机器学习 - 人工神经网络

本章目录

1. 人工神经网络发展历史

01 发展历史

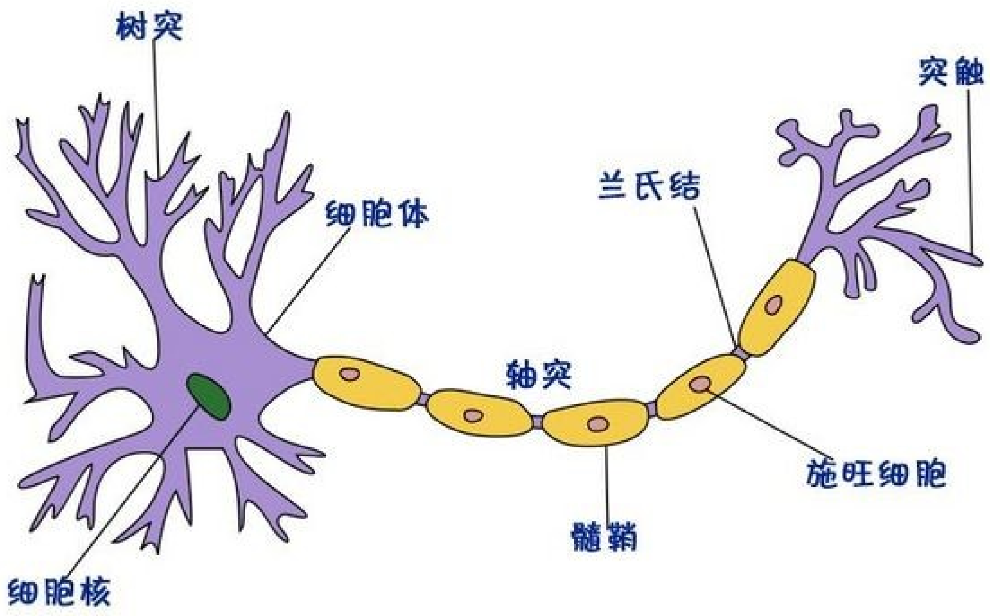
02 感知机算法

03 BP 算法

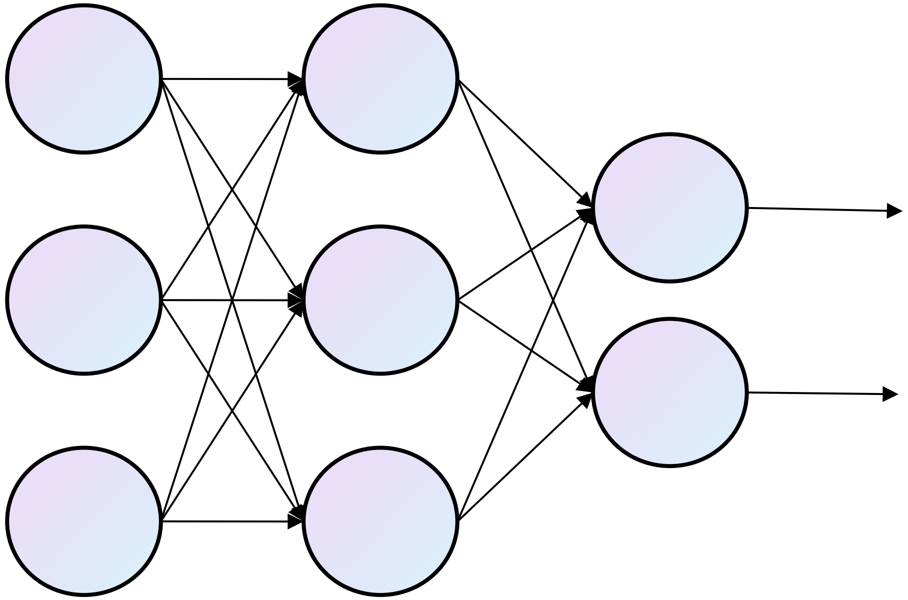
1. 人工神经网络发展历史

发展历史

1943 年，心理学家 McCulloch 和逻辑学家 Pitts 建立神经网络的数学模型， MP 模型



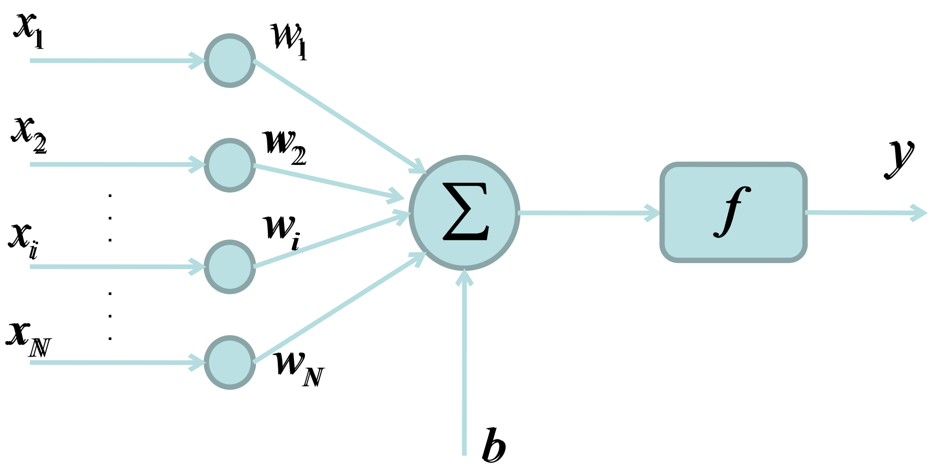
神经元生理结构



神经元数学模型

1. 人工神经网络发展历史

1960 年代，人工网络得到了进一步地发展感知机和自适应线性元件等被提出。M.Minsky 仔细分析了以感知机为代表的神经网络的局限性，指出了感知机不能解决非线性问题，这极大影响了神经网络的研究。

