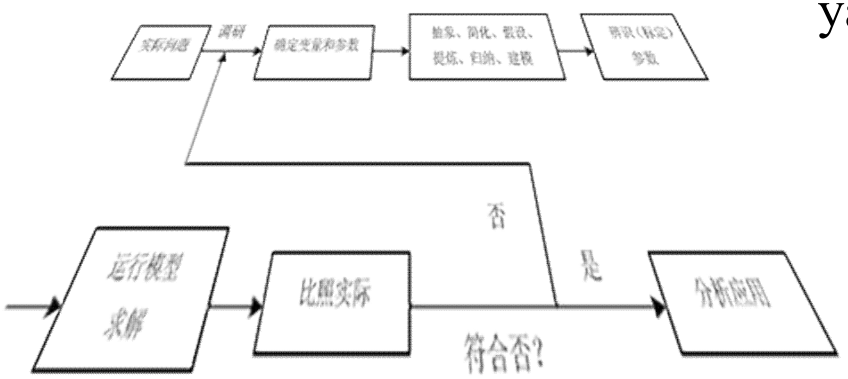
建模与仿真:
人工神经网络方法

姚　远
yaoyuan@shu.edu.cn



快速制造工程中心

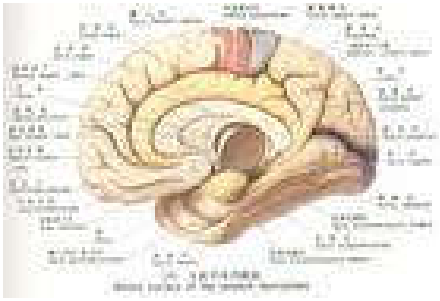
2020/10/28

大纲

- 神经网络概述
- 神经元与网络
- 感知机应用实例
- 学习算法
- 简单扩展

人工神经网络建模

当你盯着这张PPT时，
就正在使用一个约包含10^{11}个
神经元的复杂系统工作

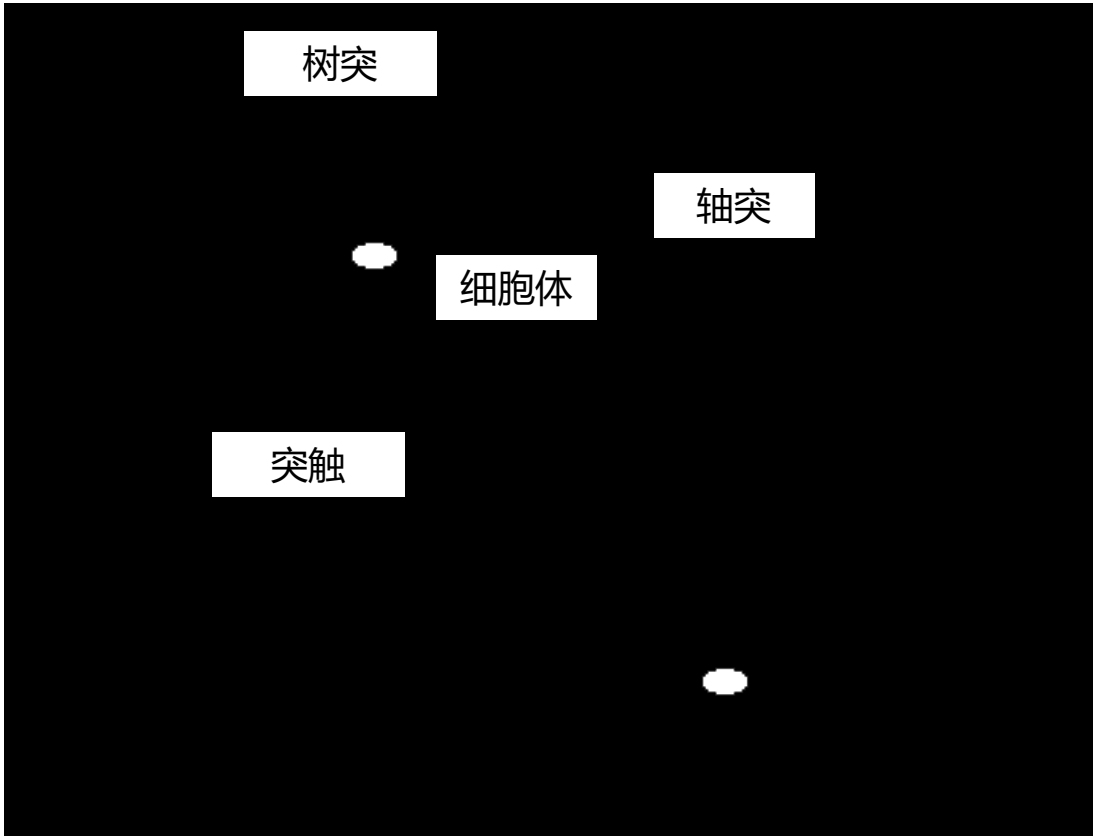


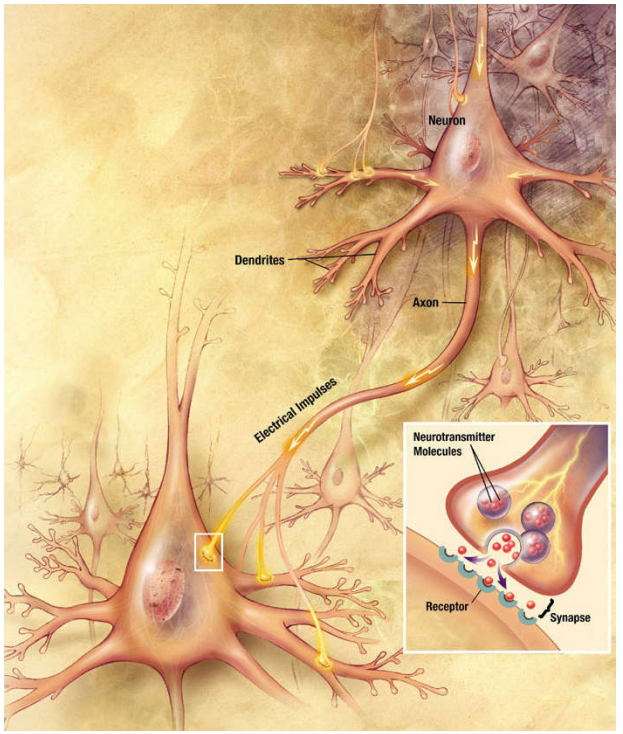
人工神经网络（Artificial Neutral Network，ANN）是对复杂系统建立模型的一种简单方法

- 人工神经网络模拟神经元的连接结构
- 人工神经网络的连接结构决定了功能

生物学的启示

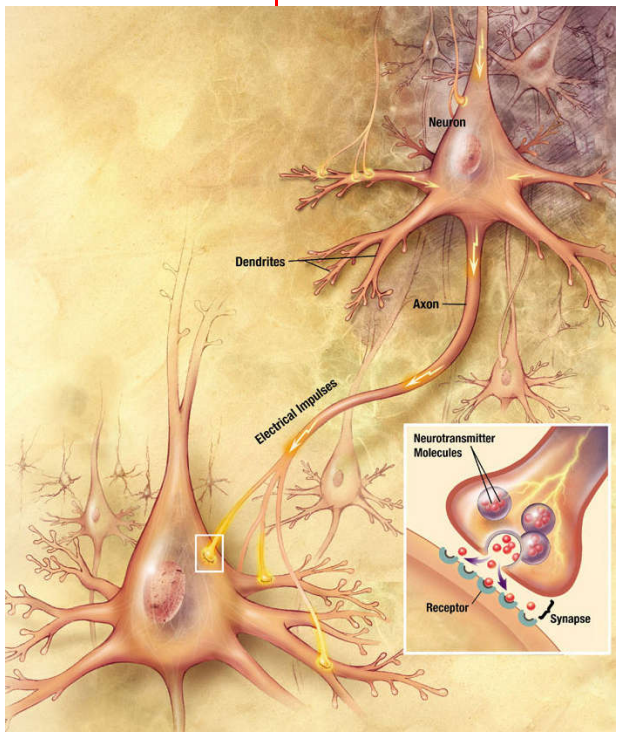
- 神经元反应速度慢
  - 神经元反应级别:10^{-3} s　电子电路反应级别:10^{-9} s
- 人脑使用了大量的并行运算
  - $\approx 10^{11}$ 神经元数目
  - $\approx 10^{4}$ 每个神经元的连接数目





