

BP 神经网络的结构

神经网络的网络结构由输入层，隐含层，输出层组成。隐含层的个数+输出层的个数=神经网络的层数，也就是说神经网络的层数不包括输入层。下面是一个三层的神经网络，包含了两层隐含层，一个输出层。其中第一层隐含层的节点数为 3，第二层的节点数为 2，输出层的节点数为 1；输入层为样本的两个特征 x_1, x_2 。

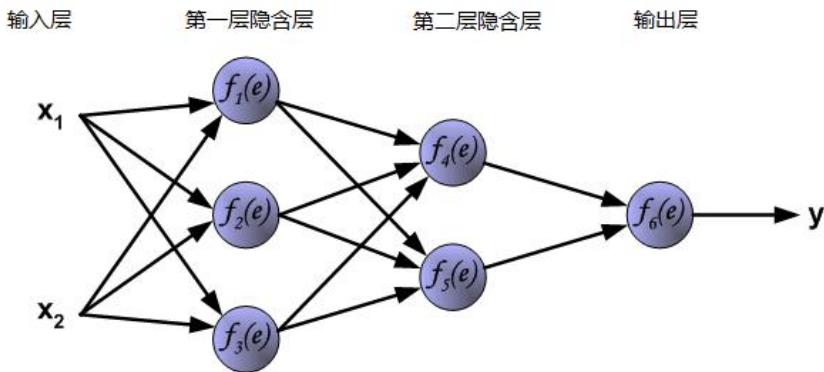


图 1 三层神经网络

在神经网络中每一个节点的都与上一层的所有节点相连，称为全连接。神经网络的上一层输出的数据是下一层的输入数据。在图中的神经网络中，原始的输入数据，通过第一层隐含层的计算得出的输出数据，会传到第二层隐含层。而第二层的输出，又会作为输出层的输入数据。

神经网络中的每一层（除了输入层）都是由神经元组成，也称为节点。每一个神经元都相当于一个感知器。如下图：

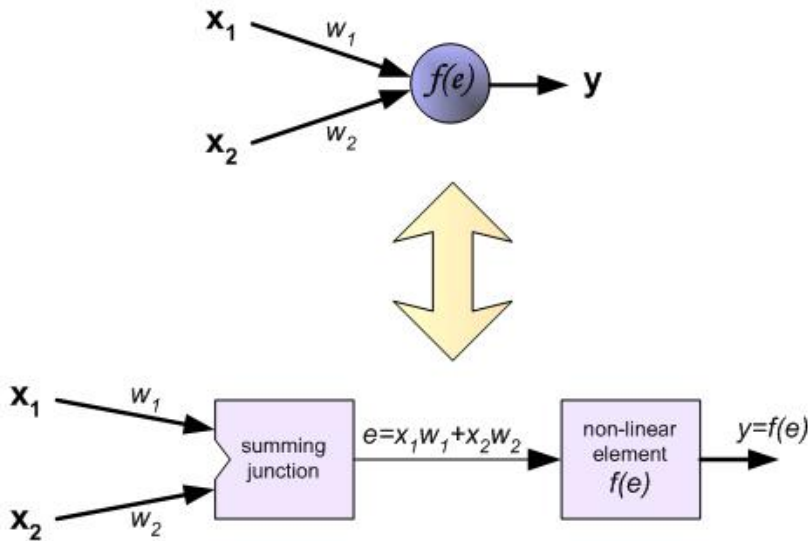


图 2 单个神经元

在神经网络中，每个节点都将计算出特征矩阵 X 与权值矩阵的加权和，得到净输入 e ，然后通过激励函数 $f(e)$ 得到该节点的输出 y 。在图 1 中，每条连线都可以看做是一个权值。

在神经网络中，可以添加输出层节点的个数来解决多分类问题。有四个类别需要分类则，则输出层的节点个数可以设为 4 个节点，每一个节点代表一个类别。

BP 神经网络的训练过程

神经网络的训练过程分为两个过程：1、向前传播得到预测数据；2、反向传播更新权重。如下图所示：

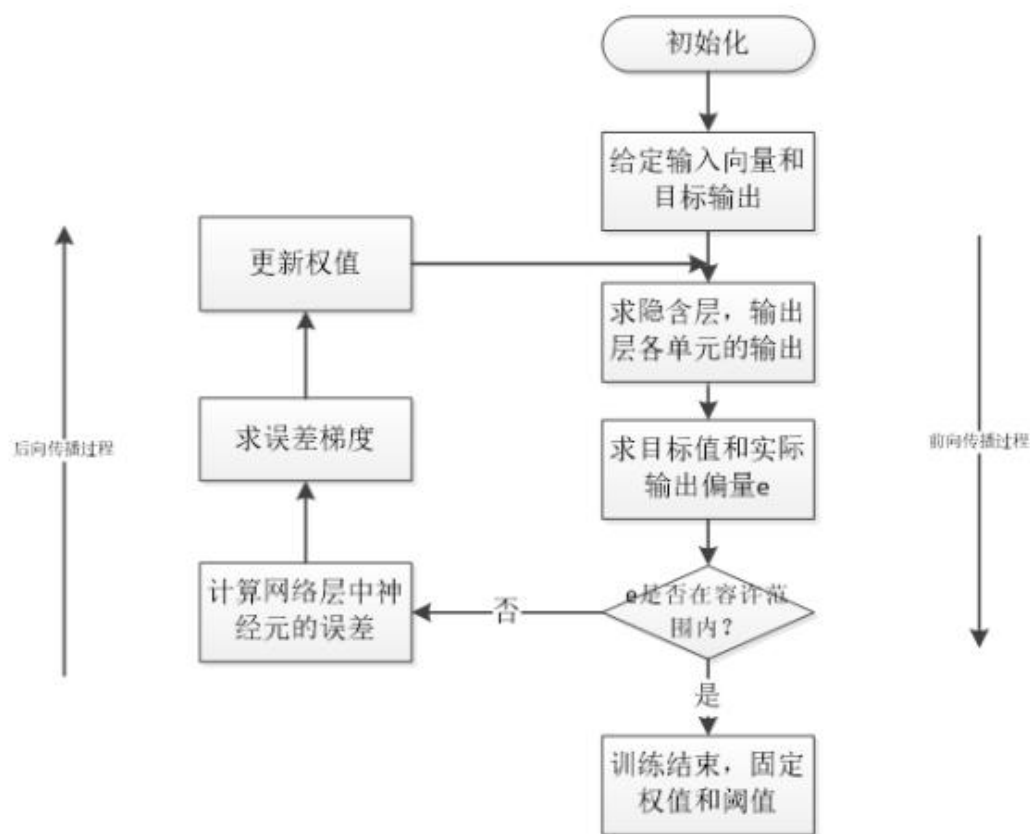
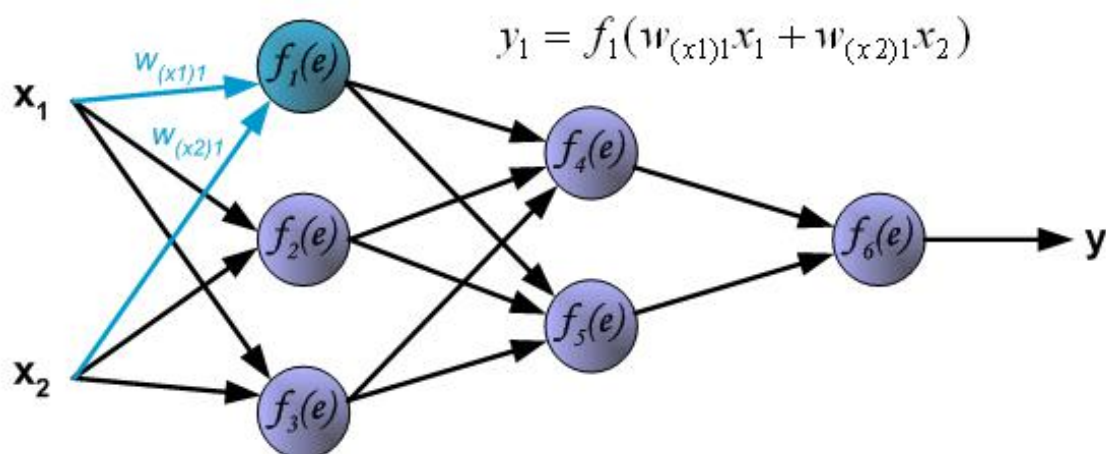