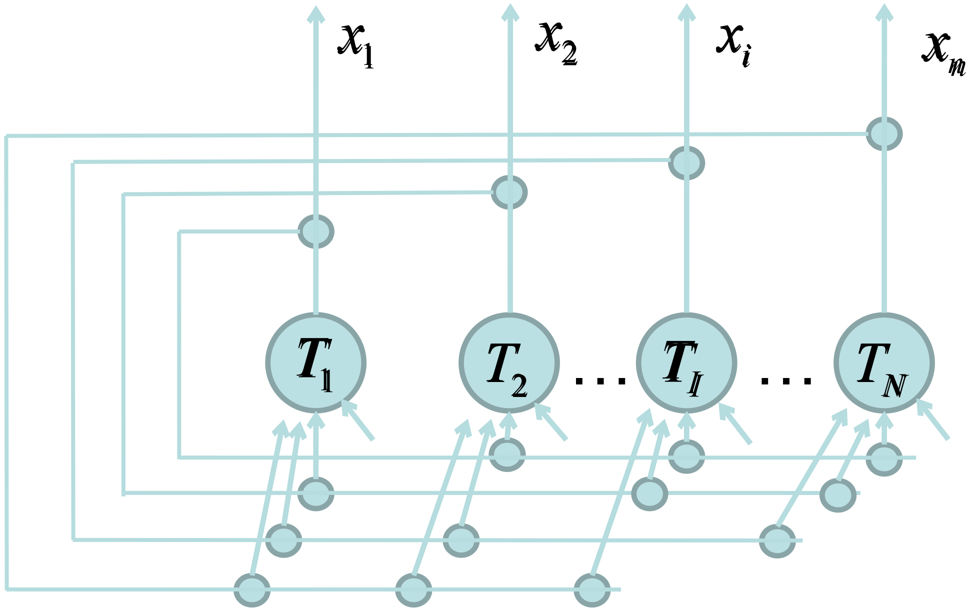


$$y = f\left(\sum_{i=1}^{N} w_i x_i + b\right)$$

单层感知机的数学模型

1. 人工神经网络发展历史

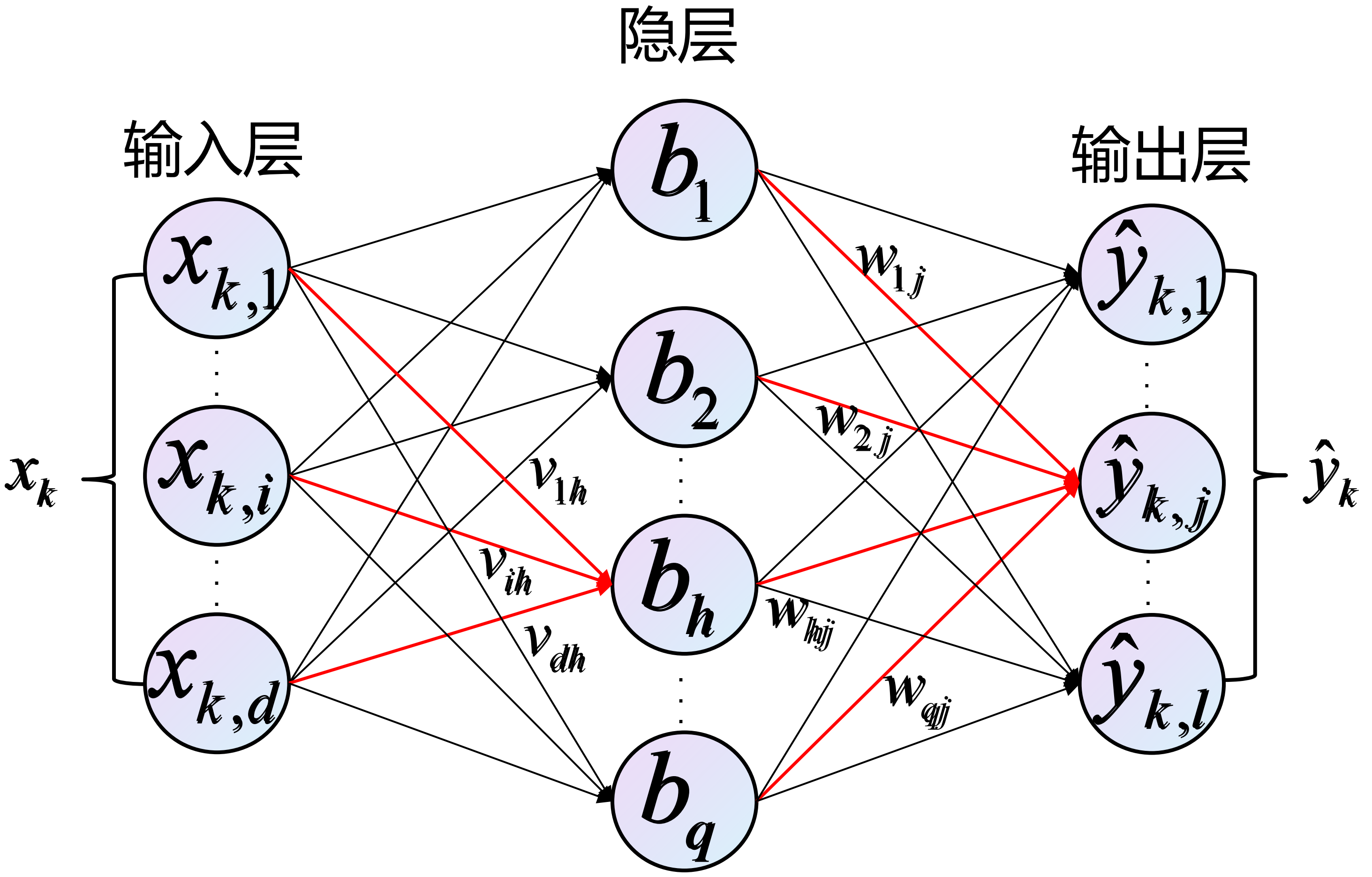
1982 年，加州理工学院 J.J.Hopfield 教授提出了 Hopfield 神经网络模型，引入了计算能量概念，给出了网络稳定性判断。



离散 Hopfield 神经网络模型

1. 人工神经网络发展历史

1986 年， Rumelhart 和 McClelland 为首的科学家提出了 BP （ Back Propagation ）神经网络的概念，是一种按照误差逆向传播算法训练的多层前馈神经网络，目前是应用最广泛的神经网络。



