神经网络

神经网络中的最基本的成分是 神经元 。
多个神经元按一定层次连接起来,组成神经网络。

M-P神经元模型:

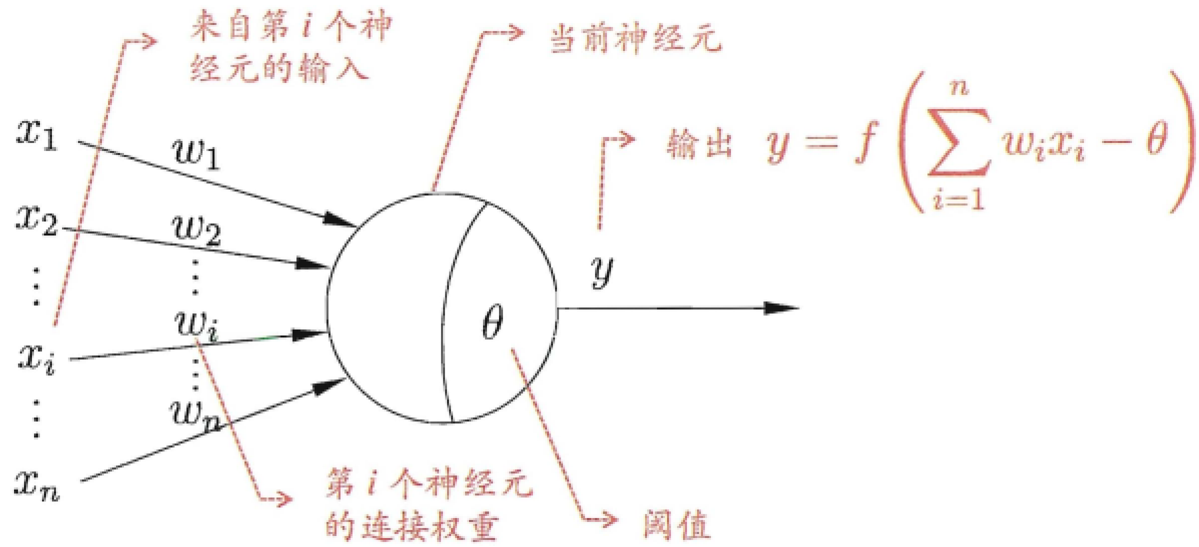
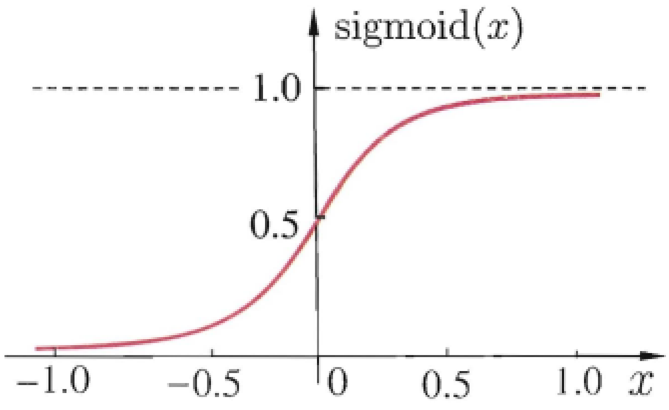


图 5.1 M-P 神经元模型

神经元接受来自n个其他神经元传递过来的输入信号,这些输入信号通过带权重的连接(connection)进行传递,神经元接受到的总输入值将与神经元的阈值进行比较,然后通过 激活函数 处理以产生神经元的输出。

常用 **Sigmod** 函数作为神经元的激活函数。



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

(b) Sigmoid 函数

感知机 Perceptron

感知机由两层神经元组成,接收层接受外界输入信号后传递给输出层,输出层是 M-P神经元。

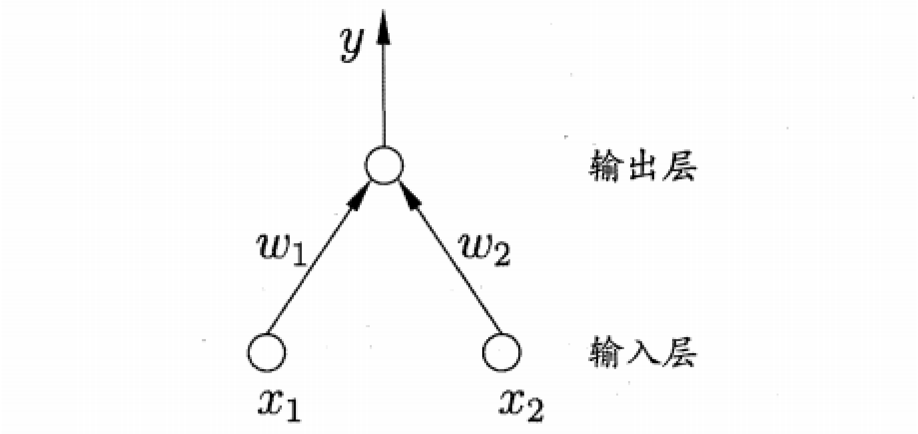


图 5.3 两个输入神经元的感知机网络结构示意图

误差逆传播算法 BP算法

给定如图的神经网络:

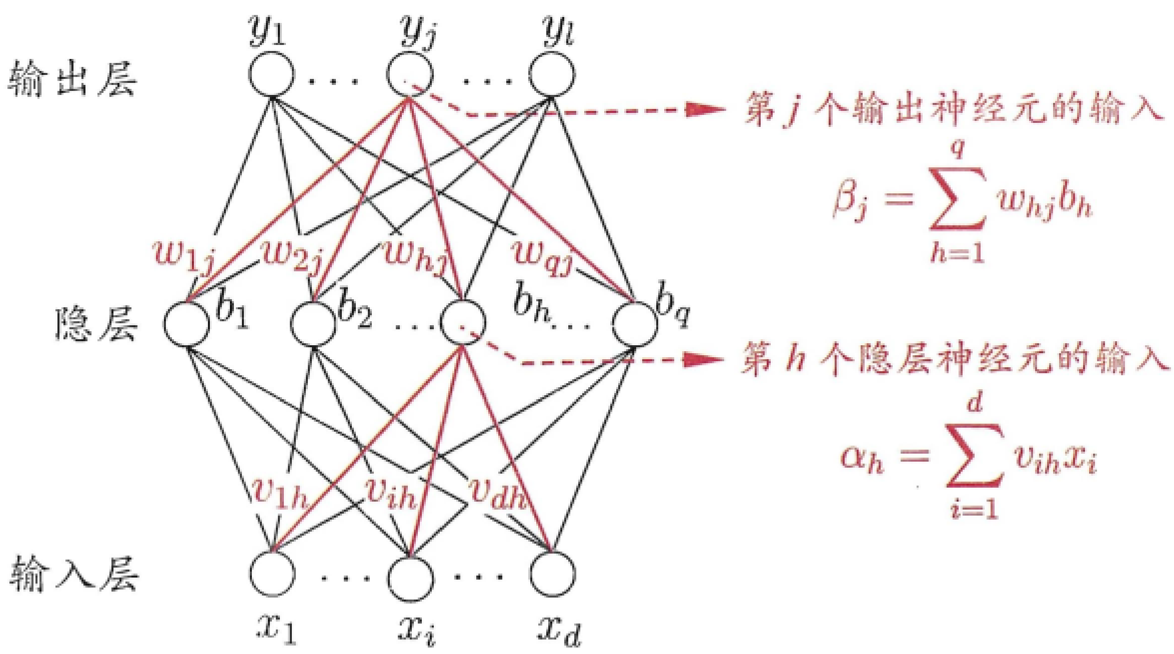


图 5.7 BP 网络及算法中的变量符号

输入层有d个神经元

隐层维有q个神经元

激活函数为$f(x) = sigmoid(x) = \frac{1}{1+e^{-x}}$

第i个输入层神经元和第h个隐层神经元的连接权为v_{ih}

第h个隐层神经元的输入:$\alpha_h = \sum_{i=1}^{d} v_{ih} x_i$

第h个隐层神经元的阈值为γ_h

则隐层神经元的输出为:$b_h = f(\alpha_h - \gamma_h)$

隐层第h个神经元和输出层第j个神经元的连接权为:w_{hj}

第j个输出层神经元的输入为:$\beta_j = \sum_{h=1}^{q} w_{hj} b_h$

第j个输出层神经元的阈值为:θ_j

则输出层第j个神经元的输出为:$\hat{y}_j = f(\beta_j - \theta_j)$

定义一个样本为(x,y),$x \in \mathrm{R}^d, y \in \mathrm{R}^l$

我们使用均方误差作为损失函数,$\hat{y}_j$为输出层第j个神经元的输出结果,则

$$E = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j - y_j)^2$$

这里的$\frac{1}{2}$是为了后面求导方便。

我们的目标就是最小化E,

使用梯度下降算法,即$v \leftarrow v + \Delta v$

> sigmoid函数有一个很好的性质:$f'(x) = f(x)\big(1 - f(x)\big)$

1. 求Δw_{hj}

   $$\Delta w_{hj} = -\eta \frac{\partial E}{\partial w_{hj}} = \eta g_j b_h$$

$$\begin{aligned}
\frac{\partial E}{\partial w_{hj}} &= \frac{\partial E}{\partial \hat{y}_j}\frac{\partial \hat{y}_j}{\partial \beta_j}\frac{\partial \beta_j}{\partial w_{hj}} && (1)\\
&= \hat{y}_j(1-\hat{y}_j)(y_j-\hat{y}_j)b_h && (2)\\
&= -g_j b_h && (3)
\end{aligned}$$

(1):链式法则
(2):展开导数

> $$\frac{\partial E}{\partial \hat{y}} = (\hat{y} - y)$$
> $$\frac{\partial f(x)}{\partial x} = f(x)\big(1 - f(x)\big)$$
> $$\frac{\partial \beta_j}{\partial w_{hj}} = b_h$$