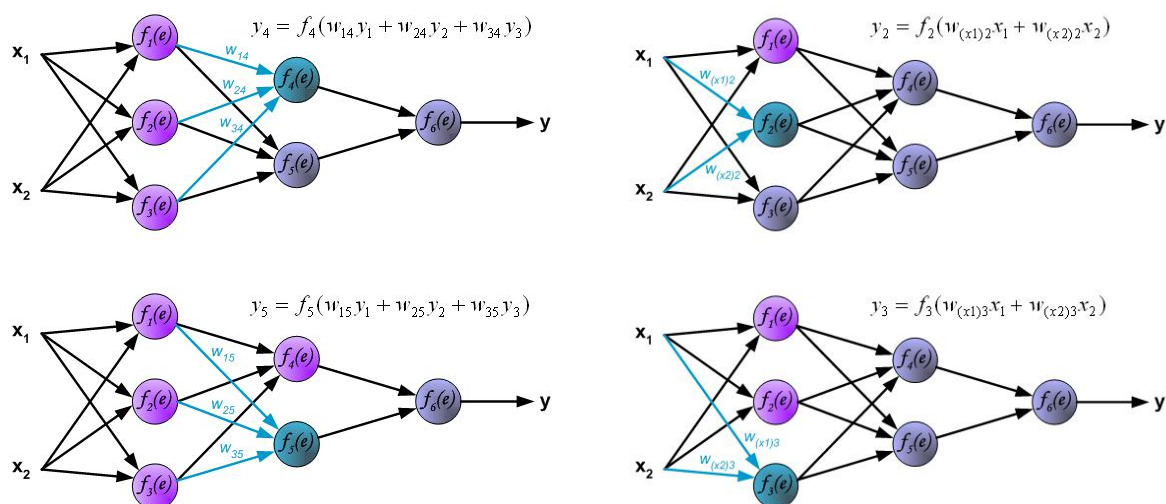
图 3 神经网络的训练过程

第一步、向前传播得到预测数据：向前传播的过程,即数据从输入层输入，经过隐含层，输出层的计算得到预测值，预测值为输出层的输出结果。网络层的输出即，该层中所有节点（神经元）的输出值的集合。我们以图一的神经网络结构为例，分析向前传播过程。

