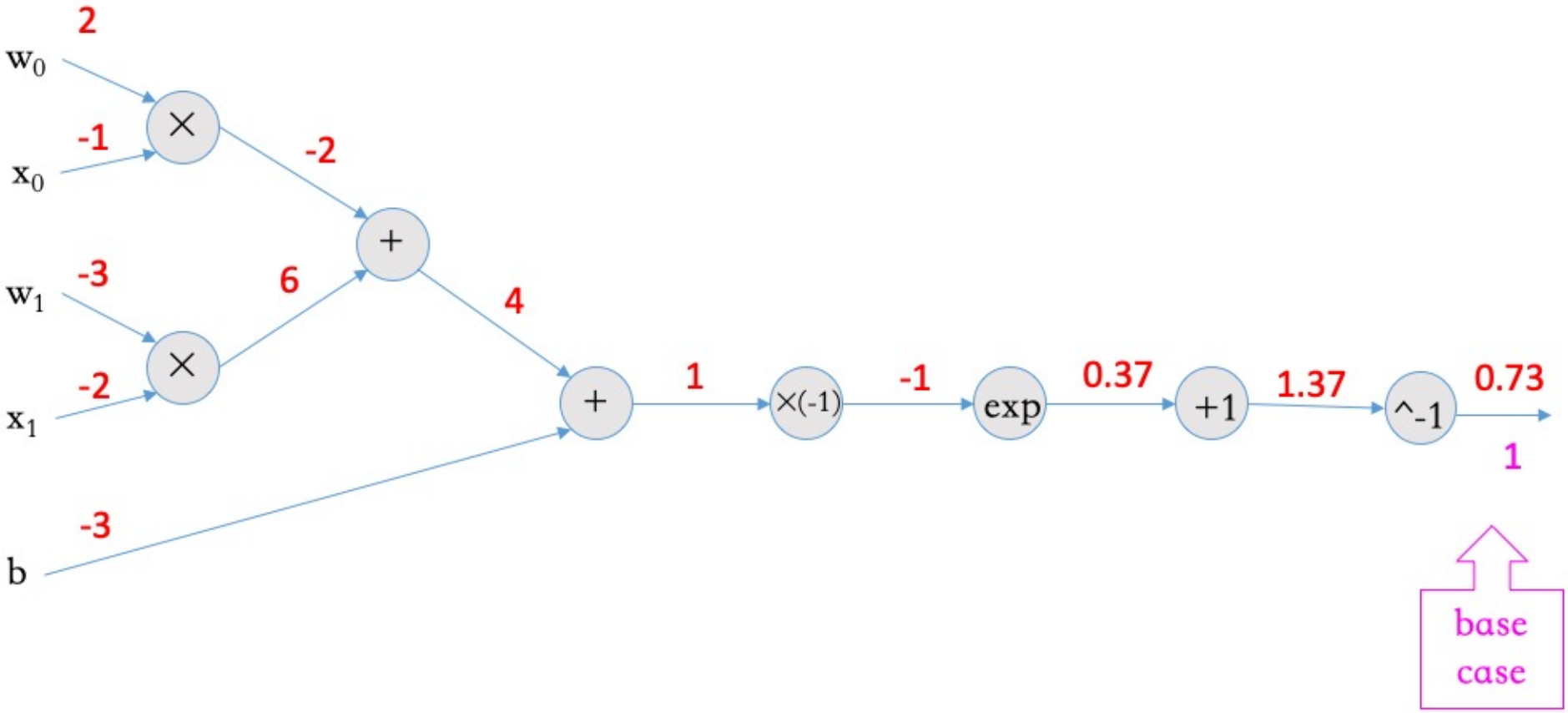
$$y=\frac{1}{x} \rightarrow y'=\frac{dy}{dx}=-\frac{1}{x^2}$$