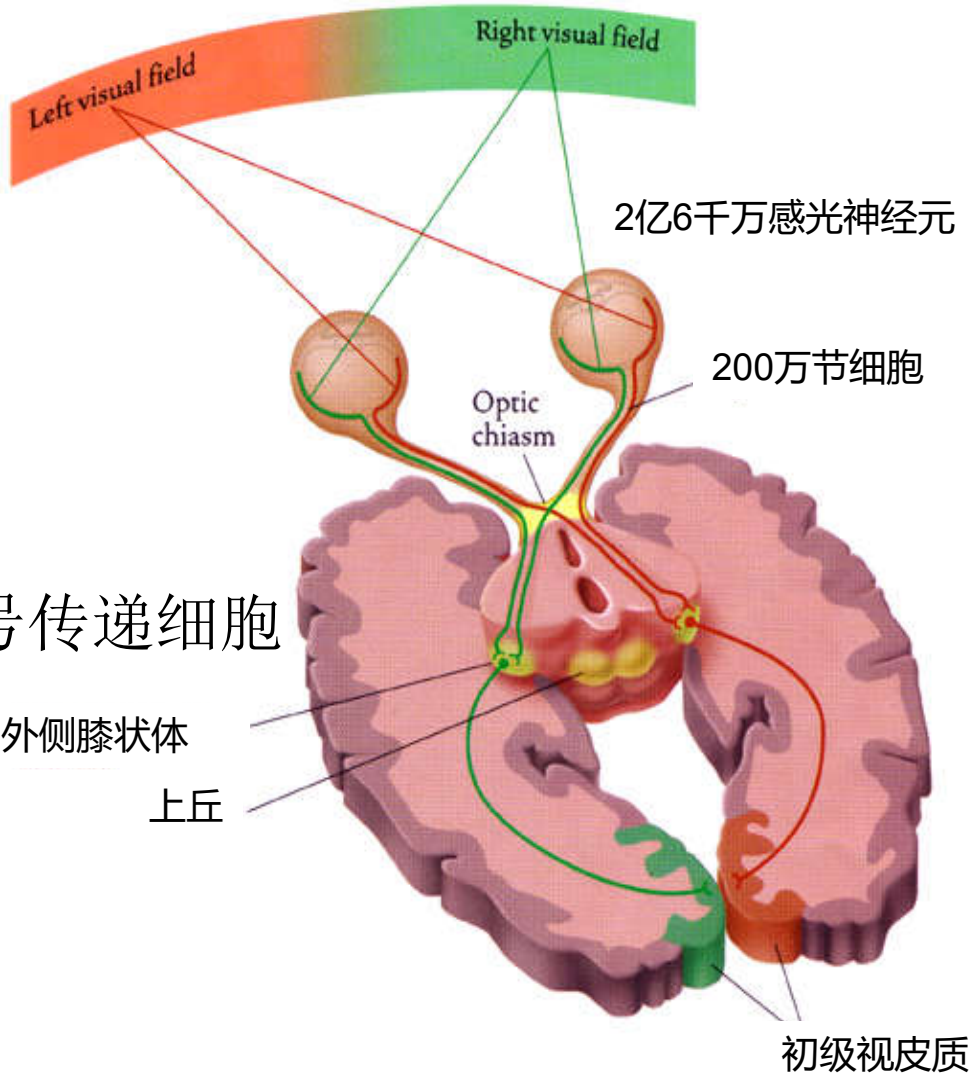
生物学的启示

- 全有和全无
  - 依赖细胞体激活
- 信号压缩
  - 感知细胞远多于信号传递细胞



生物学的启示

- 全有和全无
  - 依赖细胞体激活
- 信号压缩
  - 感知细胞远多于信号传递细胞



ANN研究的范畴

- 神经网络结构和学习方法
- 实现
  - VLSI/FPGA/DSP
  - 光学器件
  - 并行计算（机）
- 生物学
- 心理学
- 认知科学
- 统计学

ANN的历史

- **Pre-1940: von Hemholtz, Mach, Pavlov, etc.**
  - 学习理论, 视觉模型, 条件反射模型
  - 没有用特定的数学模型描述神经元的动作
- **1940s: Hebb, McCulloch and Pitts**
  - 提出生物神经元的学习机制和感知机网络
  - 证明仿生的神经元网络具有模式识别的能力
- **1950s: 康奈尔大学教授Rosenblatt, Widrow and Hoff**
  - 第一次引入一个新的学习算法用于训练线性神经网络
- **1960s: Minsky and Papert**
  - 阐明当时存在神经网络的局限性, 因为新学习算法不能解决这些问题，神经网络的进展基本停滞
- **1970s: Amari, Anderson, Fukushima, Grossberg, Kohonen**
  - 局部的研究进展, 提出可实现记忆功能的自组织网络模型

ANN的历史

- 1980s: Grossberg, Hopfield, Kohonen, Rumelhart, etc.
  - 提出用于训练多层感知机网络的算法，使人工神经网络重新快速发展
  - Hinton, David Rumelhart 反向传播方法
  - Yann Lecun，卷积网络，手写体识别
- 1990-2000:
  - 各种应用出现
  - 研究逐渐减少
  - 1992年J. Schmidhube基于递归网络提出一种训练方法
- 2000-2010:
  - 少见与主流研究领域，相对停滞
  - 深度学习概念在2007年前后开始出现
- 2010-2018:
  - 数据更易获取、计算能力的发展，及深度网络的有效性得到验证，使其重新兴起
  - Facebook：Torch框架
  - Google：Tensorflow框架
  - Amazon：DSSTNE系统

ANN的历史

- 2019:

应用

- 航空
  - 高性能自动驾驶仪, 飞行路径模拟, 飞机控制系统, 自动化驾驶, 飞机部件模拟, 飞机部件
- 汽车
  - 自动驾驶和导航系统,驾驶员行为分析器
- 银行
  - 支票和公文阅读器, 信贷申请评估器
- 国防
  - 武器操纵, 目标跟踪, 目标辨识, 面部识别, 新型传感器, 声纳, 雷达（包括图像信号处理与压缩, 特征提取,信号除噪,信号图像识别）
- 电子
  - 代码序列预测, 集成电路布局, 过程控制, 芯片故障分析, 机器视觉,语音合成, 非线性建模
- 影视娱乐