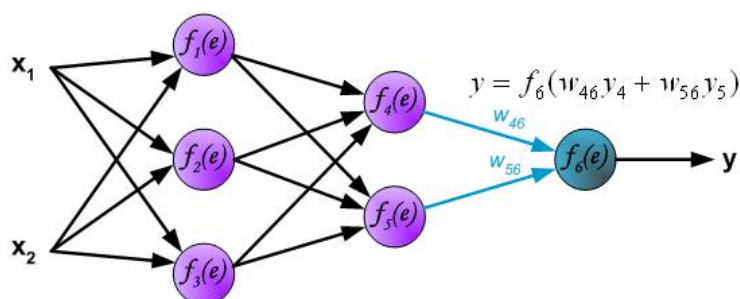
1. 得到隐含层的输出 y_1, y_2, y_3 ：



2. 获取到第二层的隐含层输出 y_4, y_5 ，输入的数据也就是第一层隐含层的输出数据 y_1, y_2, y_3 。

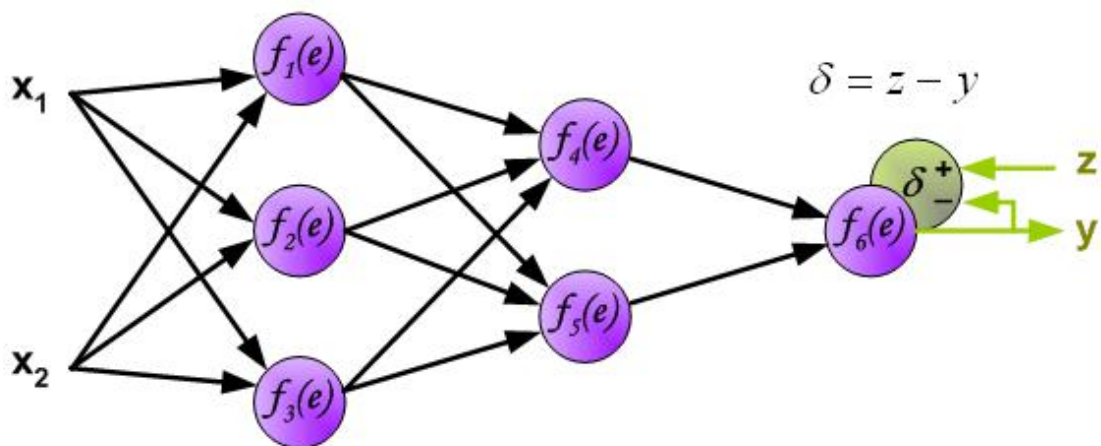


3、通过输出层，得到最后的预测值 y 。

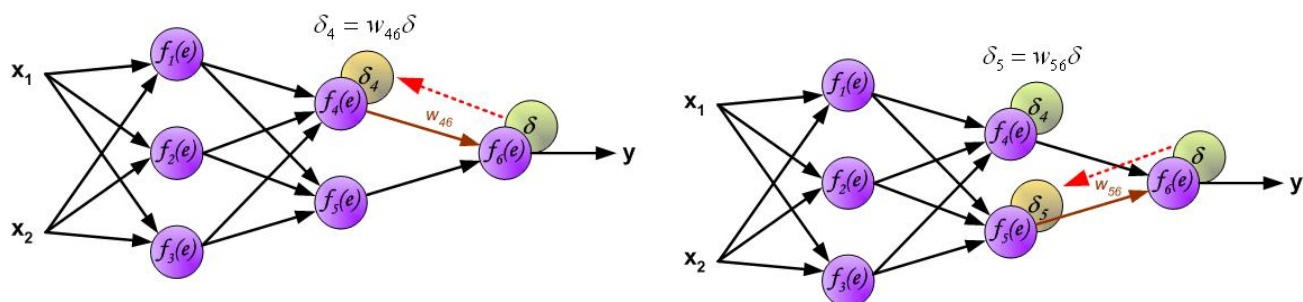


第二步、反向传播更新权重：根据样本的真实类标，计算模型预测的结果与真实类标的误差。然后将该误差反向传播到各个隐含层。计算出各层的误差，再根据各层的误差，更新权重。