BP 神经网络模型

2. 感知器算法

2. 感知机算法

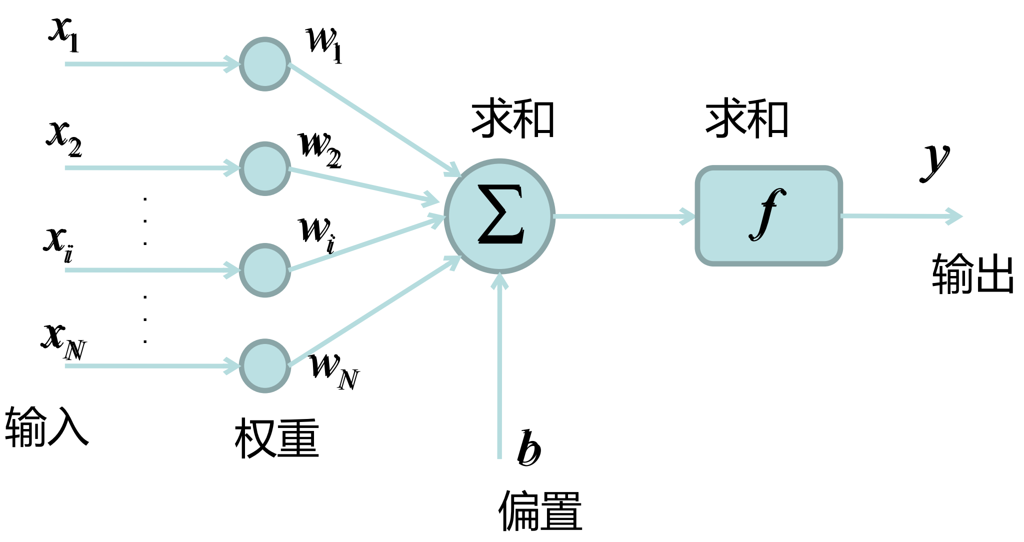
感知机（ Perceptron ）是二分类问题的线性分类模型。

用　表示数据集，用 表示标签。

需要学习的目标函数是

$$f(x) = \mathrm{sign}(w^T x + b)$$

从一堆输入输出中学习模型参数和。



2. 感知机算法

感知机算法（ Perceptron Algorithm ）：

随机选择模型参数的初始值。

选择一个训练样本。

若判别函数，且，则，。

若判别函数，且，则，。

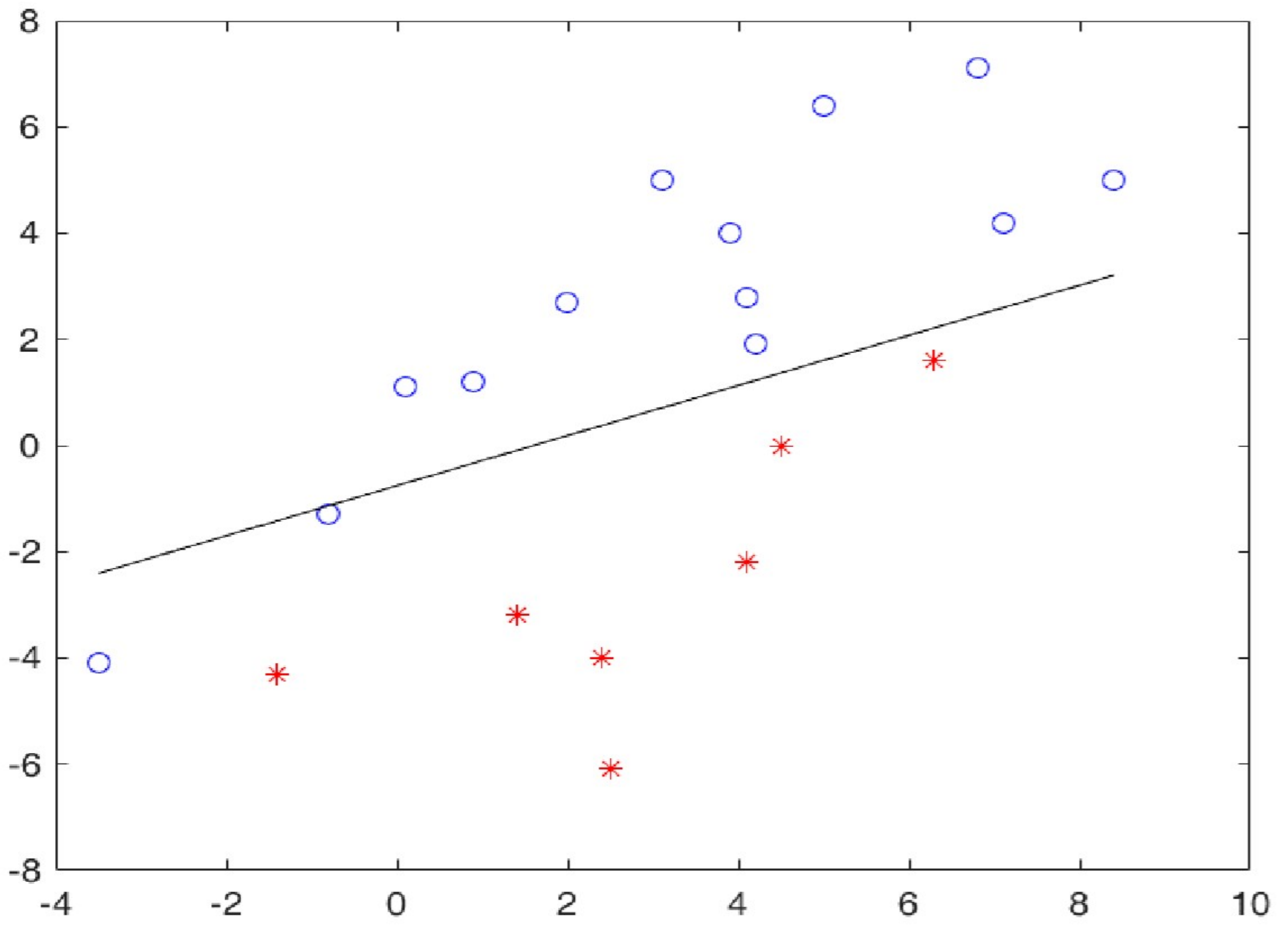
再选取另一个训练样本，回到 2 。

终止条件：直到所有数据的输入输出对都不满足 2 中的 (i) 和 (ii) 中之一，则退出循环。

2. 感知机算法

算法演示 分类问题

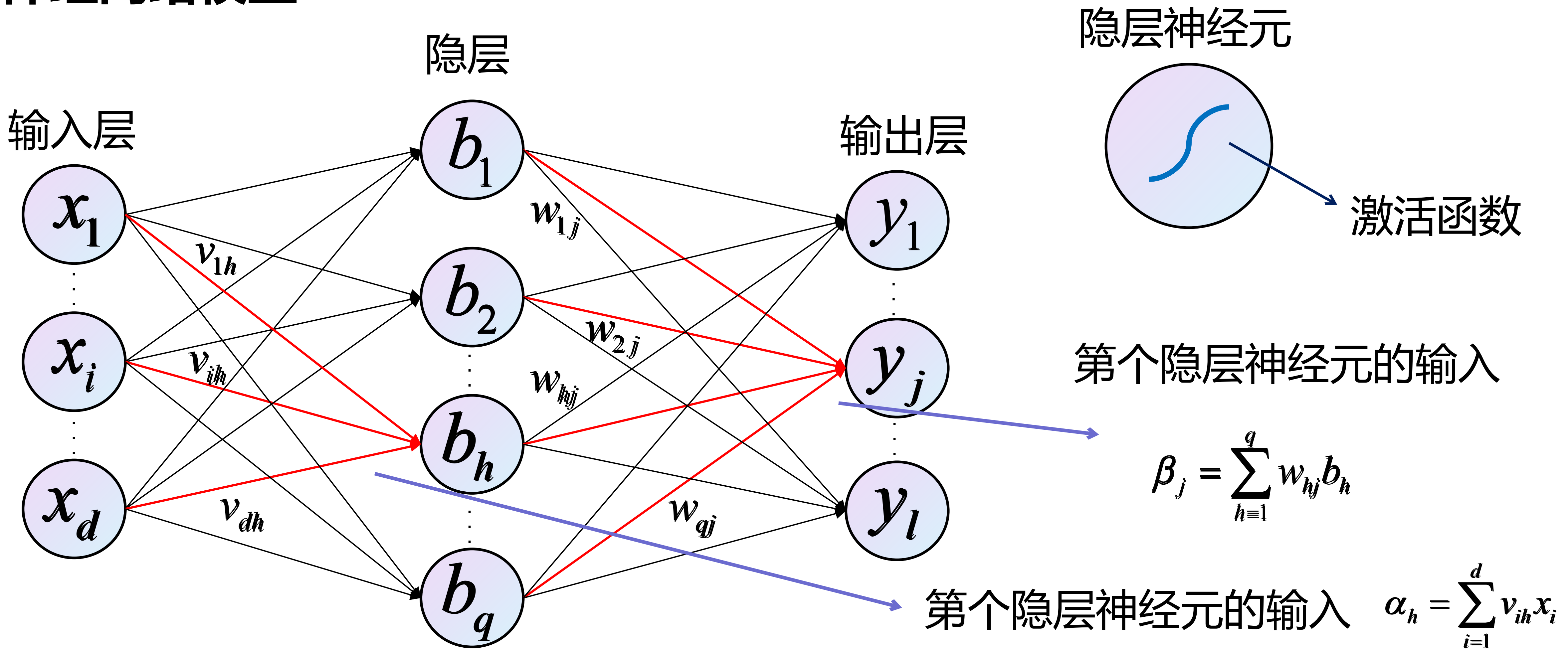
单层感知机只能处理线性问题，无法处理非线性问题！！



2. 感知器算法

3.BP 算法

神经网络模型