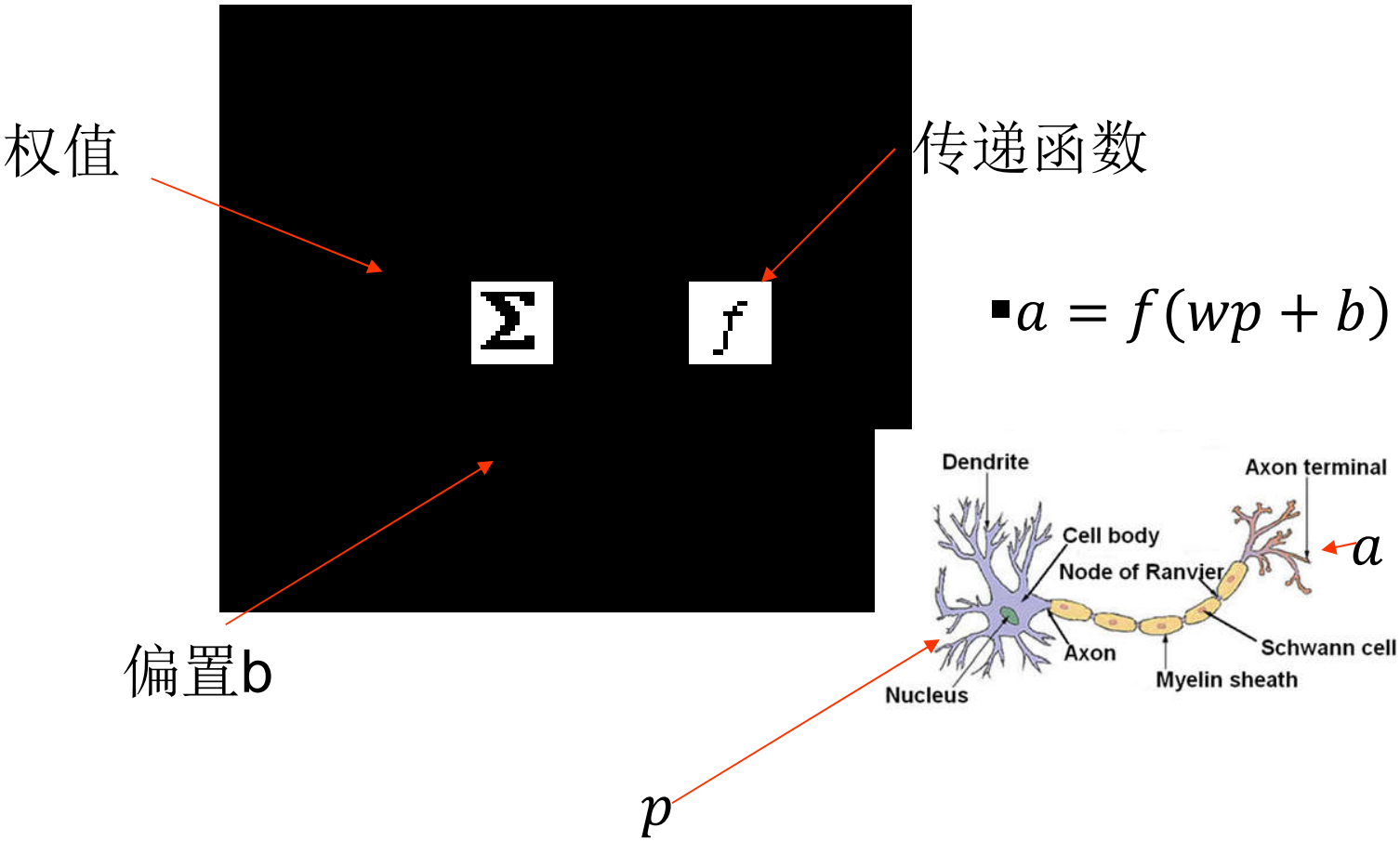
应用

- 金融
  - 不动产评估, 借代咨询,公司证券分级,投资交换程序,公司财务分析,价格预测,政策评估
- 制造业
  - 生产流程控制, **产品设计和分析**, 过程和机器诊断, 可视化质量检测系统, 啤酒检测, 焊接质量分析, 制造质量预测, 计算机芯片分析, 磨床运转分析, 化学产品预测与分析, 机器维护模式分析, 项目投标,**计划和管理**,动态化工流程建模
- 医疗
  - 乳腺癌细胞分析, 修复设计, 移植次数优化, 医院费用节流, 医疗质量改进, 急诊室规划
- 机器人
  - 轨道控制,操作手控制,视觉控制
- 交通
  - 卡车制动器诊断,车辆调度,运送系统分析

大纲

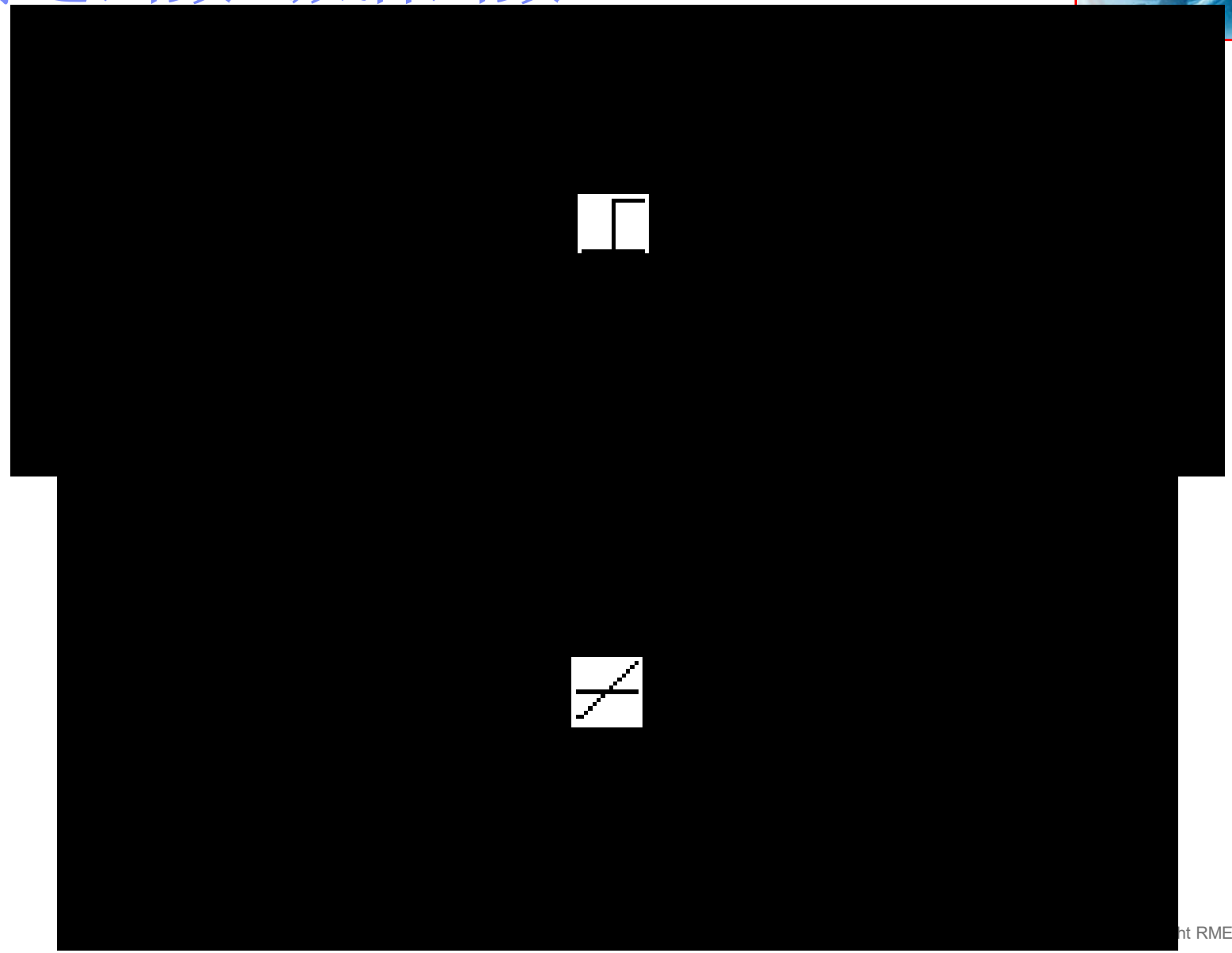
- 神经网络概述
- 感知机模型和结构
- 感知机应用实例
- 学习算法
- 简单扩展