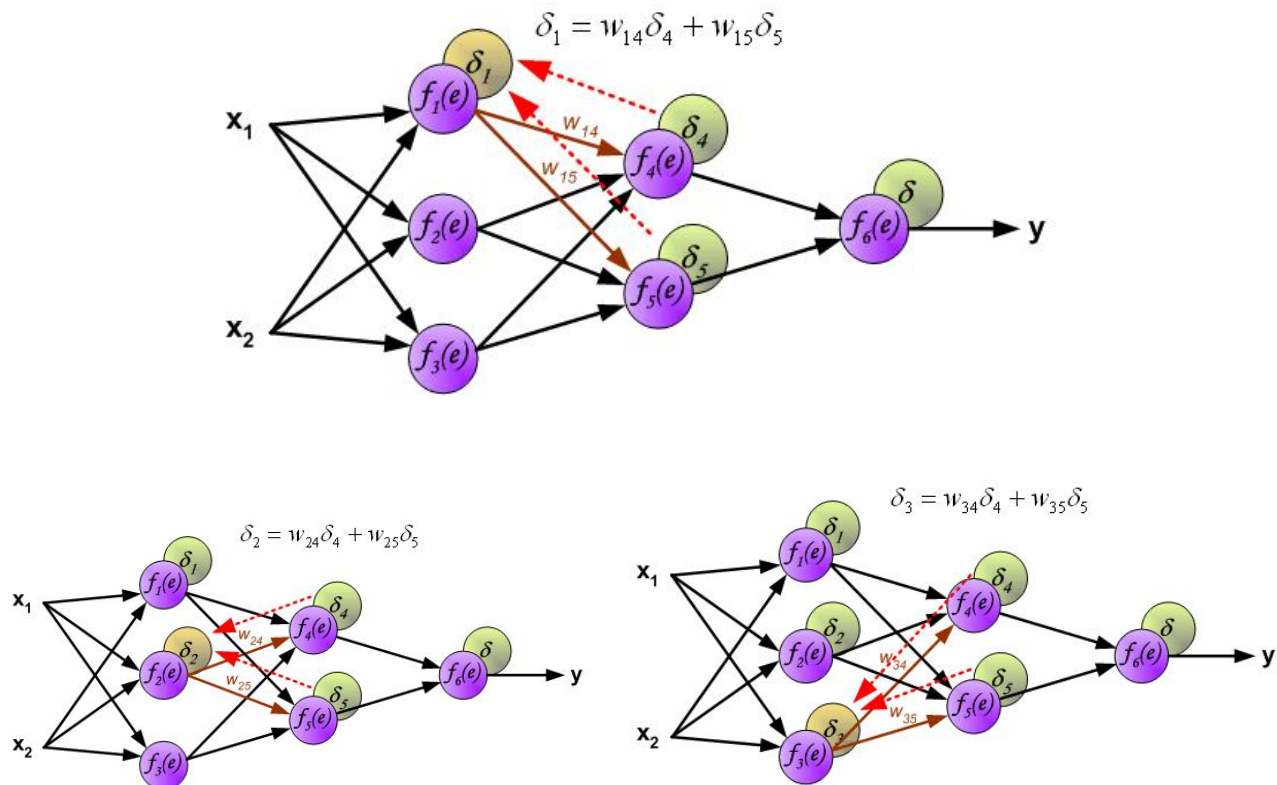
1. 计算输出层的误差：其中 z 为该样本的类标



2 计算第二层隐含层的误差

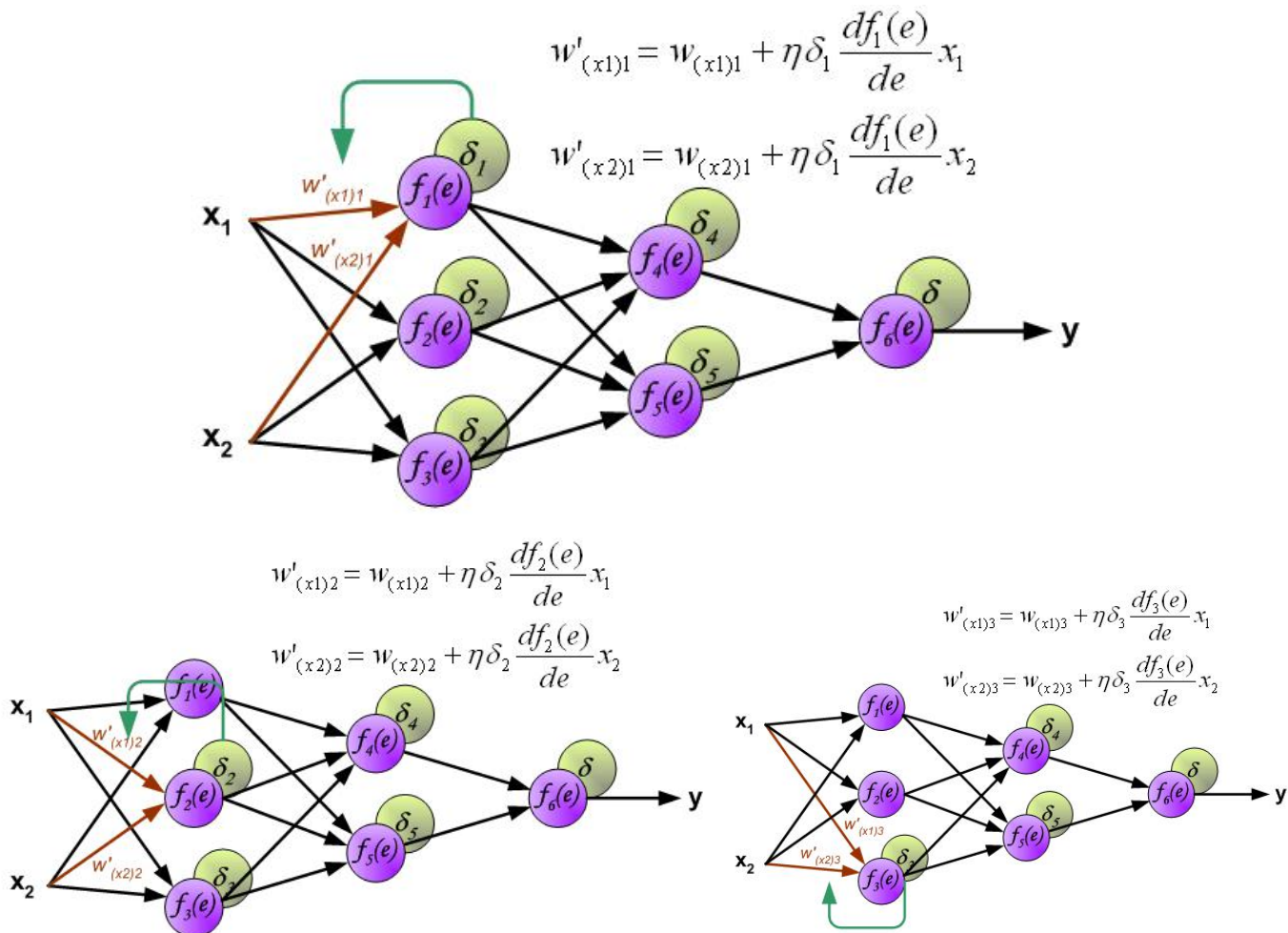


3. 计算第一次隐含层的误差：

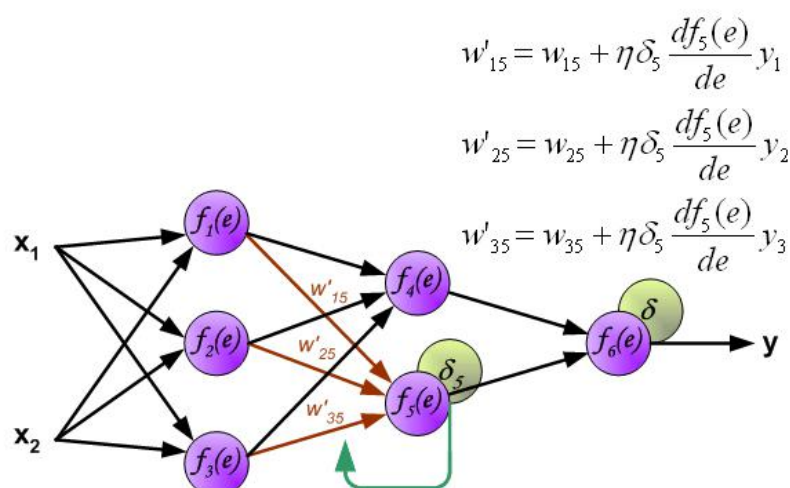
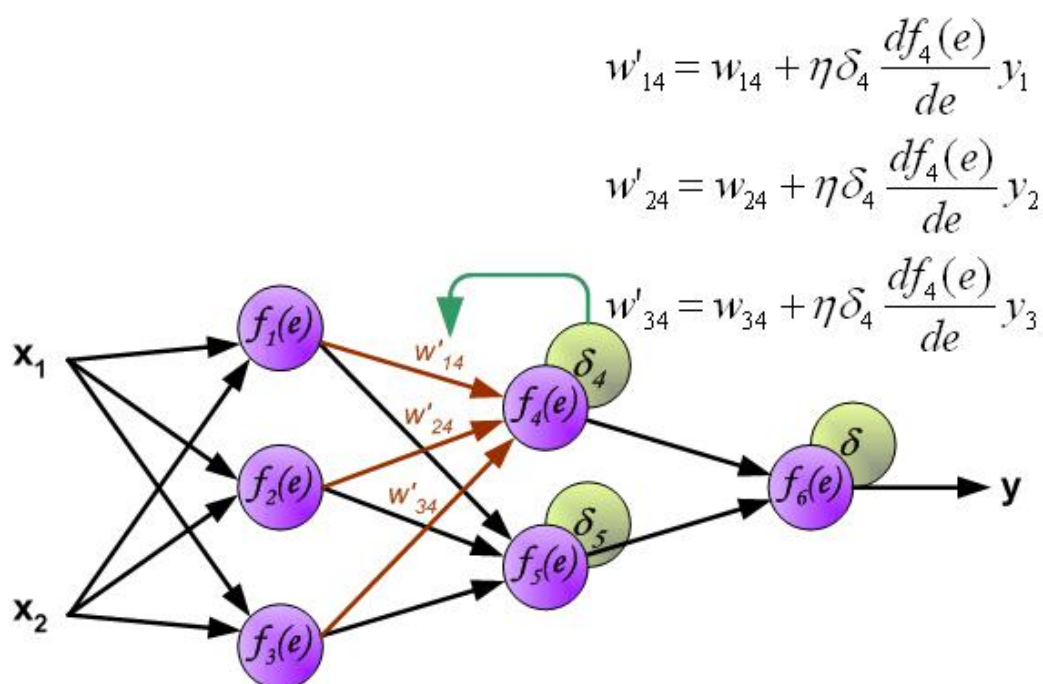