$$y=x+b \rightarrow y'=\frac{dy}{dx}=1$$

前向传播和反向传播：举例

反向计算

$$f(w,x,b) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$

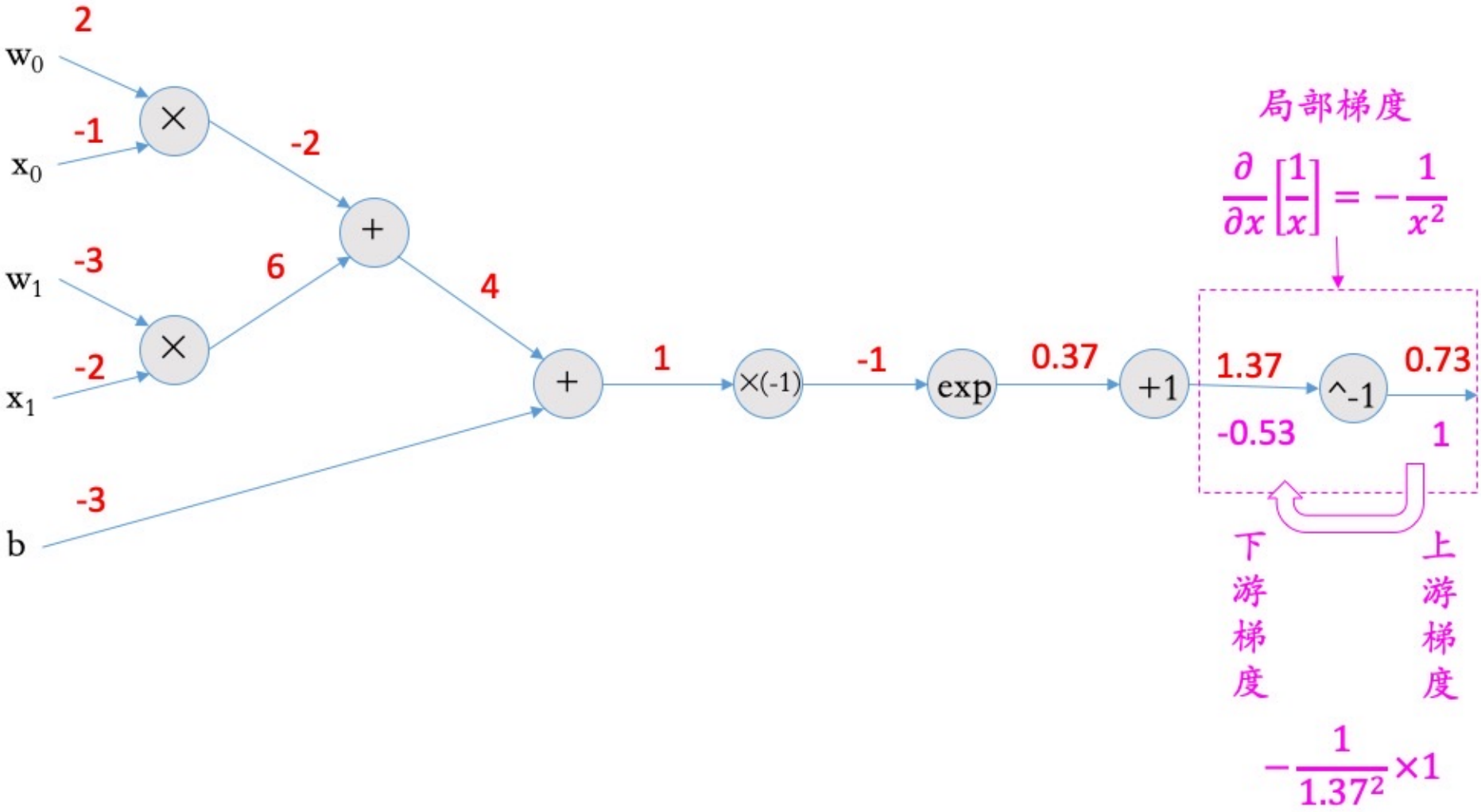


前向传播和反向传播：举例

反向计算