单一输入的神经元

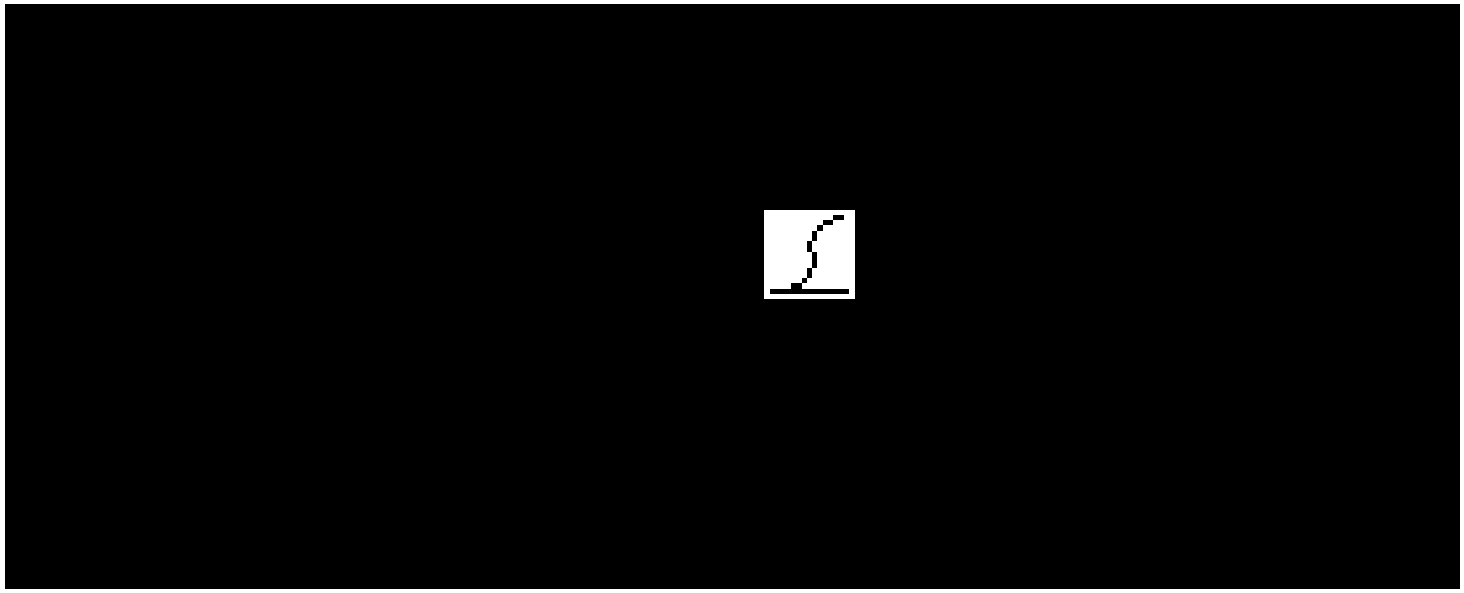


- $a = f(wp + b)$

传递函数 /激活函数

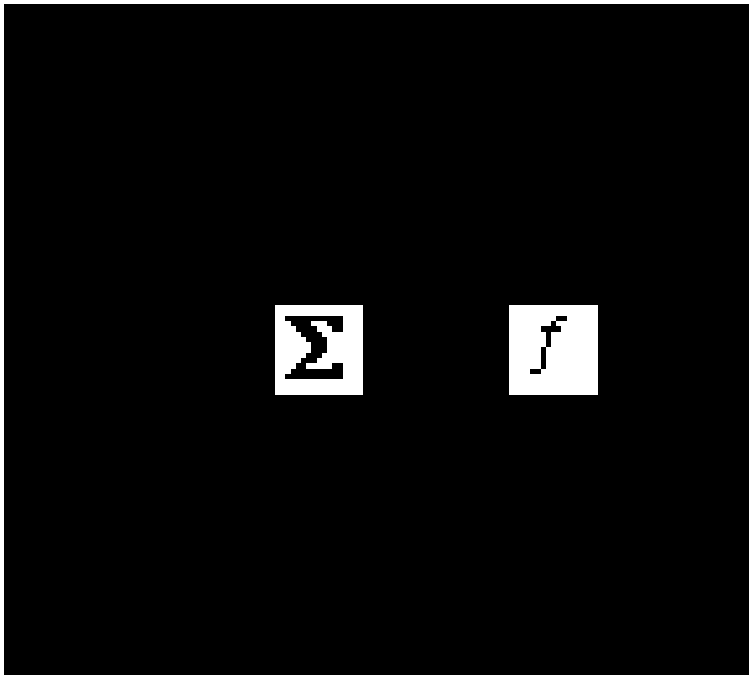


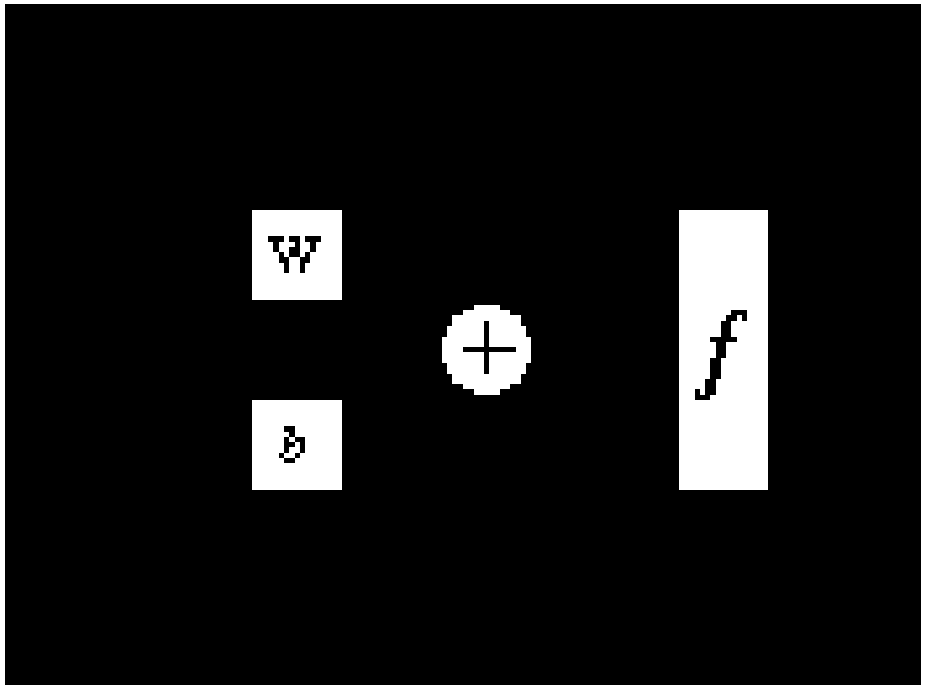
传递函数 /激活函数



- $a = \frac{1}{1+e^{-n}}$

多输入的神经元

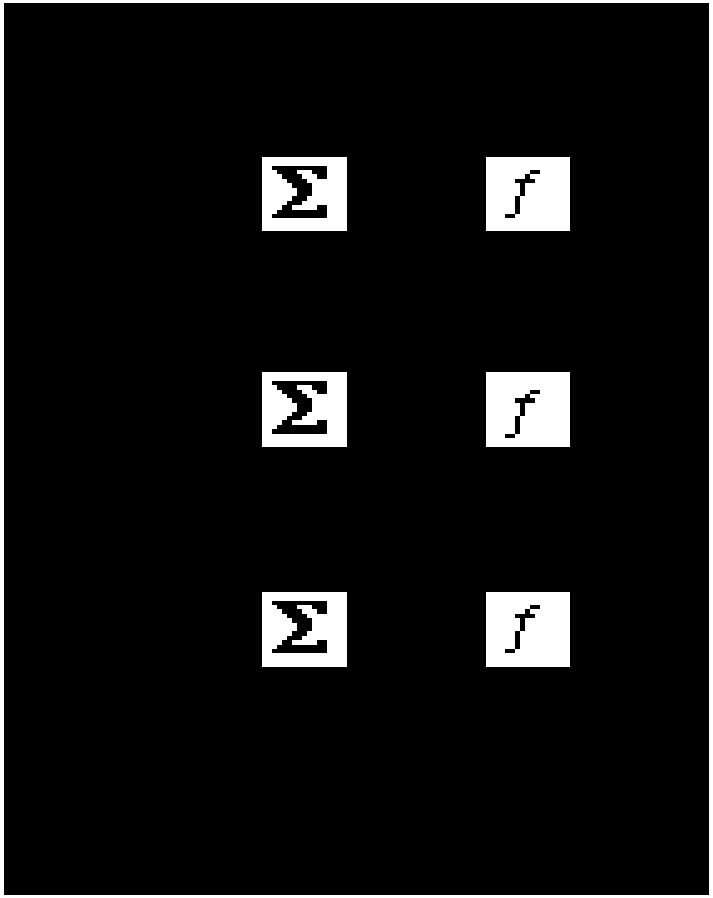




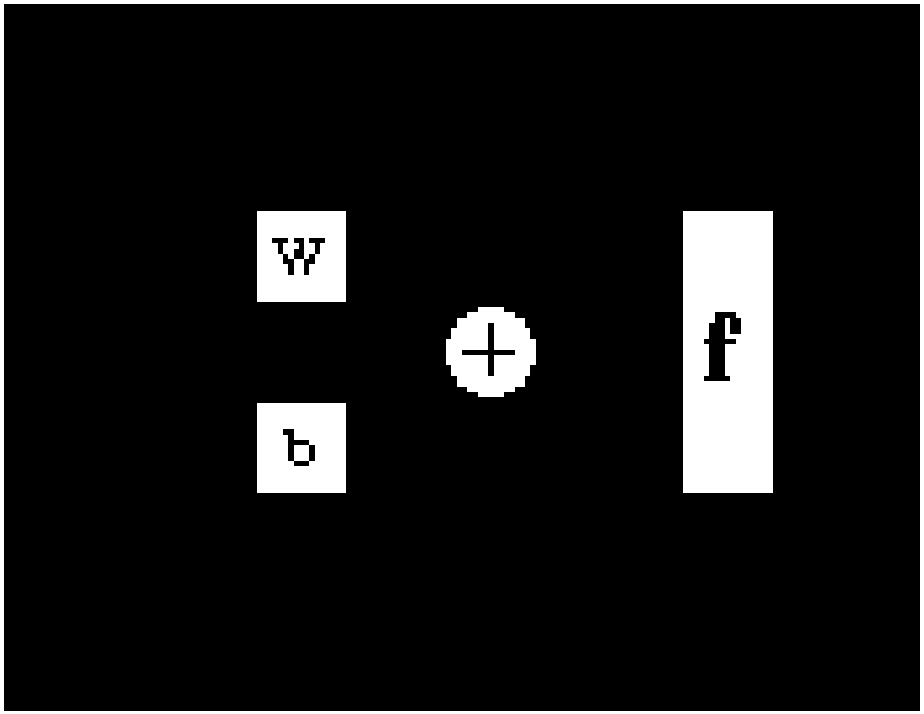
- $a = f(\boldsymbol{wp} + b)$

多输入神经元的简化表达

多输入多神经元



简化的表达方式

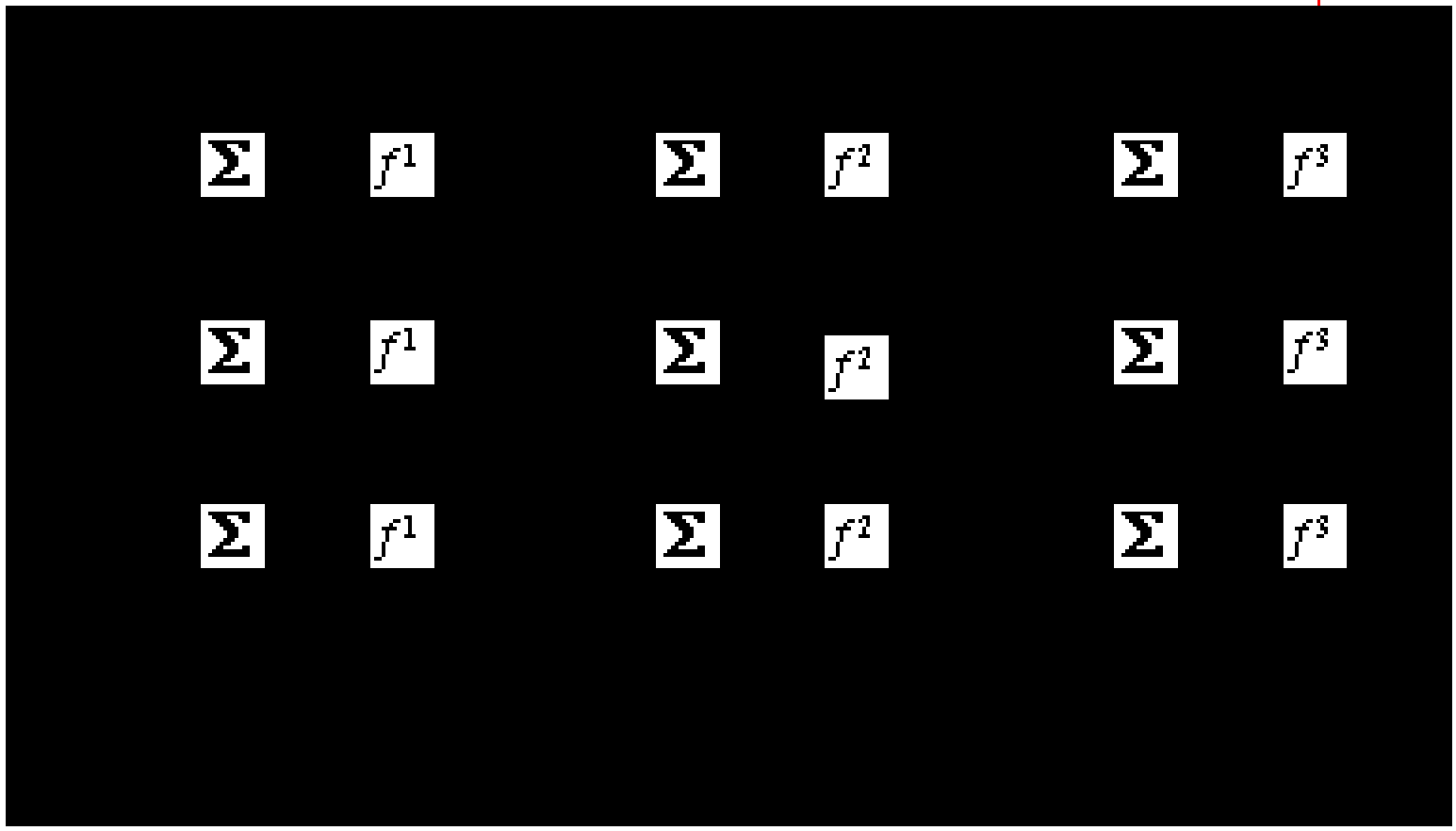