4、更新权重：新的权值=原权值+学习速率×该节点的误差×激励函数的导函数的值（f'(e)的倒数）
×与该节点相连的输入值

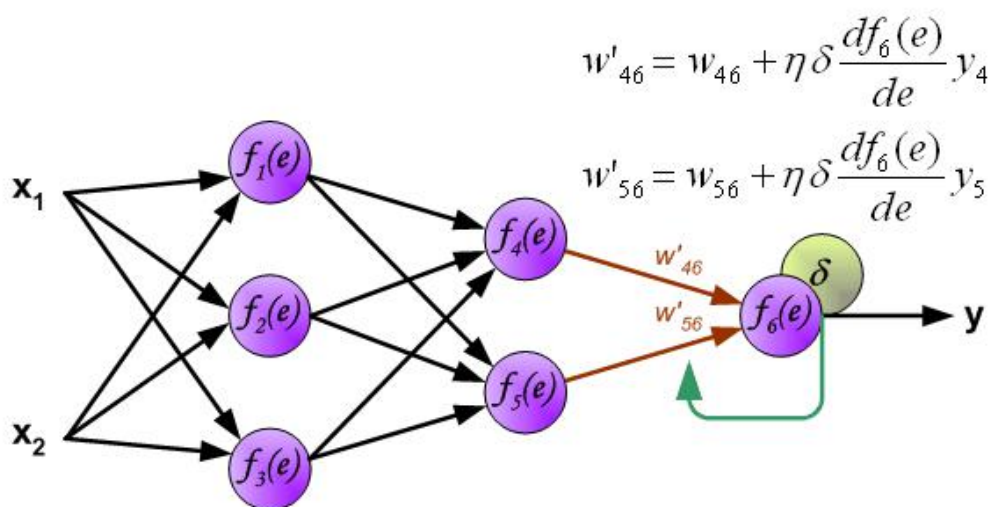
4.1 更新输入层与第一层隐含层之间的权值：



4.2 更新第一层隐含层与第二层隐含层之间的权值



4.3 更新第二层隐含层与输出层之间的权值


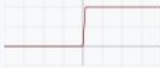


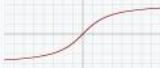








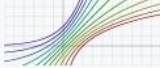





以上就是反向传播的过程。误差从输出层反向的传到输入层，然后再从输入层向前更新权值。

激活函数：

A sigmoid (S-shaped) function: $y(x) = f(x) = \frac{1}{1 + e^{-x}}$

A hyperbolic tangent transfer function: $y(x) = f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

Name	Plot	Equation	Derivative (with respect to x)	Range
Identity		$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$	$\{0, 1\}$
Logistic (a.k.a. Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	$(0, 1)$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$	$(-1, 1)$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$	$\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$
Softsign [7][8]		$f(x) = \frac{x}{1 + x }$	$f'(x) = \frac{1}{(1 + x)^2}$	$(-1, 1)$
Rectified linear unit (ReLU) [9]		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$[0, \infty)$
Leaky rectified linear unit (Leaky ReLU) [10]		$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0.01 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$(-\infty, \infty)$
Parameteric rectified linear unit (PReLU) [11]		$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$(-\infty, \infty)$
Randomized leaky rectified linear unit (RReLU) [12]		$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ [1]	$f'(\alpha, x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$(-\infty, \infty)$
Exponential linear unit (ELU) [13]		$f(\alpha, x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} f(\alpha, x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$(-\alpha, \infty)$
SoftPlus [17]		$f(x) = \ln(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$	$(0, \infty)$
Bent identity		$f(x) = \frac{\sqrt{x^2 + 1} - 1}{2} + x$	$f'(x) = \frac{x}{2\sqrt{x^2 + 1}} + 1$	$(-\infty, \infty)$
SoftExponential [18]		$f(\alpha, x) = \begin{cases} -\frac{\ln(1 - \alpha(x + \alpha))}{\alpha} & \text{for } \alpha < 0 \\ x & \text{for } \alpha = 0 \\ \frac{e^{\alpha x} - 1}{\alpha} + \alpha & \text{for } \alpha > 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} \frac{1}{1 - \alpha(x + \alpha)} & \text{for } \alpha < 0 \\ e^{\alpha x} & \text{for } \alpha \geq 0 \end{cases}$	$(-\infty, \infty)$
Sinusoid [19]		$f(x) = \sin(x)$	$f'(x) = \cos(x)$	$[-1, 1]$
Sinc		$f(x) = \begin{cases} 1 & \text{for } x = 0 \\ \frac{\sin(x)}{x} & \text{for } x \neq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x = 0 \\ \frac{\cos(x)}{x} - \frac{\sin(x)}{x^2} & \text{for } x \neq 0 \end{cases}$	$[\approx -2.17234, 1]$
Gaussian		$f(x) = e^{-x^2}$	$f'(x) = -2xe^{-x^2}$	$(0, 1]$

MP 模型

目前人们提出的神经元模型有很多，其中最早提出切影响最大的，是 1943 年提出的 MP 模型。指出了神经元的形式化数学描述和网络结构方法，证明了单个神经元能执行逻辑功能，从而开创了人工智能神经网络研究时代。

概念：把神经元视为二值开关元件，按不同方式组合可以完成各种逻辑运算，这种逻辑神经元模型被称为 MP 模型。

能够构成逻辑与、或、非，就可进而组成任意复杂的逻辑关系，因此，MP 模型是按一定方式组织起来，可以构成具有逻辑功能的神经网络。

MP 模型是最简单的网络，但是由于生物神经元本质上是模拟过程，过早地把物理量抽象为 0 和 1，会丢失许多有用信息，因此神经计算应当将模拟的和数字的技术结合起来。从最简化的观点看，仍具有一定指导意义 MP 模型应用