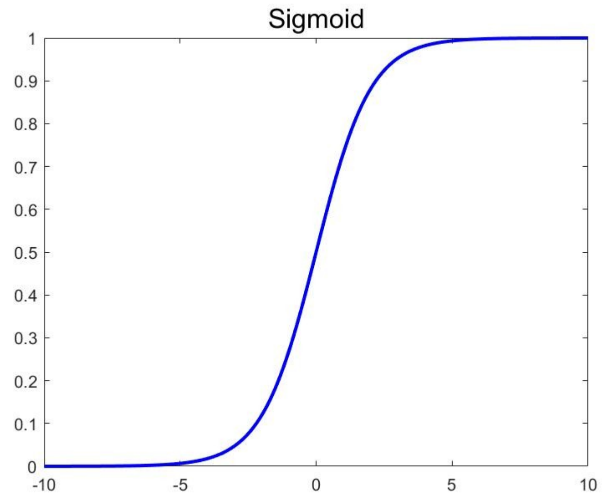


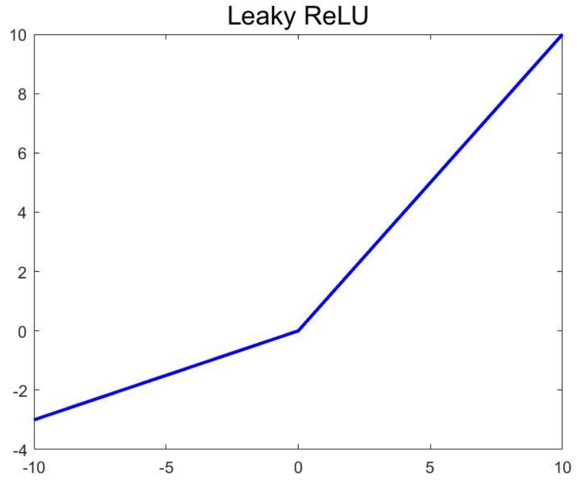
3.BP 算法

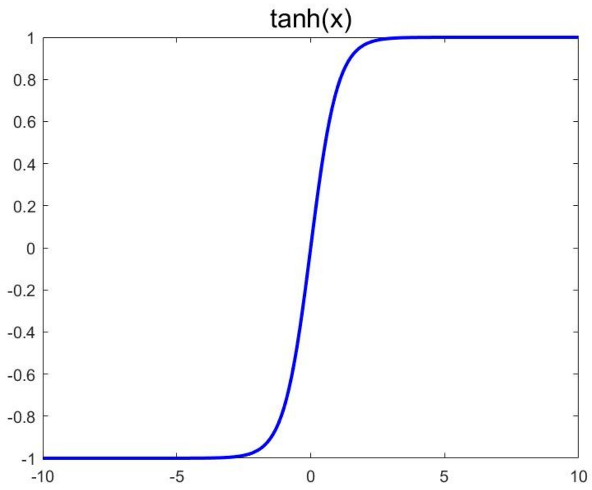
激活函数

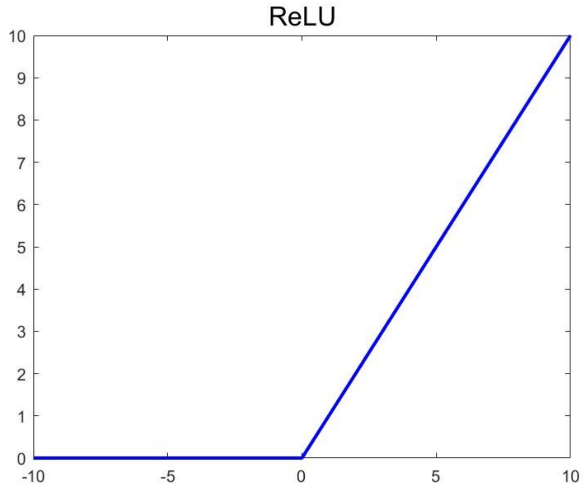
常见激活函数选择：

sigmoid 函数
tanh 函数
ReLU 函数
Leaky ReLU 函数









3.BP 算法

最常用 Sigmoid 函数的优缺点：

优点：

1. 函数处处连续，便于求导
2. 可将函数值的范围压缩至 [0,1] ，可用于压缩数据，且幅度不变
3. 便于前向传输

缺点：

1. 在趋向无穷的地方，函数值变化很小，容易出现梯度消失，不利于深层神经的反馈传输
2. 幂函数的梯度计算复杂
3. 收敛速度比较慢

3.BP 算法

主要步骤

第一步，对样本明确预测输出值与损失函数

第二步，明确参数调整策略

第三步，计算输出层阈值的梯度

第四步，计算隐层到输出层连接权值的梯度

第五步，计算隐层阈值的梯度

第六步，计算输入层到隐层连接权值的梯度

第七步，引出归纳结论