下游梯度=局部梯度×上游梯度

$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$

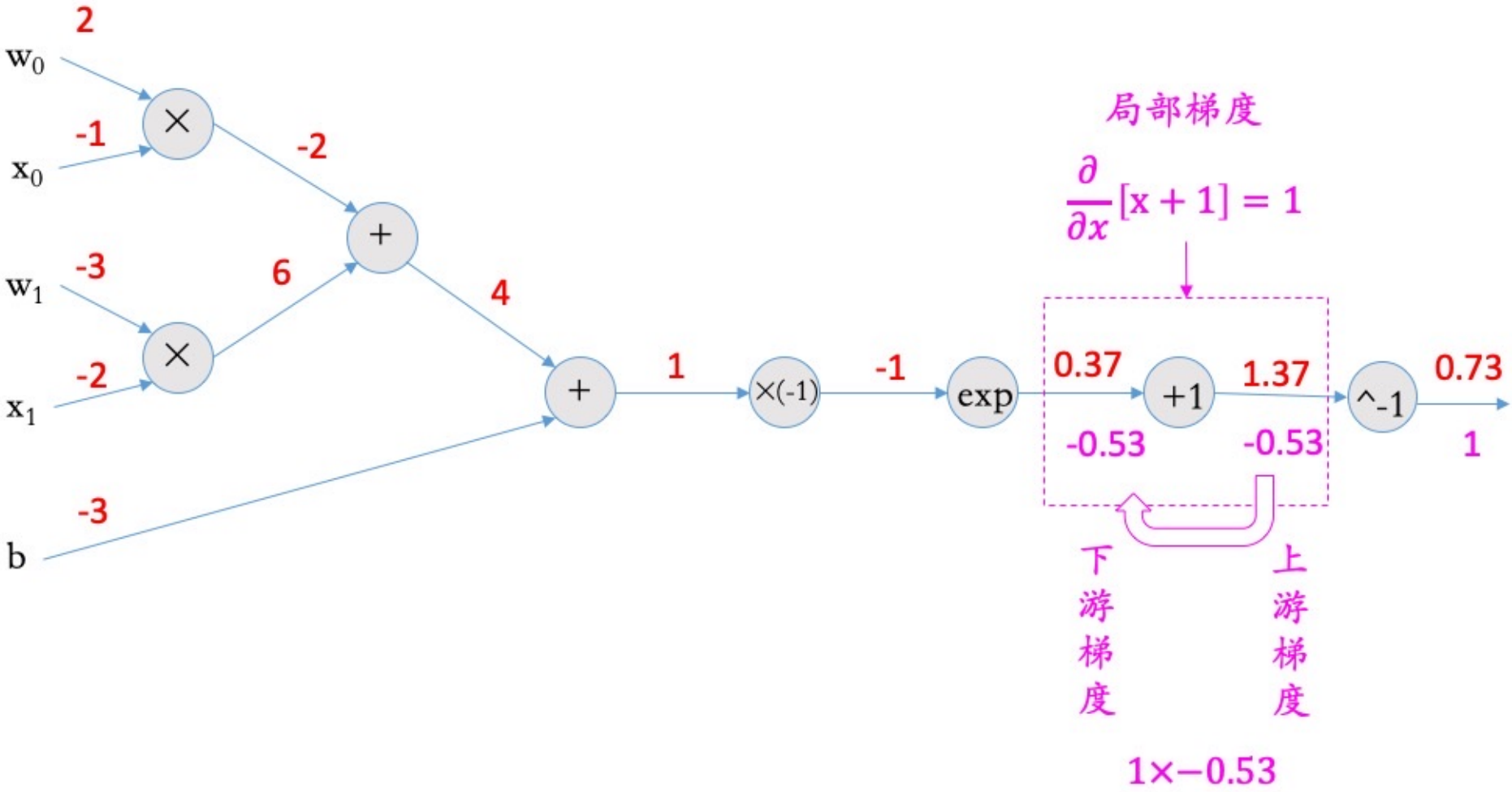


前向传播和反向传播：举例

反向计算

下游梯度=局部梯度×上游梯度

$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$