MP 模型应用：可用于实现分类、模式识别等，当前已经有许多成功的基于 M-P 神经元模型的神经网络得到应用，如 BP 算法，这种算法是实现人脸识别的主要算法之一。

感知器模型

1957 年，美国心理学家罗森布拉特（Frank Rosenblatt）提出一种具有单层计算单元的神经网络，成为 Perceptron, 即为感知器。

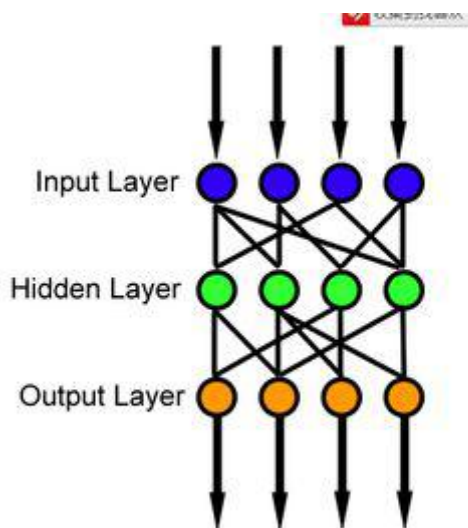
感知器是一种前馈网络，同层内无互连，不同层间无反馈，由下层向上层传递。其输入、输出均为离散值，神经元对输入加权求和后，由阈值函数决定其输出。

单层感知器的结构与功能都非常简单，但却是要就其他网络的基础。

由于在感知器中第一次引入了学习的概念，使人脑所具备的学习功能在基于符号处理的数学模型中得到了一定程度的模拟，所以引起了广泛的关注。

简单感知器模型实际上仍然是 M-P 模型的结构，但是它通过采用有监督学习来逐步增强模式划分的能力，达到学习的目的

前馈神经网络(feedforward neural network)



感知器网络

感知器（又叫感知机）是最简单的前馈网络，它主要用于模式分类，也可用在基于模式分类的学习控制和多模态控制中。感知器网络可分为单层感知器网络

感知器（MLP, Multilayer Perceptron）是一种前馈人工神经网络模型，其将输入的多个数据集映射到单一的输出的数据集上，可以解决任何线性不可分问题。和多层感知器网络。

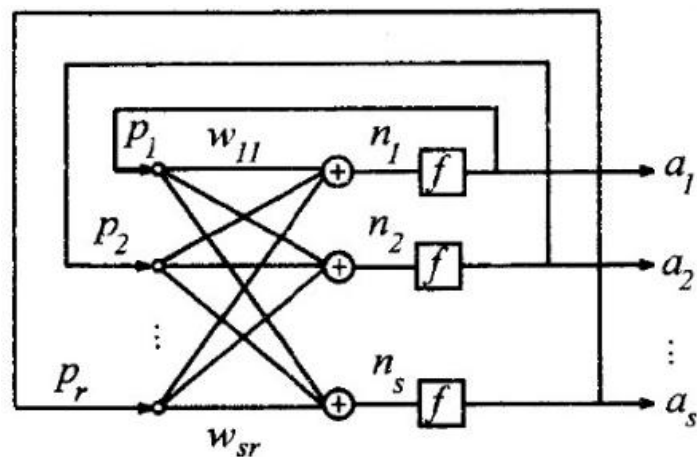
BP 网络

BP 网络是指连接权调整采用了反向传播 (Back Propagation) 学习算法的前馈网络。与感知器不同之处在于, BP 网络的神经元变换函数采用了 S 形函数 (Sigmoid 函数), 因此输出量是 0~1 之间的连续量, 可实现从输入到输出的任意的非线性映射。

RBF 网络

RBF 网络是指隐含层神经元由 RBF 神经元组成的前馈网络。RBF 神经元是指神经元的变换函数为 RBF (Radial Basis Function, 径向基函数) 的神经元。典型的 RBF 网络由三层组成: 一个输入层, 一个或多个由 RBF 神经元组成的 RBF 层 (隐含层), 一个由线性神经元组成的输出层。

反馈神经网络(Recurrent Network)

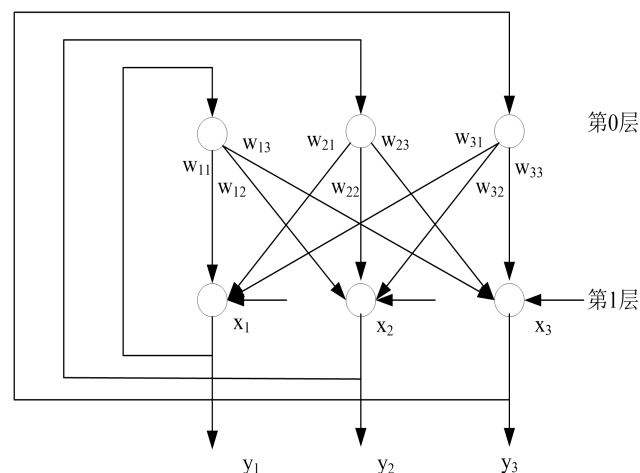


反馈神经网络, 又称自联想记忆网络, 其目的是为了设计一个网络, 储存一组平衡点, 使得当给网络一组初始值时, 网络通过自行运行而最终收敛到这个设计的平衡点上。反馈神经网络是一种将输出经过一步时移再接入到输入层的神经网络系统。

反馈网络能够表现出非线性动力学系统的动态特性。它所具有的主要特性为以下两点:

- (1) . 网络系统具有若干个稳定状态。当网络从某一初始状态开始运动, 网络系统总可以收敛到某一个稳定的平衡状态;
- (2) . 系统稳定的平衡状态可以通过设计网络的权值而被存储到网络中。

反馈网络是一种动态网络, 它需要工作一段时间才能达到稳定。该网络主要用于联想记忆和优化计算。在这种网络中, 每个神经元同时将自身的输出信号作为输入信号反馈给其他神经元, 它需要工作一段时间才能达到稳定。



Hopfield 网络是神经网络发展历史上的一个重要的里程碑。由美国加州理工学院物理学家 J. J. Hopfield 教授于 1982 年提出，是一种单层反馈神经网络。Hopfield 神经网络是反馈网络中最简单且应用广泛的模型，它具有联想记忆的功能。