$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,R} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,R} \\ & & & \\ w_{S,1} & w_{S,2} & \cdots & w_{S,R} \end{bmatrix}$$

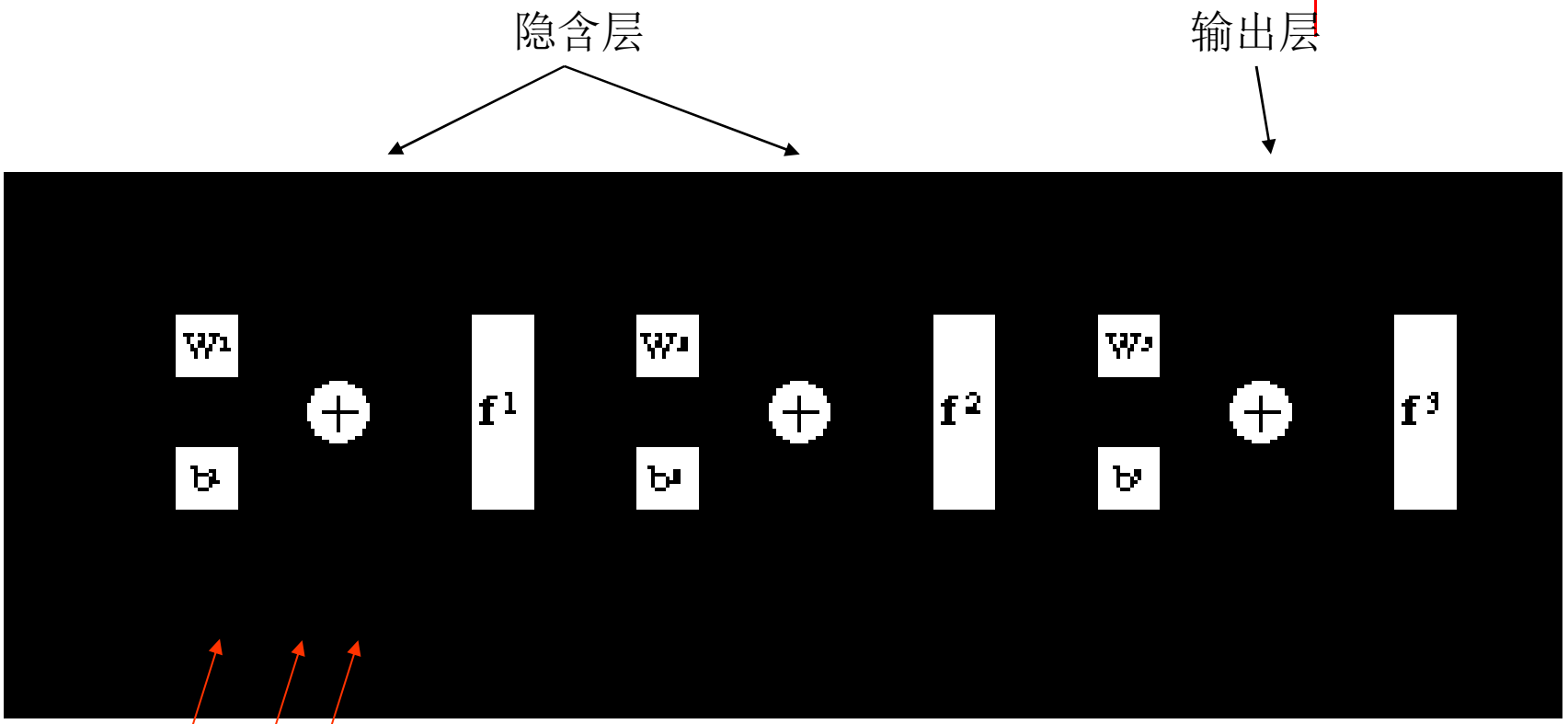
$$\mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ \\ p_R \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \\ b_S \end{bmatrix} \quad \mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \\ a_S \end{bmatrix}$$