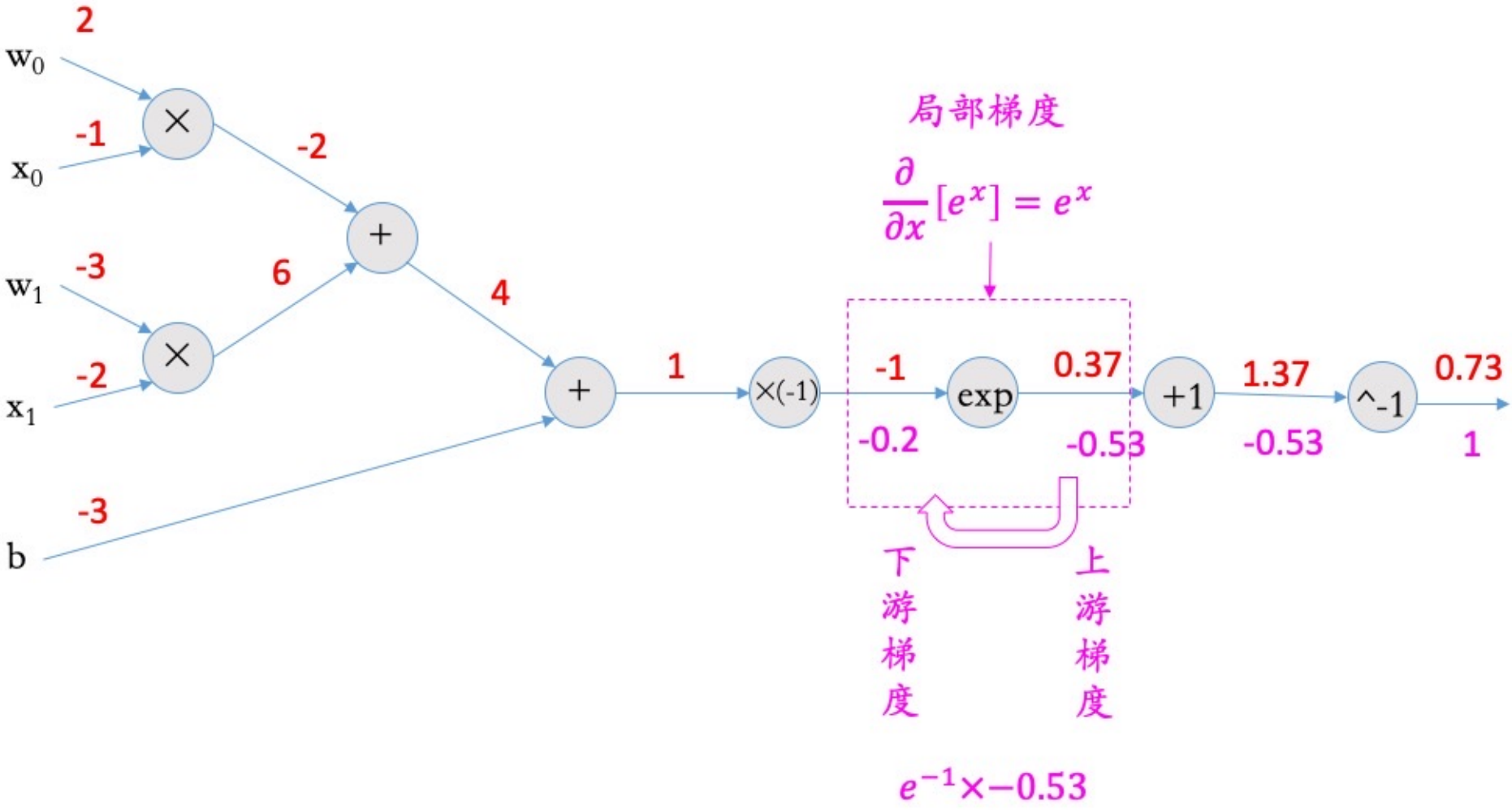


前向传播和反向传播：举例

反向计算

下游梯度=局部梯度×上游梯度

$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$

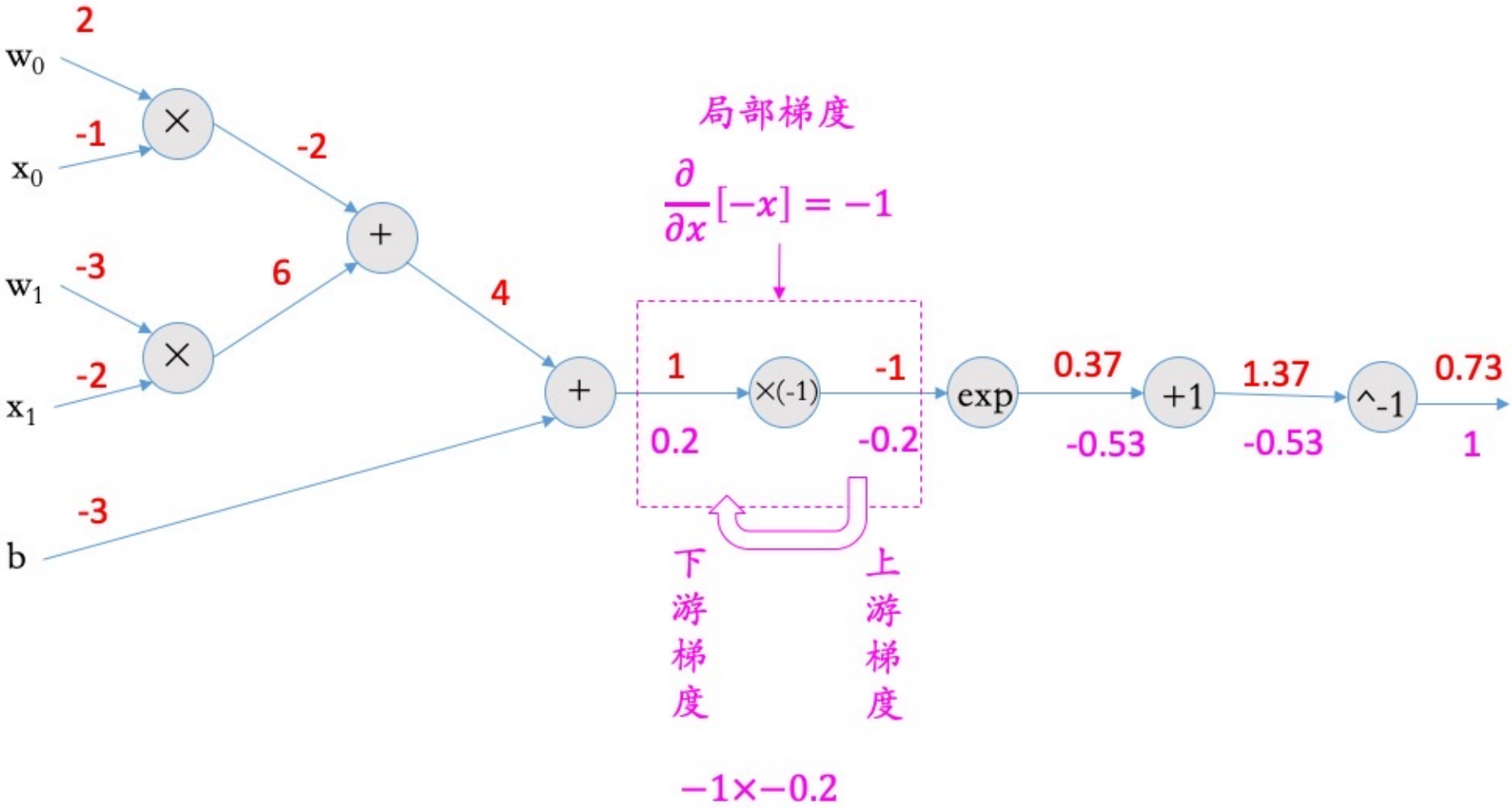


前向传播和反向传播：举例