Hopfield 神经网络模型是一种循环神经网络，从输出到输入有反馈连接。在输入的激励下，会产生不断的状态变化。

反馈网络有稳定的，也有不稳定的，如何判别其稳定性也是需要确定的。对于一个 Hopfield 网络来说，关键在于确定它在稳定条件下的权系数。

图中，第 0 层是输入，不是神经元；第二层是神经元。

1984 年，Hopfield 设计并研制了网络模型的电路，并成功地解决了旅行商 (TSP) 计算难题 (快速寻优问题)。

根据网络的输出是离散量或是连续量，Hopfield 网络也分为离散和连续的两种。

Hopfield 神经网络有两种：离散 Hopfield 网络 (DHNN) 和连续 Hopfield 网络 (CHNN)。

1) 离散 Hopfield 网络 (DHNN)：神经元的输出只取 1 和 0，分别表示神经元处于激活和抑制状态。

对于二值神经元，它的计算公式如下
$$u_j = \sum_i w_{ij} y_i + x_j$$

其中， x_i 为外部输入。并且有：
$$\begin{cases} y_i = 1, & \text{当 } u_i \geq 0 \text{ 时} \\ y_i = 0, & \text{当 } u_i < 0 \text{ 时} \end{cases}$$

2) 连续 Hopfield 网络 (CHNN) 拓扑结构和 DHNN 的结构相同。不同之处在于其函数 g 不是阶跃函数，而是 S 形的连续函数。一般取 $G(u) = 1/(1 + e^{-u})$

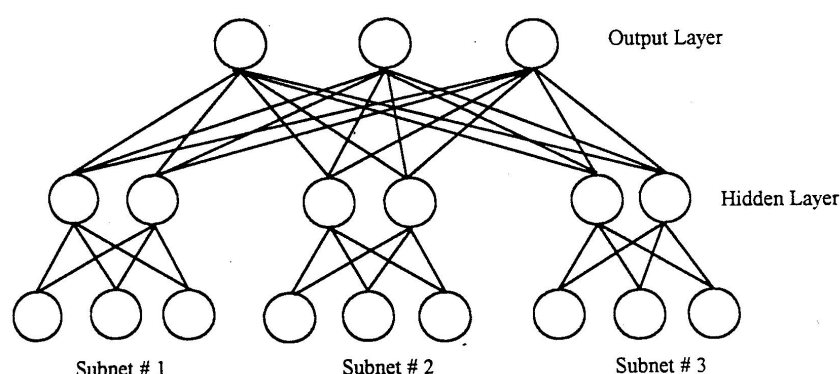
sig

自联想神经网络(autoassociative neural network)

既然自联想神经网络能够实现对输入数据的重构，如果这个网络结构已经训练好了，那么其中间层，就可以看做是对原始输入数据的某种特征表示。如果我们把它的第三层去掉，这样就是一个两层的网络。如果，我们把这个学习到特征再用同样的方法创建一个自联想的三层 BP 网络，如上图所示。换言之，第二次创建的三层自联想网络的输入是上一个网络的中间层的输出。用同样的训练算法，对第二个自联想网络进行学习。那么，第二个自联想网络的中间层是对其输入的某种特征表示。如果我们按照这种方法，依次创建很多这样的由自联想网络组成的网络结构

- 输入层节点数和输出层节点数相关
- 中间层是对前一层数据的抽象化表达
- 可用于降噪和降维

分层神经网络(hierarchical neural network)



- 输入向量被划分为对输出响应有类似影响的组。

优势：

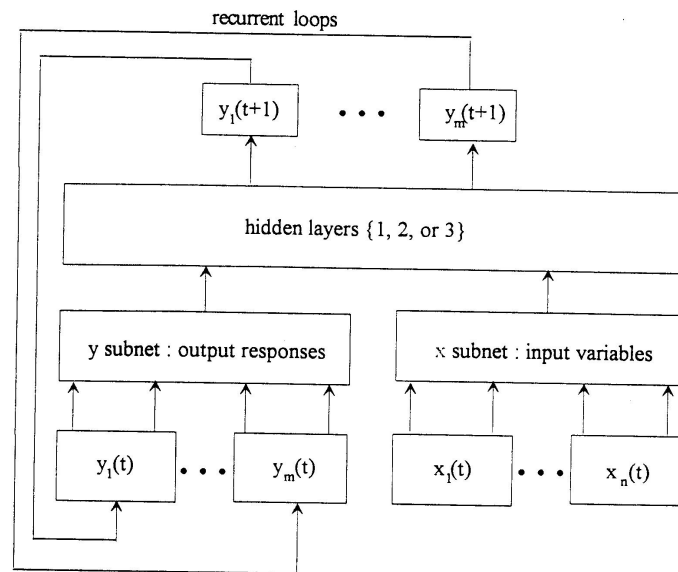
- 多个节点使用较少的权重因子，提高效率。因此，网络培训需要较少的例子和更少的时间。
- 相关变量的定义明确的提示，帮助网络在正确的学习方向。
- 同构于同一任务的专家系统或基于模型的算法。我们可以映射的每个有用的子网的规则（或小套规则）或基于模型的局部分析。

分层神经网络有几个隐藏层被分割成子网，其中输入向量根据它们对输出响应的影响被分成组。两种主要类型：

- *移动窗口网络的时间依赖进程
- *输入 - 压缩网络，用于处理大型输入变量集

移动窗口网络(moving windows network)

循环神经网络(recurrent neural network)

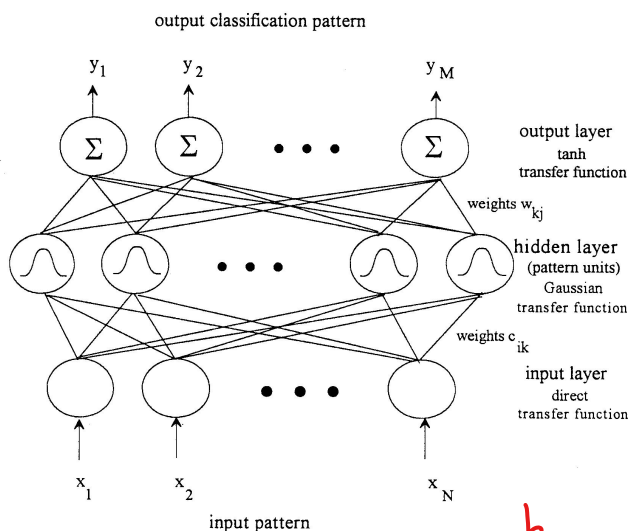


- 循环网络结合了神经网络的反馈和前馈连接。它只是一个将输出响应连接到输入层的循环的神经网络。
- 循环网络提供了一种建模技术，可用于时间相关过程的过程优化和自适应过程控制。