- R:输入个数
- b:偏置
- S:神经元个数
- a:输出
- W的下标为（输出神经元，输入神经元）

多层网络



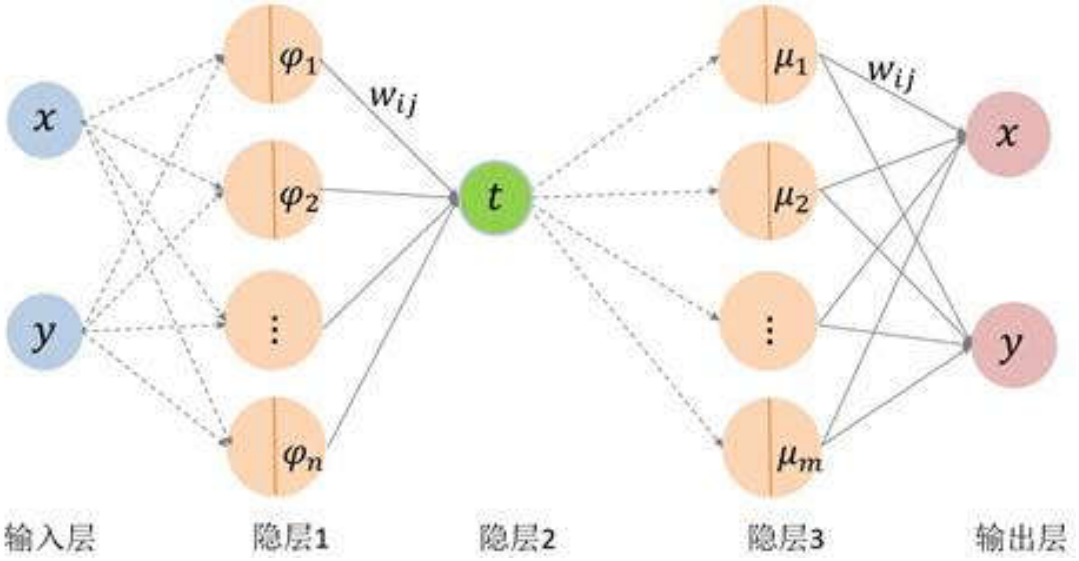
简化表达



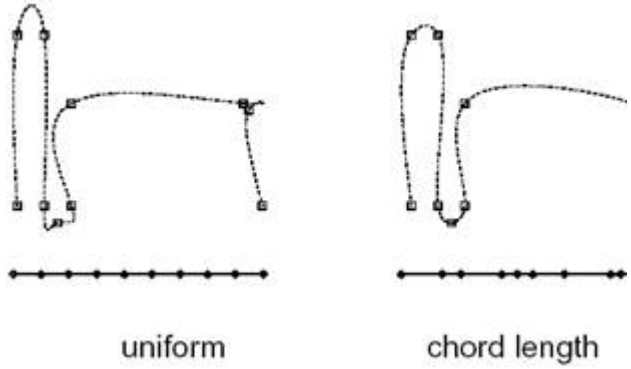
多层神经元使用上标区分

理解多层感知机

- 多层网络仍可以看做是一个拟合函数
- 隐含层中的输出可以看做特征



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



问题

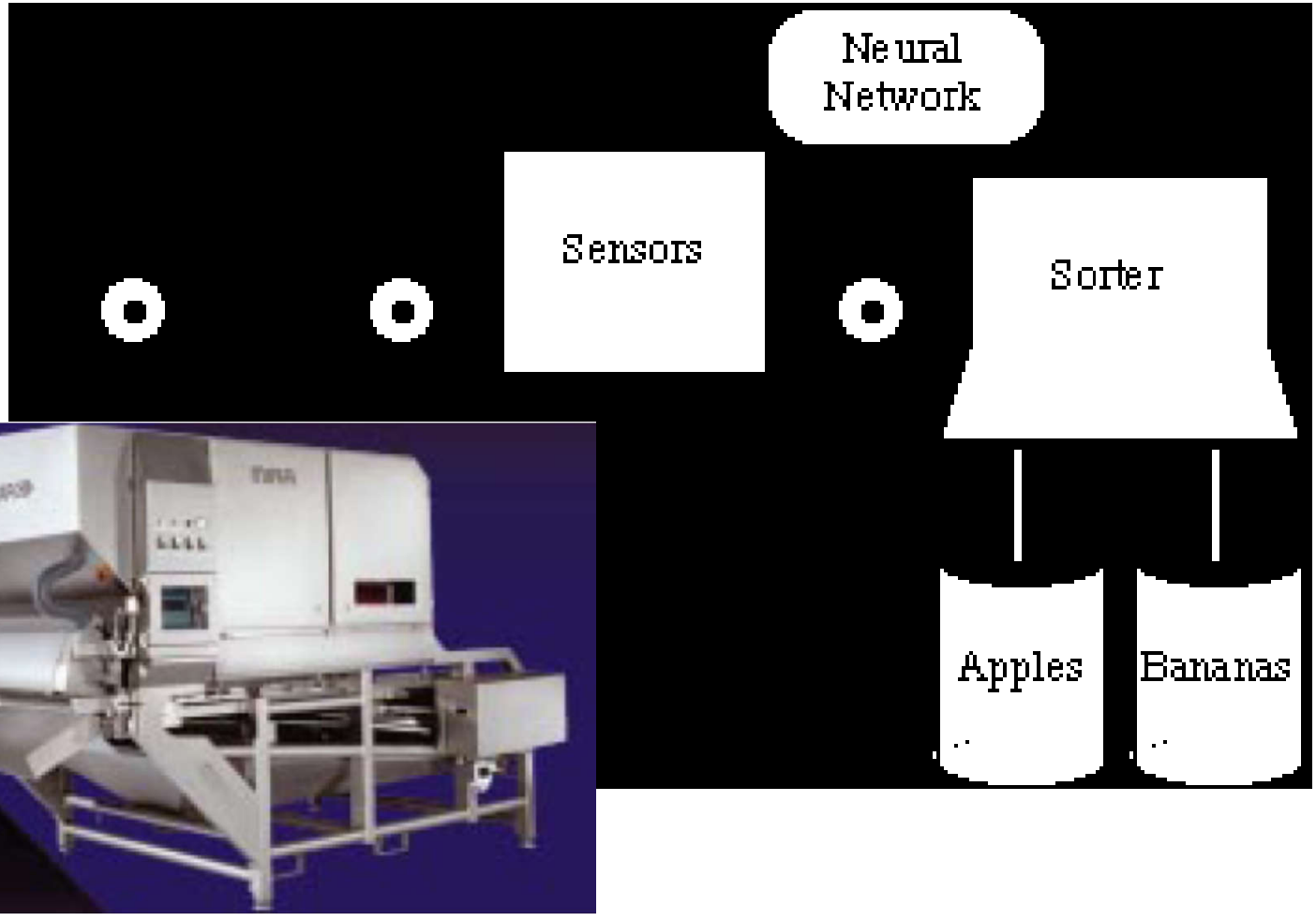
现有一个单层网络,具有6个输入和2个输出.输出被限制在0,1之间,叙述网络的结构,请说明:

- 需要多少个神经元? 2个
- 权值矩阵的维数是多少? 2×6
- 能够采用什么传递函数? 对数s形函数
- 需要采用偏置吗? 视情况而定

大纲

- 神经网络概述
- 神经元模型和网络结构
- 感知机应用实例
- 简单扩展
- 学习算法

感知机应用实例



问题描述

评价向量

$$p=\begin{bmatrix}形状\\纹理\\重量\end{bmatrix}$$

形状: {1 : 球形; -1 :非球形}
纹理: {1 : 光滑; -1 :粗糙}
重量: {1 : > 0.5 lKg. ; -1 : < 0.5 Kg.}

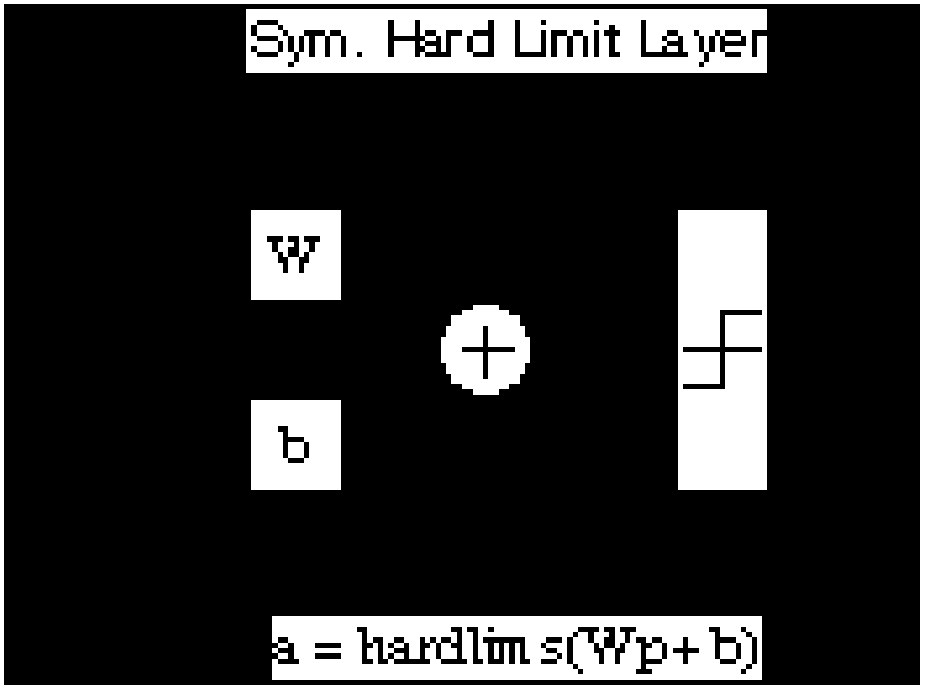
香蕉的标准向量

$$\mathbf{p}_1 = \begin{bmatrix}-1\\1\\-1\end{bmatrix}$$

苹果的标准向量

$$\mathbf{p}_2 = \begin{bmatrix}1\\1\\-1\end{bmatrix}$$

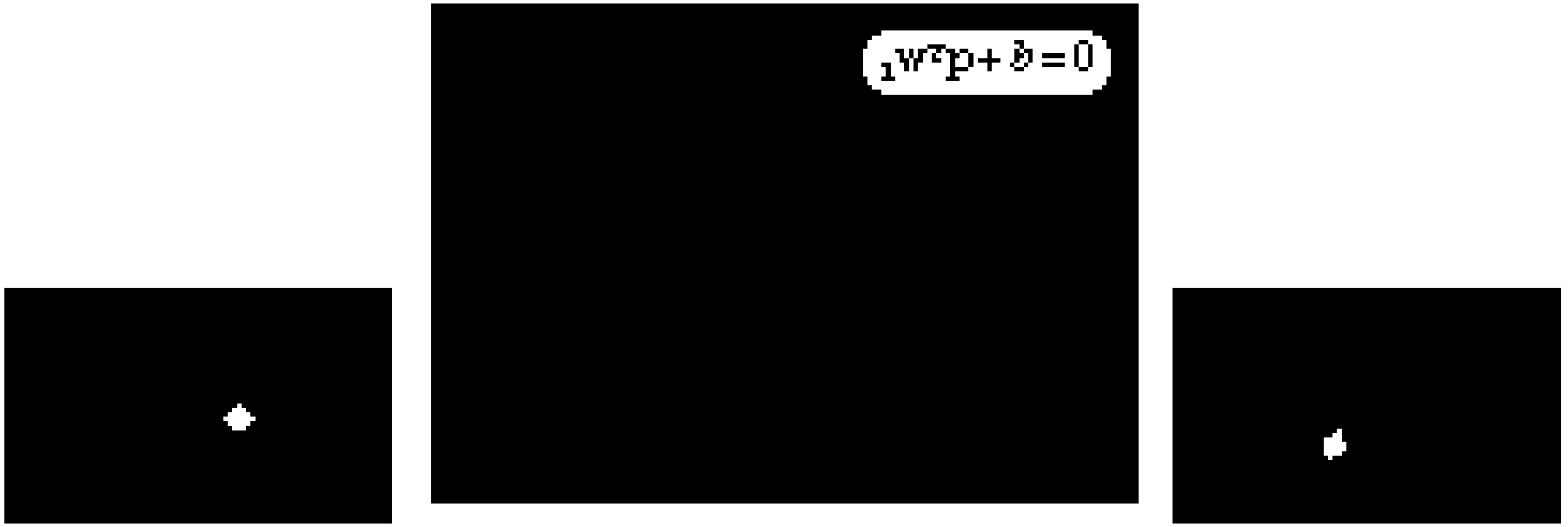
感知机(Perceptron)



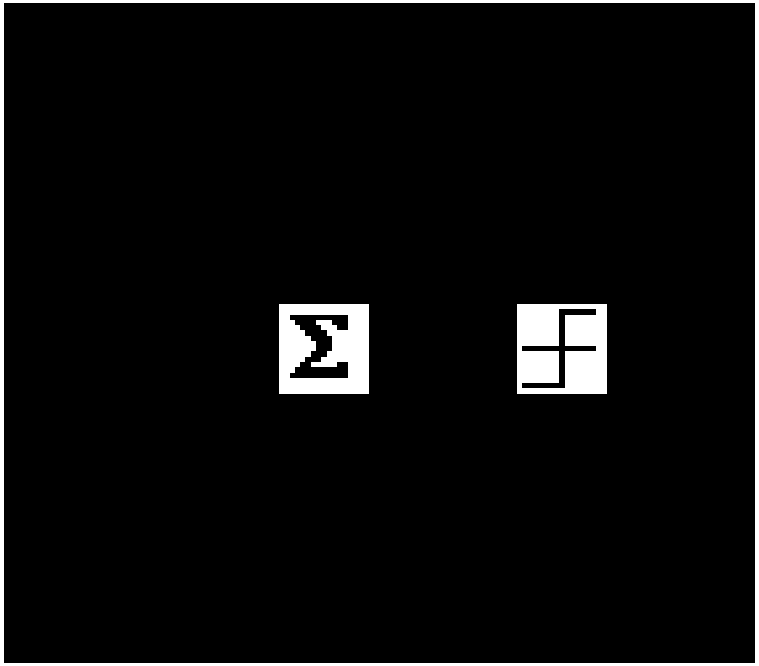
感知机的判定边界

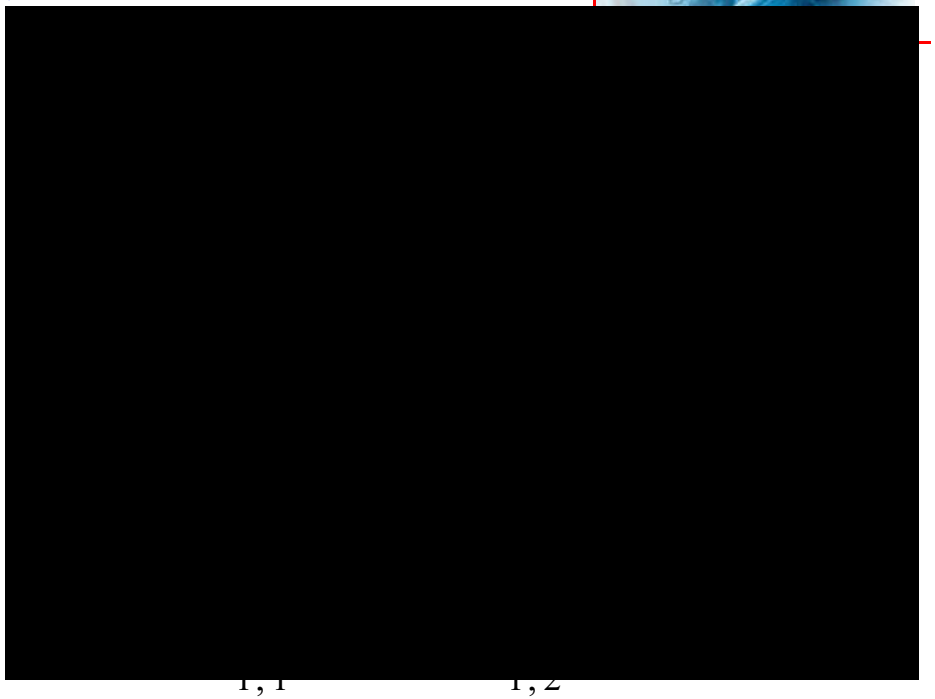
$${}_1\mathbf{w}^T\mathbf{p} + b = 0 \qquad {}_1\mathbf{w}^T\mathbf{p} = -b$$

- 对位于判定边界上的所有点而言，它们的输入向量与权值向量的内积均相等
- 因此这些输入向量在权值向量上的投影都相同，所以他们在空间所在直线必定垂直于权值向量。



接收两个输入的感知机





$$a = hardlims(n) = hardlims(\begin{bmatrix}1 & 2\end{bmatrix}\mathbf{p} + (-2))$$