反向计算

下游梯度=局部梯度×上游梯度

$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$

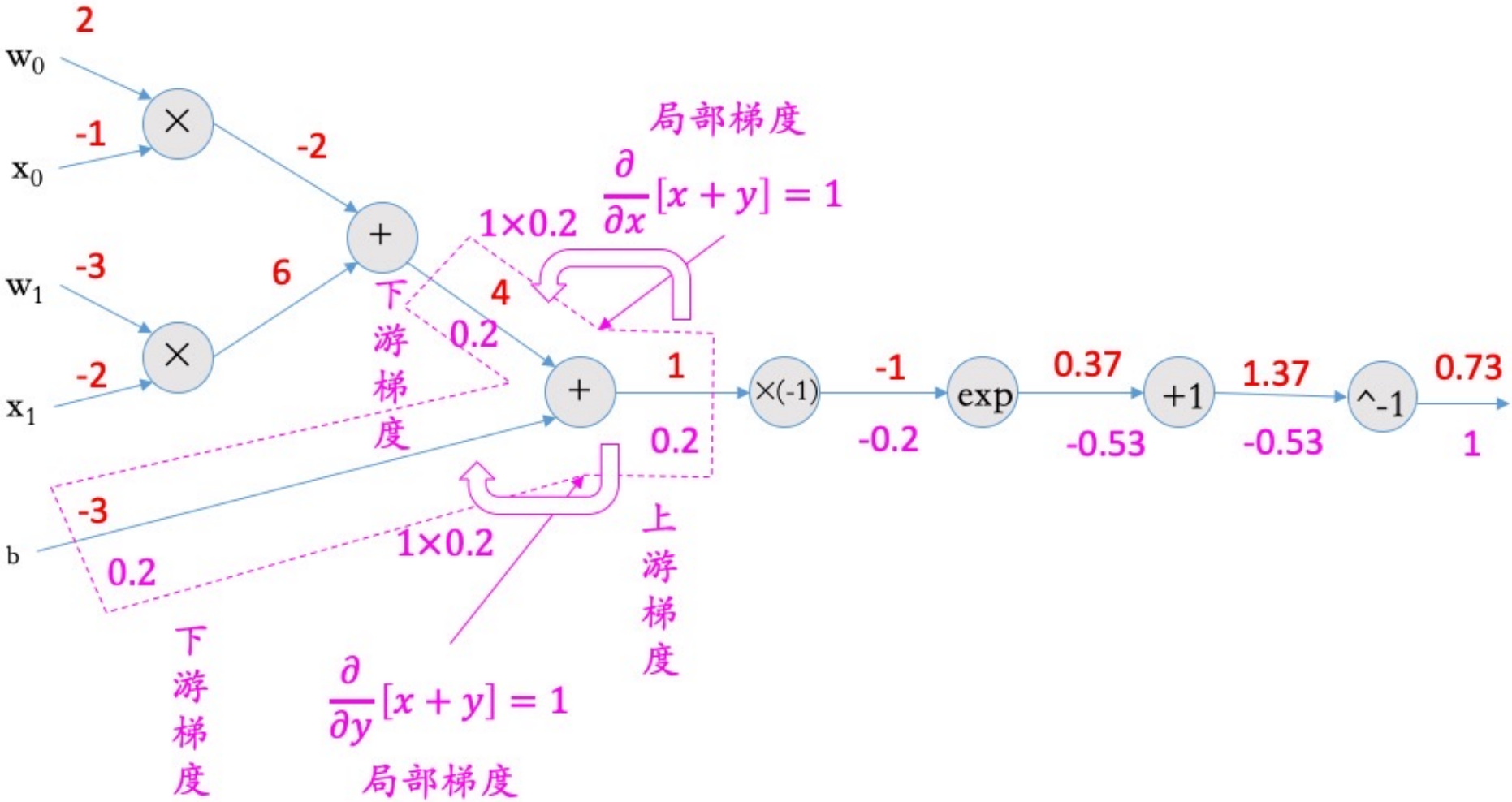


前向传播和反向传播：举例

反向计算

下游梯度=局部梯度×上游梯度

$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$