适应启发式评价(adaptive heuristic critic)

reservoir neural network

径向基网络(RBF,radial basis function network)



- 输入层和输出层节点数相同
- 其节点满足径向对称的独特性质
- 输入空间中的中心矢量 c_k ，由集群中心组成，元素 c_{ik} ($i = 1$ 到 N)。矢量通常被存储为从输入层到隐藏层的权重因子。
- 距离度量来确定具有元素 x_i ($i = 1$ 到 N) 的输入向量 x 离中心向量 c_k 有多远。
- 我们使用 x 和 c_k 之间的标准欧式距离度量来定义欧几里德和， I_k ($k = 1$ 到 L)，其中 L 是隐藏层中的节点的数量。

$$I_k = \|x - c_k\|_2 = \sqrt{\sum_{i=1}^N (x_i - c_{ik})^2}$$

- (c) 将欧几里德总和 I_k ($k = 1$ 到 L) 变换成每个节点的输出的传递函数。

$$v_k = e^{-\frac{\|x - c_k\|^2}{2\sigma_k^2}}$$

- 其中 σ_k 是高斯函数的宽度
- 我们首先假设 L 个聚类中心的初始集合，即中心矢量 c_k ，其对应于隐藏层中的 L 个节点。
- c_k 的元素 c_{ik} ($i = 1$ 到 N ; $k = 1$ 到 L) 被存储为输入和隐藏层之间的权重因子。
- 假设有 N 个节点的输入层有 T 个训练样本，并用 $x_i(t)$ ($i = 1$ 到 N ; $t = 1$ 到 T) 的训练向量 $x(t)$ 表示。
- 然后，自适应 K 均值聚类算法迭代地找到一组期望的 L 个中心矢量 c_k ，其将使 T 个训练矢量 $x(t)$ 与它们的最近 L 个中心 c_k ($k = 1$ 到 L) 之间的距离的平方和最小。

input training neural network

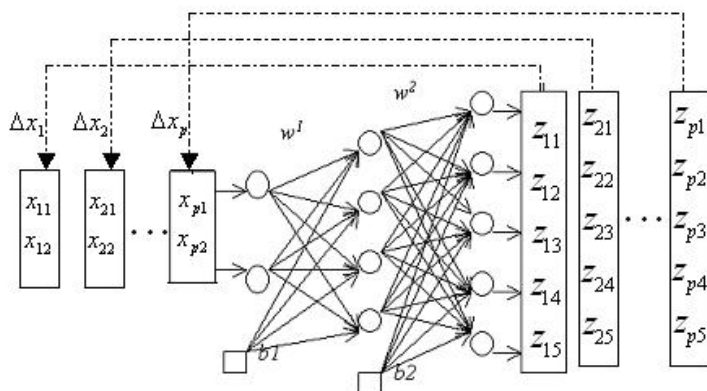


Figure Input Training Neural Network Structure

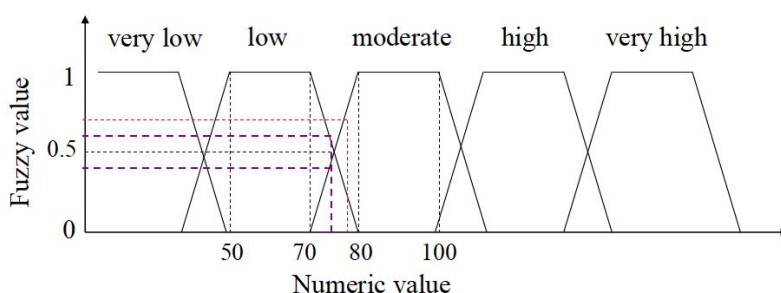
- 令: x_{pk} 是第 p 个样本中第 k 个观测变量的值,
- z_{pk} 是 IT-NN 的相应输出
- 随机定义输入, 产生的输出与输入进行梯度运算

超限学习机网络(extreme learning machine)

组合神经网络(ensemble neural network)

模糊神经网络(Fuzzy Neural Network)

- 神经模糊网络通过神经网络实现模糊逻辑推理。
- 模糊逻辑源于量化基于规则的系统的愿望。 它提供了量化某些量词的方法, 例如大概, 通常, 很少, 几个, 很少, 甚至非常。
- Fuzzy set: 用 0 (不是成员) 和 1 (绝对是成员) 之间的值来量化成员的度数。
- 如何通过模糊化器将数字变量转换为模糊逻辑变量, 并通过解模糊器将模糊逻辑变量转换回数字变量。对于过渡区域中的数值, 我们在区域的开始和结束值之间使用线性插值。



自联想

自联想神经网络 (Auto-Associative Neural Network, 缩写为 AANN) 是 1992 年 Kramer 提出的, 是 BP 神经网络的一种特殊情形。其特点是有对称拓扑结构, 即输出量等于输入量。

异联想

异联想与自联想类似, 不同之处在于输出与输入不等 ($P_i \neq T_i$)。

广义 delta 规则是训练 bp 网络最常用的方法, 是一种最小化均方误差的迭代梯度下降法。这种技术使用动量项来加快训练速度。

在**建造人工神经网络**时, 建设者必须作出许多决定:

- *培训和测试数据的规模
- *规范输入和输出数据集
- *学习算法
- *拓扑
- *要使用的传递函数
- *学习率和动量系数

学习曲线提供了一个可视化的网络性能来回忆和概括的好方法。

自联想网络将输入模式与自身相关联, 并且用于数据压缩和过滤, 以及用于输入向量的降维。

用于时间依赖系统的**循环网络**结合了神经网络的反馈和前馈连接, 提供了一种通过循环使用网络输出响应作为附加输入变量的方法。

RBF 网络隐藏层的内部表示有一个更自然的解释。

神经网络广泛应用于建模, 仿真, 控制, 运行故障识别, 特征分类等。