(3):定义符号g_j

$$\begin{aligned}
g_j &= -\frac{\partial E}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial \beta_j}\\
&= -(\hat{y}_j - y_j)f'(\beta_j - \theta_j)\\
&= \hat{y}_j(1-\hat{y}_j)(y_j-\hat{y}_j)
\end{aligned}$$

则$\Delta w_{hj} = \eta g_j b_n$

2. 求$\Delta \theta_j$

   $$\begin{aligned}
   \frac{\partial E}{\partial \theta_j} &= \frac{\partial E}{\partial \hat{y}_j}\frac{\partial \hat{y}_j}{\partial \theta_j}\\
   &= g_j
   \end{aligned}$$

则$\Delta \theta_j = -\eta g_j$

3. 求Δv_{ih}

$$
\begin{aligned}
\frac{\partial E}{\partial v_{ih}} &= \sum_{j=1}^{l} \frac{\partial E}{\partial \hat{y}_j} \frac{\partial \hat{y}_j}{\partial \beta_j} \frac{\partial \beta_j}{\partial b_h} \frac{\partial b_h}{\partial \alpha_h} \frac{\partial \alpha_h}{\partial v_{ih}} \\
&= \sum_{j=1}^{l} -g_j w_{hj} b_h (1-b_h) x_i \\
&= -x_i b_h (1-b_h) \sum_{j=1}^{l} -g_j w_{hj} \\
&= -x_i e_h
\end{aligned}
$$

$$e_h = b_h(1-b_h)\sum_{j=1}^{l} -g_j w_{hj} = \frac{\partial E}{\partial b_h}\frac{\partial b_h}{\partial \alpha_h}$$

则$\Delta v_{ih} = \eta e_h x_i$

4. 求$\Delta \gamma_h$

$$\frac{\partial E}{\partial \gamma_h} = \sum_{j=1}^{l} \frac{\partial E}{\partial \hat{y}_j} \frac{\partial \hat{y}_j}{\partial \beta_j} \frac{\partial \beta_j}{\partial b_h} \frac{\partial b_h}{\partial \gamma_h} = e_h$$

则$\Delta \gamma_h = -\eta e_h$

算法过程为:

输入: 训练集 $D = \{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^{m}$;
学习率 η.

过程:

1: 在(0, 1)范围内随机初始化网络中所有连接权和阈值
2: **repeat**
3:     **for all** $(\boldsymbol{x}_k, \boldsymbol{y}_k) \in D$ **do**
4:         根据当前参数和式(5.3) 计算当前样本的输出 $\hat{\boldsymbol{y}}_k$;
5:         根据式(5.10) 计算输出层神经元的梯度项 g_j;
6:         根据式(5.15) 计算隐层神经元的梯度项 e_h;
7:         根据式(5.11)-(5.14) 更新连接权 w_{hj}, v_{ih} 与阈值 θ_j, γ_h
8:     **end for**
9: **until** 达到停止条件

输出: 连接权与阈值确定的多层前馈神经网络

图 5.8 误差逆传播算法

第4行为正向传播,正向激活神经元
第4第5步为误差反向传播
第6步根据反向传播回来的误差进行参数的更新

由隐层神经元的梯度e_h的表达式可以看出e_h 和输出层神经元的梯度g_j有关,那么进行计算的过程,可以看成是把输出层的误差,反向传播回到了隐层。

自编码器

自编码器也是神经网络的一种,经过训练后尝试将输入复制到输出。
"复制"过程可以分解为"编码"和"解码"两个过程,希望经过这两个过程之后,提取出输入最显著的特征作为输出。

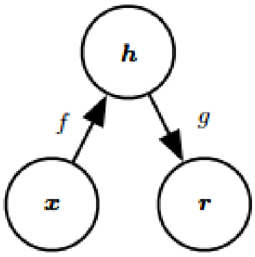


图 14.1: 自编码器的一般结构,通过内部表示或编码 $\boldsymbol{h}$ 将输入 $\boldsymbol{x}$ 映射到输出(称为重构)$\boldsymbol{r}$。自编码器具有两个组件:编码器 f(将 $\boldsymbol{x}$ 映射到 $\boldsymbol{h}$)和解码器 g(将 $\boldsymbol{h}$ 映射到 $\boldsymbol{r}$)。

从代价函数的角度来看,我们的目标就是最小化一下这个代价函数:
$L(x, g\big(f(x)\big))$

一个特殊的情况是,把输入完全复制到了输出而不没有任何的变化。

这显然是无用,因此我们需要添加一些约束条件来使得输出r只能近似等于x,而不是完全等于x。

欠完备自编码器

欠完备自编码器的思想是 限制隐层神经元的数目,使得中间结果h的维度小于样本的维度。这样就会丢失部分信息,而使得输出r不能完全等于x

正则自编码器

若要求隐层的容量(h的维度)大于输入x,若不加以约束,更大可能的学习结果是完全复制,因此在代价函数中,还需要加入一个正则项加以约束。

稀疏自编码器