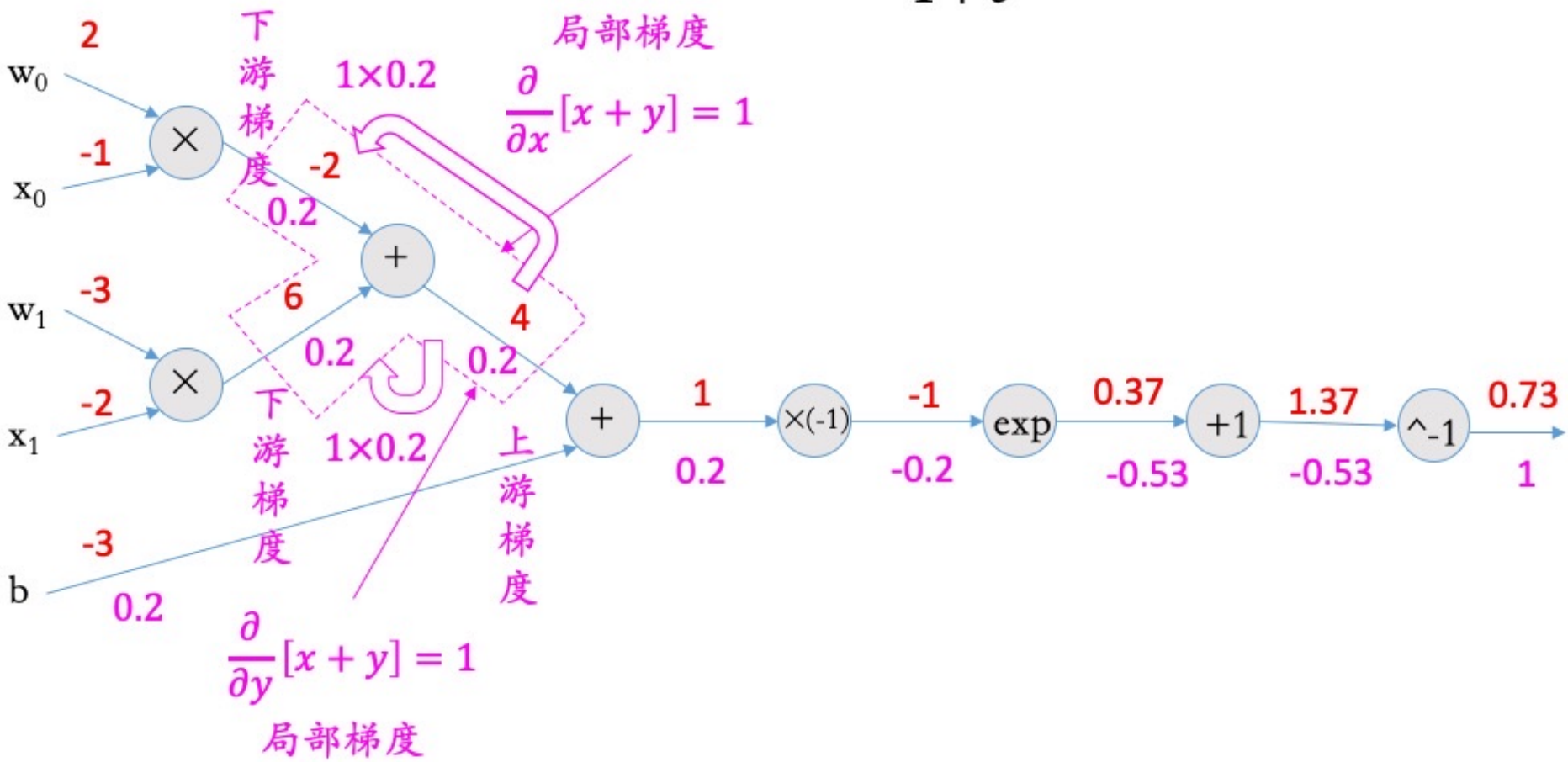


前向传播和反向传播：举例

反向计算

下游梯度=局部梯度×上游梯度

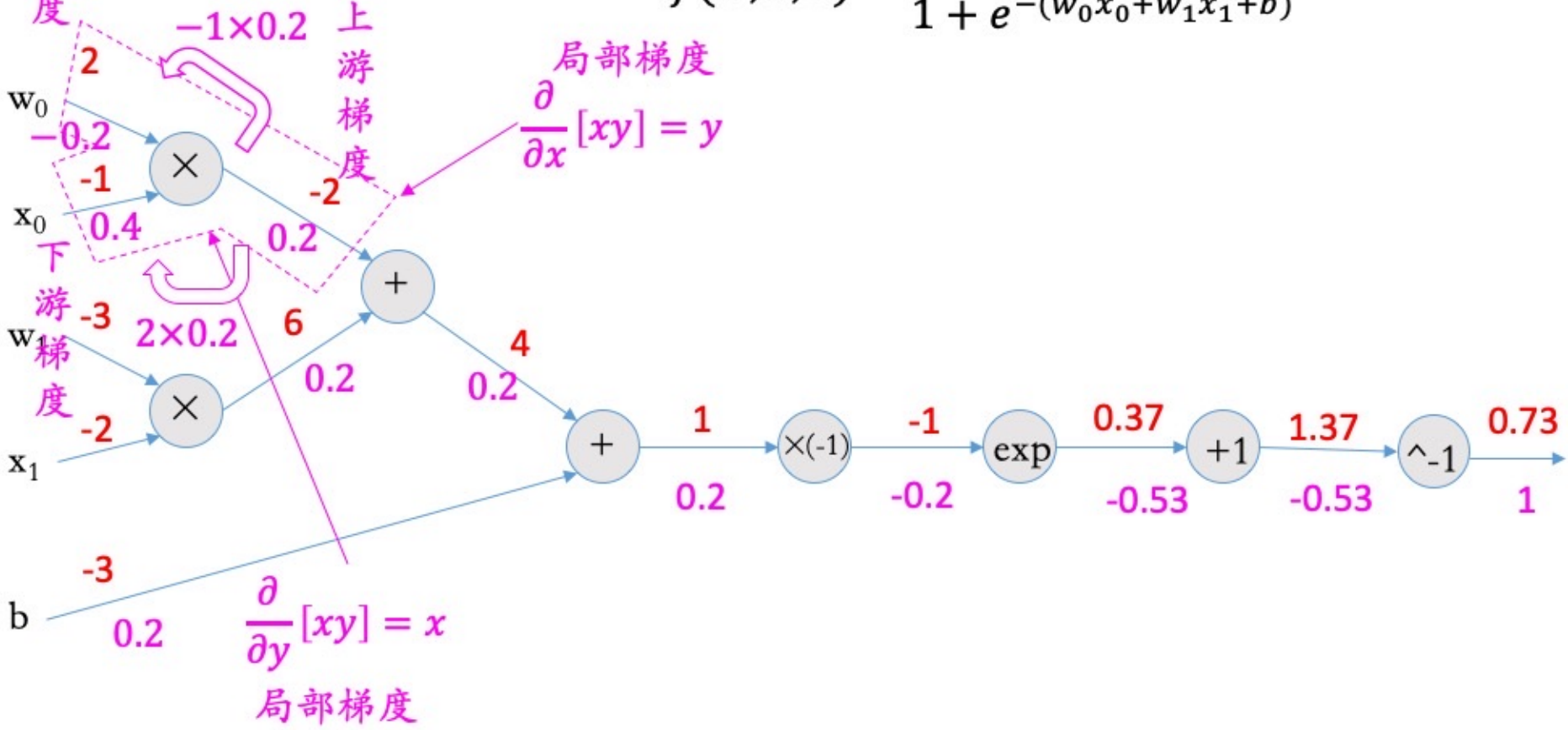
$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$