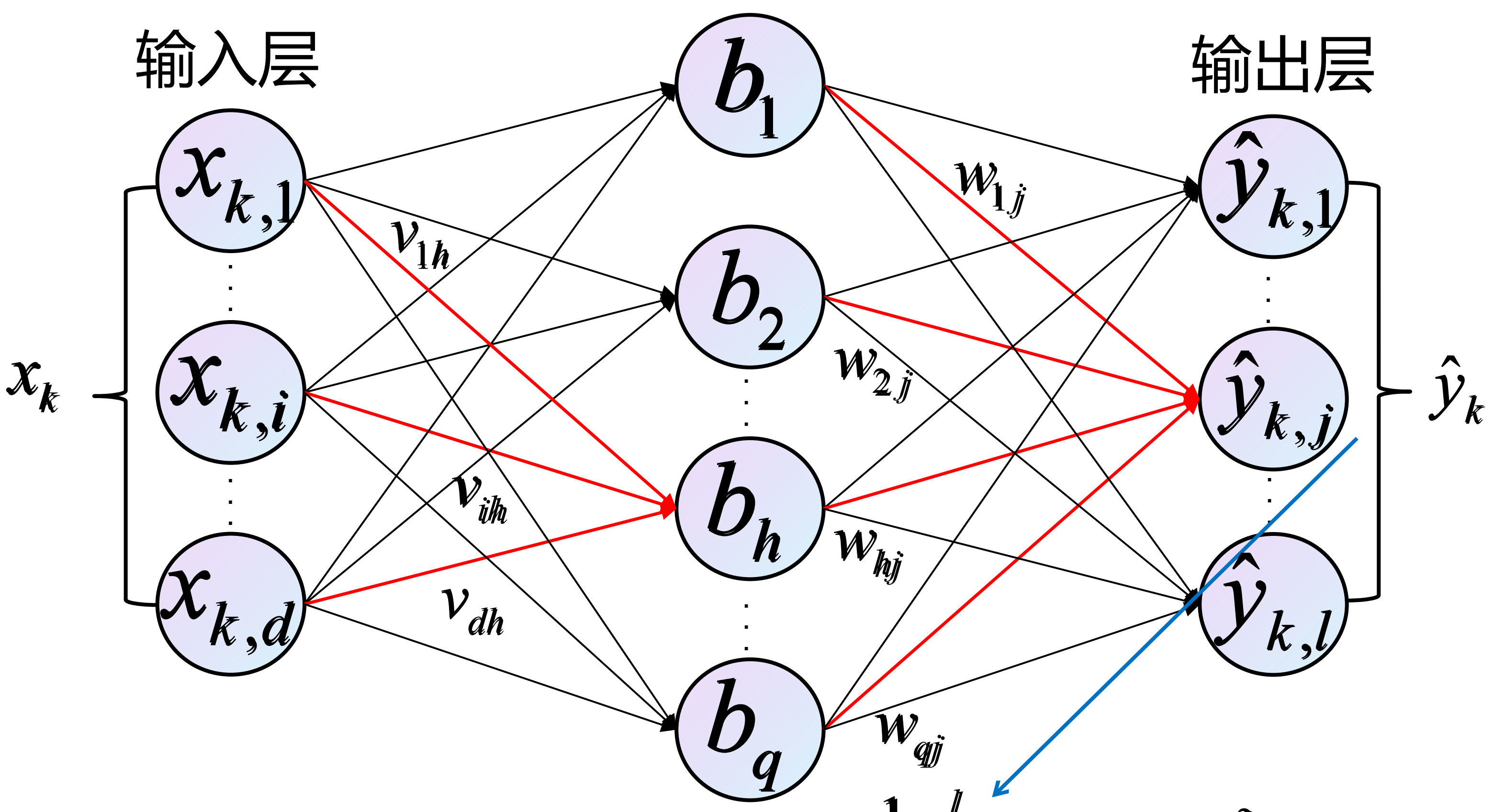
3.BP 算法

第一步，明确损失函数

对样本，神经网络的预测输出值为。

全网络在样本上的均方误差

$$E_k = \frac{1}{2}\sum_{j=1}^{l}\left(\hat{y}_j^k - y_j^k\right)^2$$

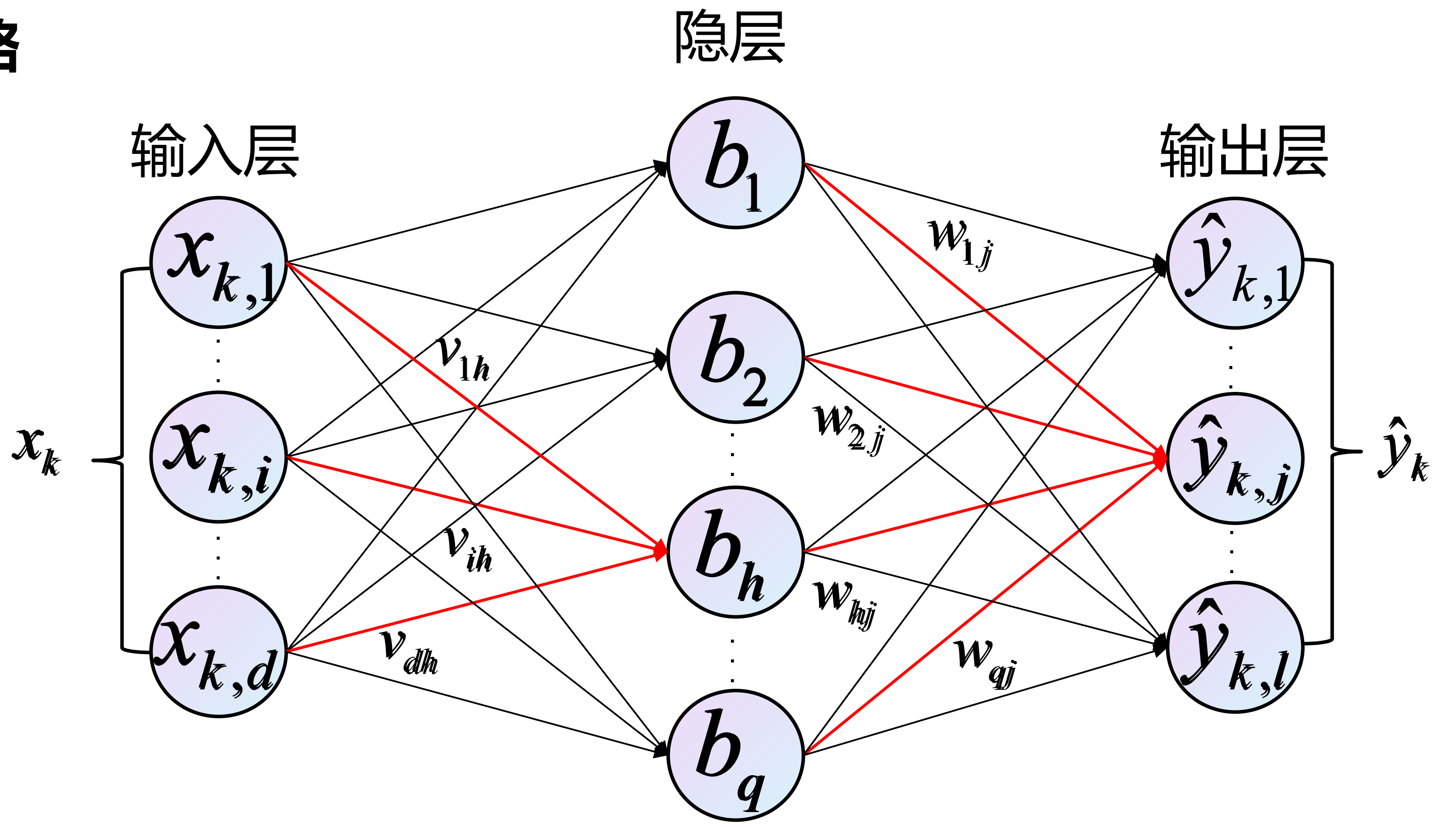


预测均方误差 $E_k = \frac{1}{2}\sum_{j=1}^{l}\left(\hat{y}_j^k - y_j^k\right)^2$

3.BP 算法

第二步，明确参数调整策略

基于梯度下降（ Gradient Descent ）策略，以目标的负梯度方向对参数进行调整



3.BP 算法

第三步，计算输出层阈值的梯度

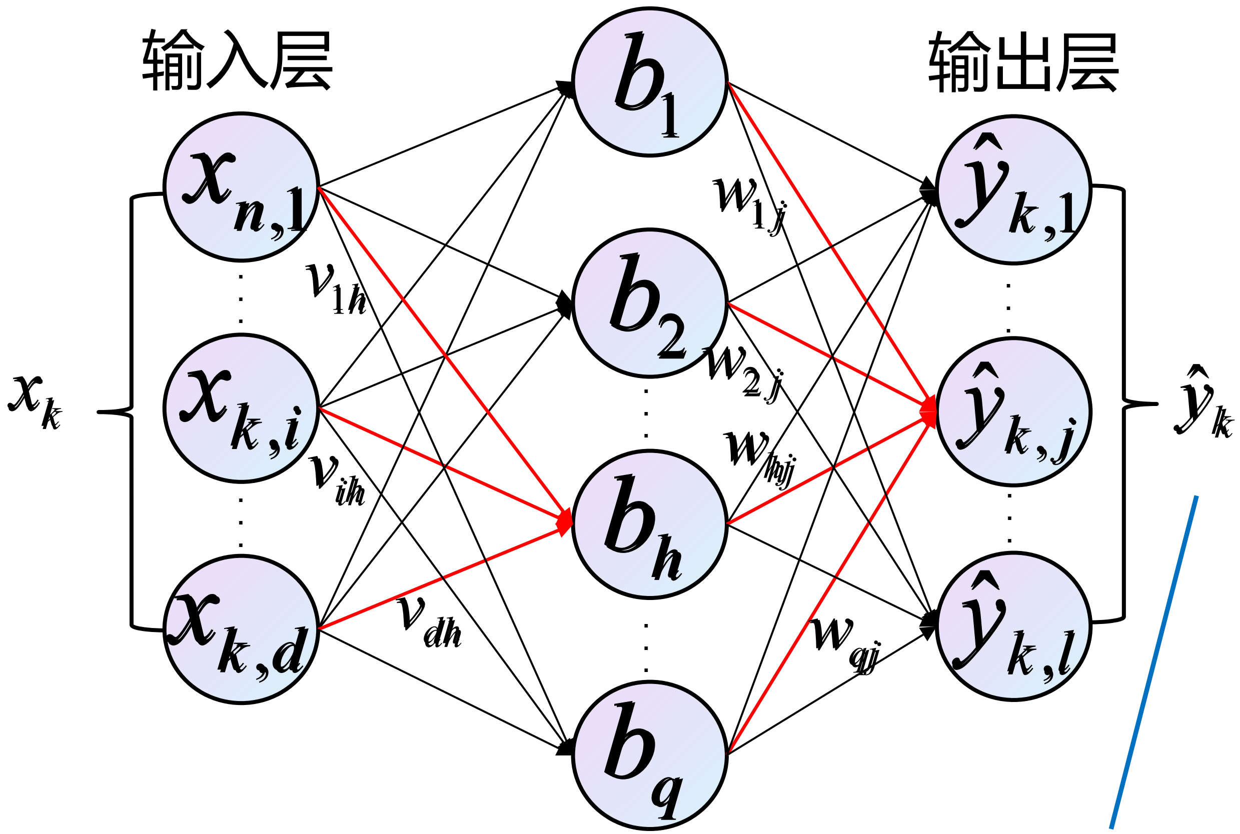
利用链式法则，可得

$$\frac{\partial E_k}{\partial \theta_j} = \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \theta_j}$$