前向传播和反向传播：举例

反向计算

$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$

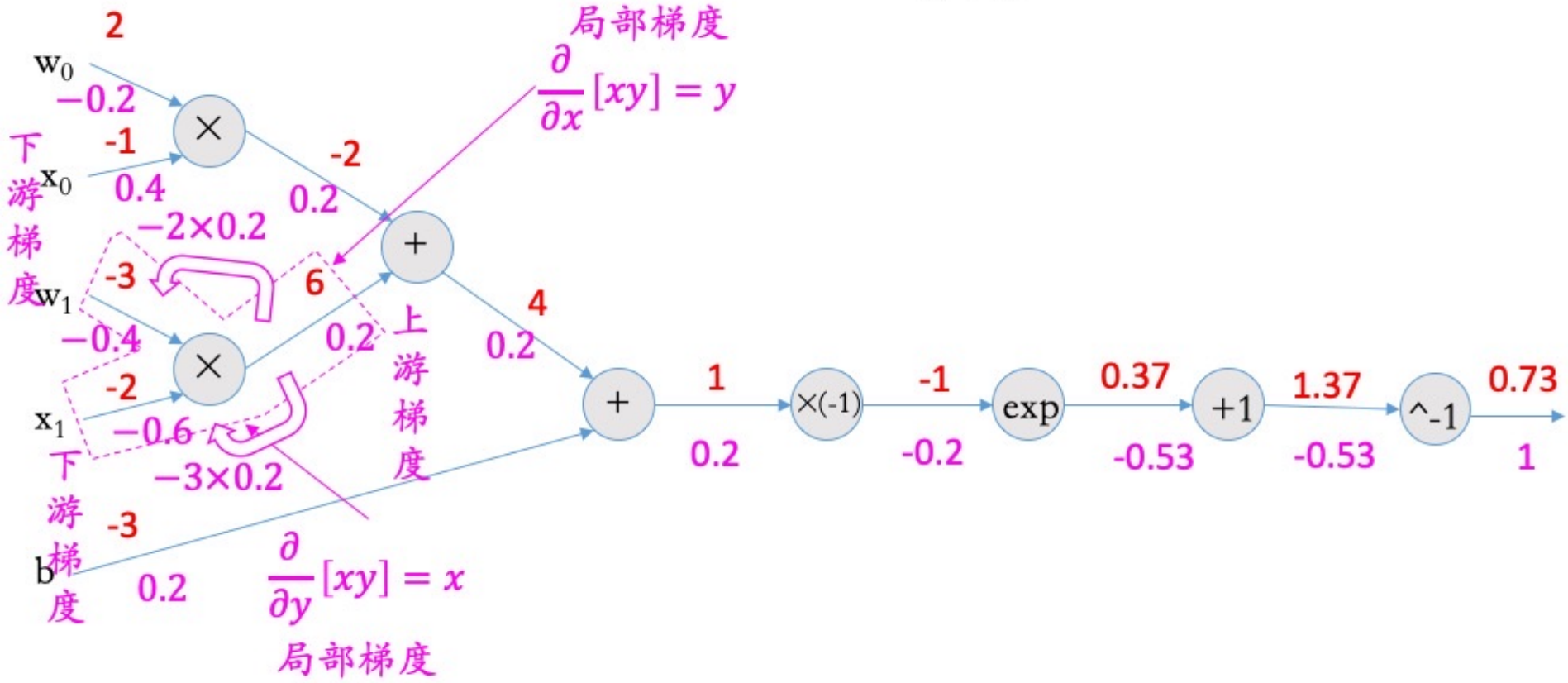


前向传播和反向传播：举例

反向计算

下游梯度=局部梯度×上游梯度

$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$

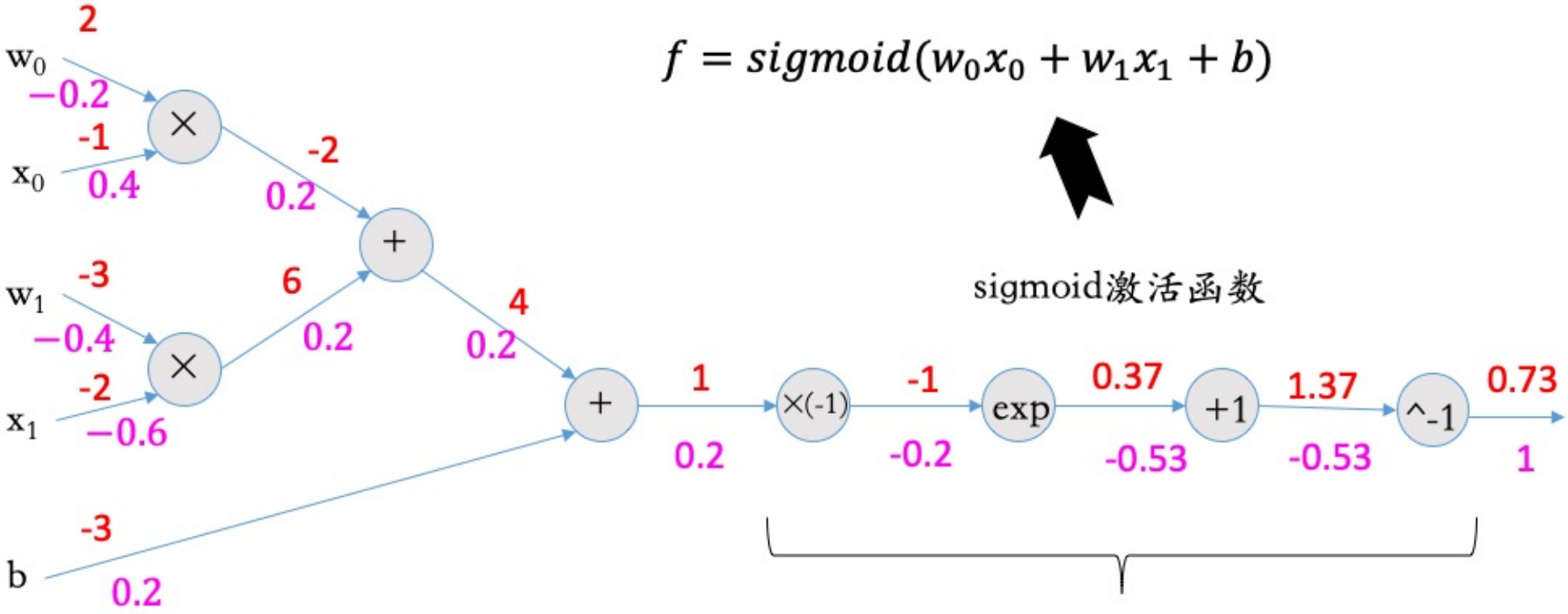


前向传播和反向传播：举例

下游梯度=局部梯度×上游梯度

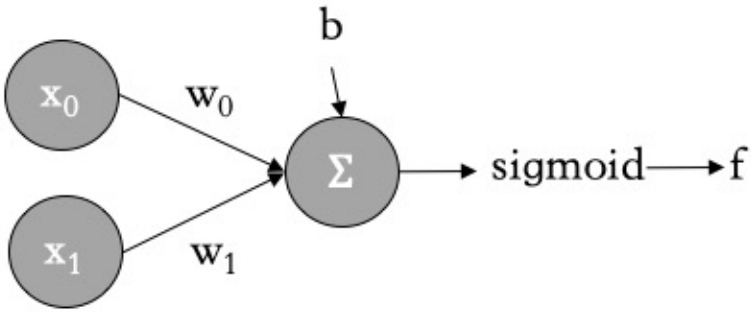
$$f(w,x,b)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+b)}}$$

$$f=sigmoid(w_0x_0+w_1x_1+b)$$



sigmoid激活函数

$$\varphi(x)=\frac{1}{1+e^{-x}}$$

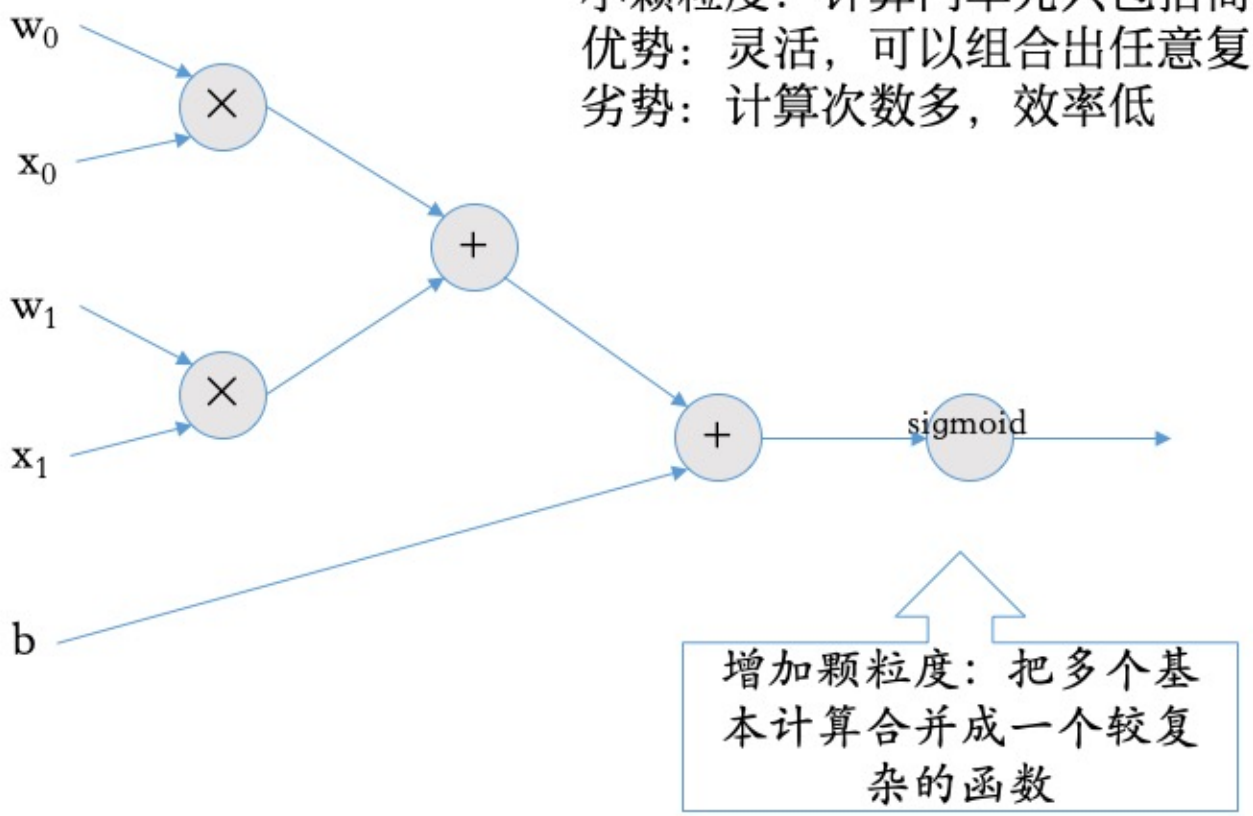


计算图的要点

- 计算图可以表达任意的复杂函数。
- 每个计算门单元根据输入值计算输出值，并可计算该单元输出值和输入值之间的局部梯度。
- 可以利用链式法则，计算整个网络的输出结果和该计算门单元输入值之间的梯度→回溯到最初的输入层，可计算整个网络的梯度。

计算图的颗粒度

小颗粒度：计算门单元只包括简单计算
优势：灵活，可以组合出任意复杂函数
劣势：计算次数多，效率低



$$f = sigmoid(w_0x_0 + w_1x_1 + b)$$

计算图的颗粒度

Caffe

Caffe

颗粒度较大
优点：计算快
缺点：需要定义复杂函数

TensorFlow

颗粒度小
优点：定义简单、组合灵活
缺点：计算效率低

Pop Quiz

1. CIFAR-10中的图像经过flatten操作后，形成的向量长度为?

A. 680 | B. 1024 | C. 3072 | D. 5120

2. 线性分类器中，权值矩阵W的行数为?

A. 图像向量长度 | B. 任务类别数 | C. 数据集样本数 | D. 图像的边长

3.多类支撑向量机损失（合页损失）取值范围的最小值是多少?

A. $-\infty$ | B. -1 | C. 0 | D. 1

4. 在阴性/阳性的二分类任务中，描述“所有被分类为阳性的样本中，真实的阳性样本占多少？”的是以下哪一项?

A. 精准度 Precision | B. 召回率 Recall | C. 正确率 Accuracy | D. F1分数

5. 以下哪一项不是超参数?

A. 损失函数的正则化占比 | B. 神经网络的层数 | C. 学习率 | D. 偏置