其中，$\frac{\partial E_k}{\partial \hat{y}_j^k} = \hat{y}_j^k - y_j^k$　　$\frac{\partial \hat{y}_j^k}{\partial \theta_j} = -\hat{y}_j^k\left(1-\hat{y}_j^k\right)$

所以，$g_j = \frac{\partial E_k}{\partial \theta_j} = \hat{y}_j^k\left(1-\hat{y}_j^k\right)\left(y_j^k - \hat{y}_j^k\right)$

更新公式　$\theta_j \leftarrow \theta_j - \eta g_j$



对阈值求导 $\frac{\partial E_k}{\partial \theta_j}$

3.BP 算法

第四步，计算隐层到输出层连接权值的梯度

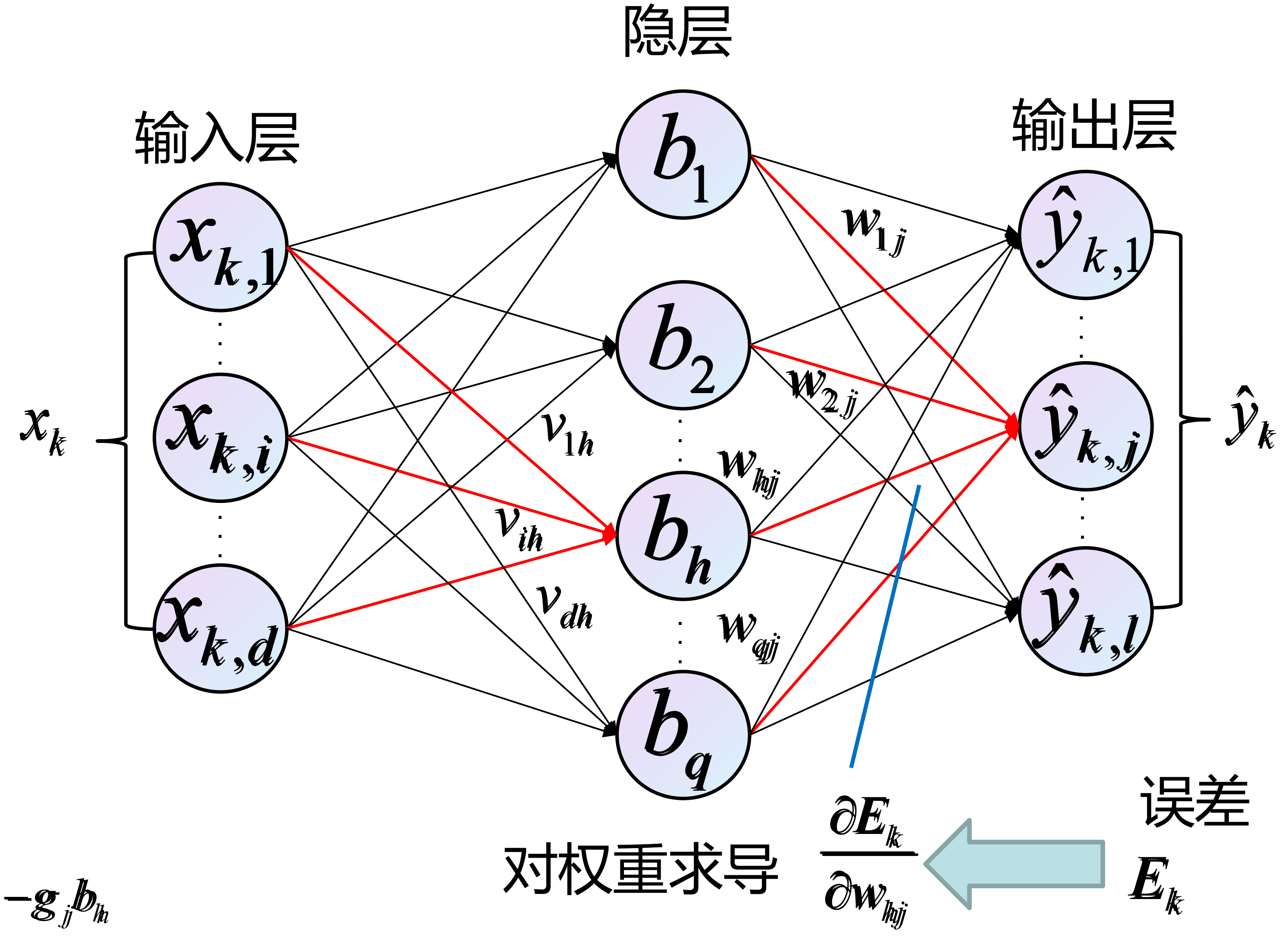
利用链式法则，可得

$$\frac{\partial E_k}{\partial w_{hj}}=\frac{\partial E_k}{\partial \hat{y}_j^k}\cdot\frac{\partial \hat{y}_j^k}{\partial \beta_j}\cdot\frac{\partial \beta_j}{\partial w_{hj}}$$

其中，$\frac{\partial E_k}{\partial \hat{y}_j^k}=\hat{y}_j^k-y_j^k$ $\quad \frac{\partial \hat{y}_j^k}{\partial \beta_j}=\hat{y}_j^k\left(1-\hat{y}_j^k\right)$

可得 $\frac{\partial \beta_j}{\partial w_{hj}}=b_h$

综上可得 $\frac{\partial E_k}{\partial w_{hj}}=\hat{y}_j^k\cdot\left(\hat{y}_j^k-y_j^k\right)\cdot\left(1-\hat{y}_j^k\right)\cdot b_h=-g_j b_h$



3.BP 算法

第五步，计算隐层阈值的梯度

利用链式法则，可得

$$\frac{\partial E_k}{\partial \gamma_h}=\frac{\partial E_k}{\partial b_h}\cdot\frac{\partial b_h}{\partial \gamma_h}$$

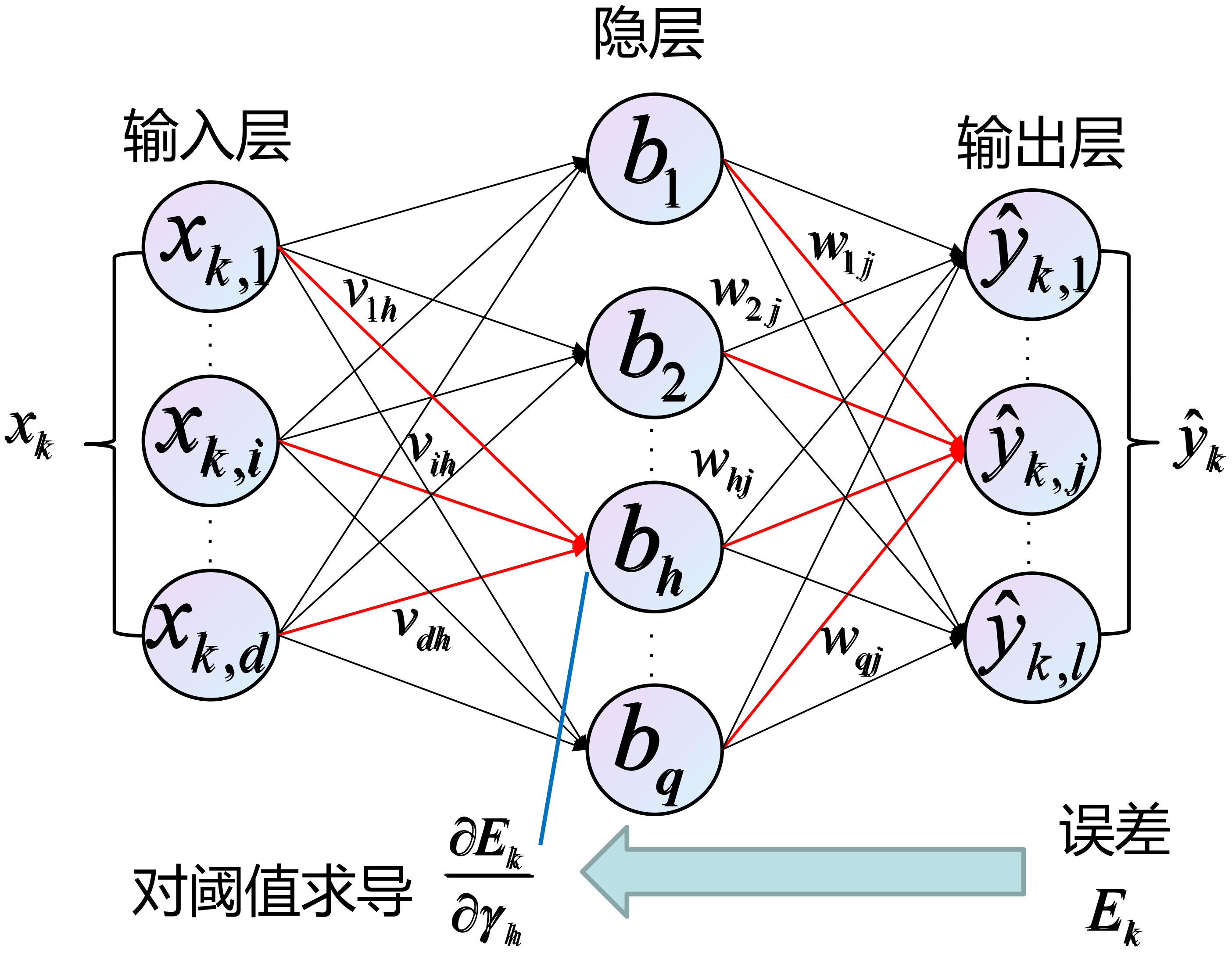
其中，$\frac{\partial E_k}{\partial b_h}=\sum_{j=1}^{l}\frac{\partial E_k}{\partial \hat{y}_j^k}\cdot\frac{\partial \hat{y}_j^k}{\partial \beta_j}\cdot\frac{\partial \beta_j}{\partial b_h}=-\sum_{j=1}^{l}g_j w_{hj}$

$$\frac{\partial b_h}{\partial \gamma_h}=\frac{\partial}{\partial \gamma_h}f(\alpha_h-\gamma_h)=-b_h(1-b_h)$$

所以有 $\frac{\partial E_k}{\partial \gamma_h}=b_h(1-b_h)\sum_{j=1}^{l}w_{hj}g_j$

令 $e_h=b_h(1-b_h)\sum_{j=1}^{l}w_{hj}g_j$

更新公式 $\gamma_h \leftarrow \gamma_h-\eta e_h$



3.BP 算法

第六步，计算隐层权重的梯度

利用链式法则，可得

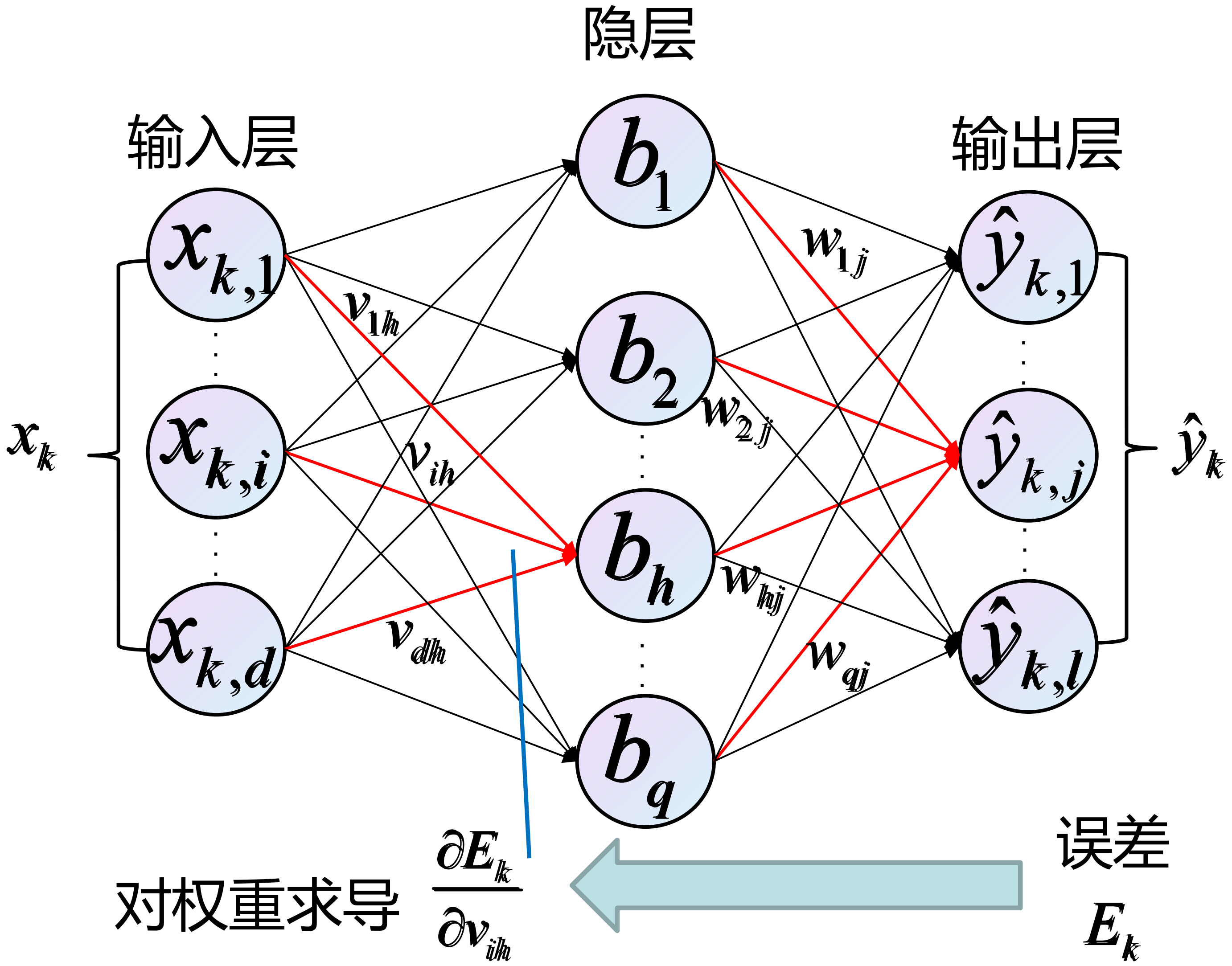
$$\frac{\partial E_k}{\partial v_{ih}}=\frac{\partial E_k}{\partial b_h}\cdot\frac{\partial b_h}{\partial \alpha_h}\cdot\frac{\partial \alpha_h}{\partial v_{ih}}$$

其中，$\frac{\partial E_k}{\partial b_h}=-\sum_{j=1}^{l} g_j w_{hj}$ $\quad \frac{\partial b_h}{\partial \alpha_h}=b_h(1-b_h)$

$$\frac{\partial \alpha_h}{\partial v_{ih}}=x_i$$

所以有 $\frac{\partial E_k}{\partial v_{ih}}=-b_h(1-b_h)x_i\sum_{j=1}^{l} w_{hj}g_j=-e_h x_i$

更新公式 $v_{ih}\leftarrow v_{ih}+\eta e_h x_i$



3.BP 算法

第七步，引出结论

观察，可知

隐层阈值梯度取决于隐层神经元输出、输出层阈值梯度和隐层与输出层的连接权值。

在阈值的调整过程中，当前层的阈值梯度取决于下一层的阈值，这就是 BP 算法的精髓。

观察，可知

当前层的连接权值梯度，取决于当前神经元阈值梯度和上层神经元输出。

3.BP 算法

第七步，引出结论

只要知道上一层神经元的阈值梯度，即可计算当前层神经元阈值梯度和连接权值梯度。

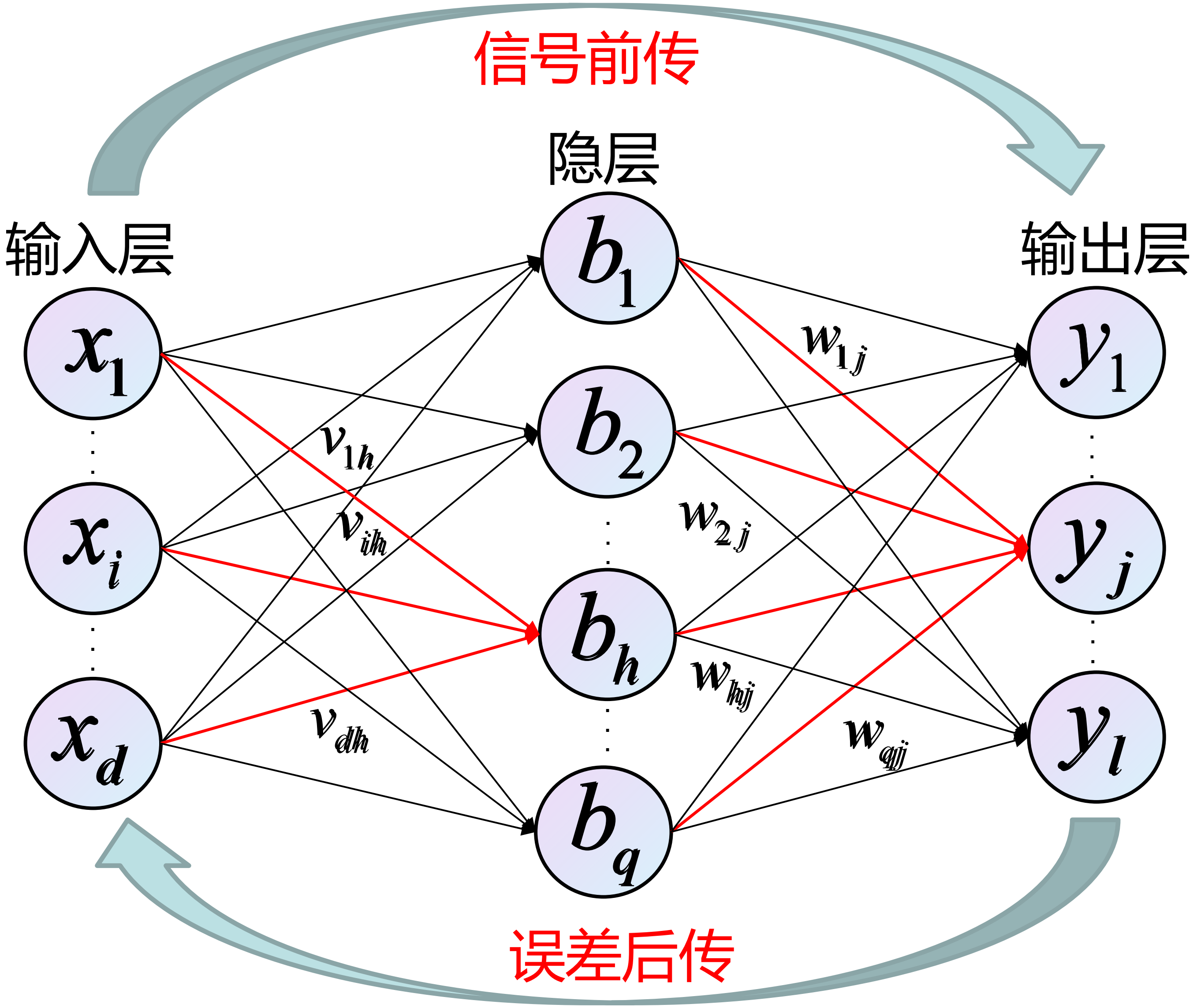
随后可以计算输出层神经元阈值梯度，从而计算出全网络的神经元阈值和连接权值梯度。

最终达到训练网络的目的。

3.BP 算法

算法流程回顾：

1. 将输入样本提供给输入层神经元
2. 逐层将信号前传至隐层、输出层，产生输出层的结果
3. 计算输出层误差
4. 将误差反向传播至隐藏层神经元
5. 根据隐层神经元对连接权重和阈值进行调整
6. 上述过程循环进行，直至达到某些停止条件为止



3.BP 算法

优点：

1. 能够自适应、自主学习。 BP 可以根据预设参数更新规则，通过不断调整神经网络中的参数，已达到最符合期望的输出。
2. 拥有很强的非线性映射能力。
3. 误差的反向传播采用的是成熟的链式法则，推导过程严谨且科学。
4. 算法泛化能力很强。

缺点：

1. BP 神经网络参数众多，每次迭代需要更新较多数量的阈值和权值，故收敛速度比较慢。
2. 网络中隐层含有的节点数目没有明确的准则，需要不断设置节点数字试凑，根据网络误差结果最终确定隐层节点个数
3. BP 算法是一种速度较快的梯度下降算法，容易陷入局部极小值的问题。

参考文献

[1] 《统计学习方法》，清华大学出版社，李航著， 2019 年出版

[2] 《机器学习》，清华大学出版社，周志华著， 2016 年出版

[3] Andrew Ng. Machine Learning[EB/OL]. Stanford University,2014. https://www.coursera.org/course/ml

[4] Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, 2006

[5] Minsky, Marvin and Papert, Seymour. Perceptrons : An Introduction to Computational Geometry. The MIT Press, 1969.

[6] DE Rumelhart, Hinton G E, Williams R J. Learning Representations by Back Propagating Errors[J]. Nature, 1986, 323(6088):533-536.

[7] Bishop C M. Neural Networks for Pattern Recognition[J]. Advances in Computers, 1993, 37:119-166.

[8] Lecun Y, Bengio Y. Convolutional Networks for Images, Speech, and Time-Series[J]. Handbook of Brain Theory & Neural Networks, 1995.