正因为边界是线性的，所以只能应用于线性可分的问题

$$\mathbf{W}\mathbf{p} + b = 0 \qquad \begin{bmatrix}1 & 2\end{bmatrix}\mathbf{p} + (-2) = 0$$

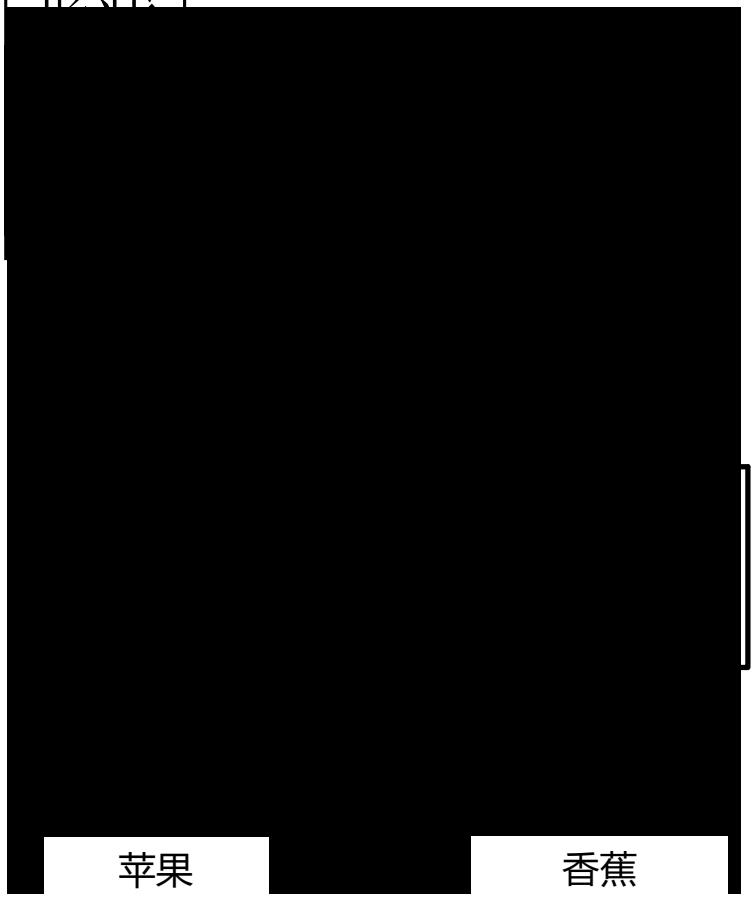
香蕉苹果分类器的例子

$$a = hardlims\left(\begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \end{bmatrix}\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} + b\right)$$

判定边界应该可以区分苹果和香蕉的评价向量.

$p_1 = 0$ 形状: {1 : 球形; -1 :非球形}

p=



$\mathbf{p}_2$

权值向量的设定：

- 垂直于判定边界,
- 指向偏向于产生输出为1的评价向量 的方向

偏置决定了判定边界的位置

$$\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} + 0 = 0$$

网络测试

香蕉:

$$a = hardlims\left(\begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}-1\\+1\\-1\end{bmatrix}+0\right) = -1$$

香蕉

苹果:

$$a = hardlims\left(\begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}+1\\+1\\-1\end{bmatrix}+0\right) = 1$$

苹果

具有“粗糙”外表的香蕉

$$a = hardlims\left(\begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}-1\\-1\\-1\end{bmatrix}+0\right) = -1$$

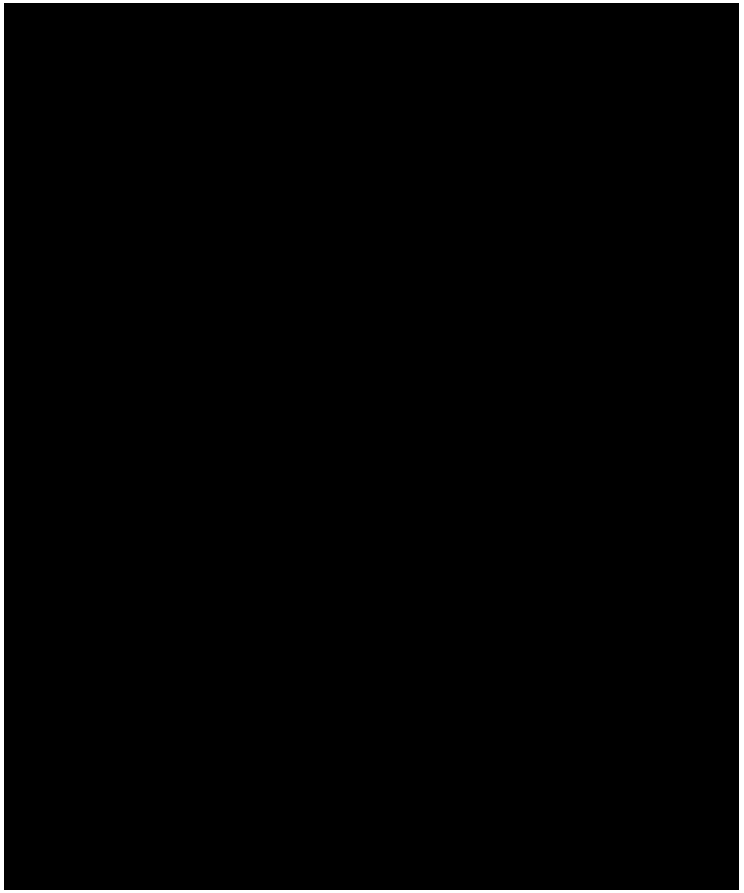
香蕉

问题

- 我们介绍了三种不同网络识别香蕉和苹果，现在假设要区分香蕉和菠萝，请设计一个感知机来识别这两种模式。

$$p=\begin{bmatrix}\text{形状}\\ \text{纹理}\\ \text{重量}\end{bmatrix} \qquad p_1=\begin{bmatrix}1\\ -1\\ 1\end{bmatrix} \qquad p=\begin{bmatrix}1\\ 1\\ -1\end{bmatrix}$$

问题



空间维数较高时，如何设计感知器网络？

大纲

- 神经网络概述
- 神经元模型和网络结构
- 感知机应用实例
- 学习算法
- 简单扩展

感知机学习

回顾

学习是根据样本不断改善性能的过程
（也称为训练）

感知机的学习规则

- 有监督的学习

提供一组描述网络行为的实例集合 (输入/结果)，这组集合可称为训练集。

$$\{\mathbf{p}_1,\mathbf{t}_1\},\{\mathbf{p}_2,\mathbf{t}_2\},\ldots,\{\mathbf{p}_Q,\mathbf{t}_Q\}$$

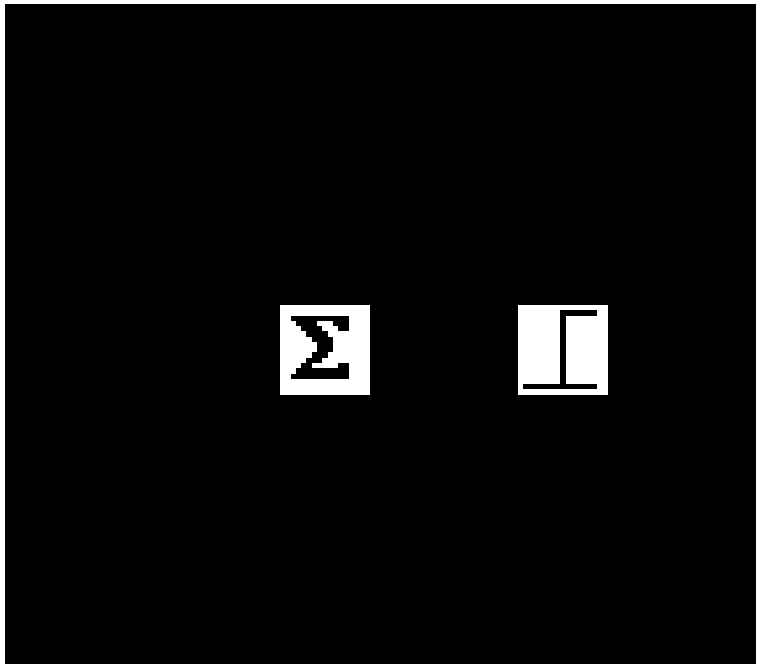
- 增强学习

给出基本样本，或仅仅给出一些级别，表明网络在某些输入下性能的测度。

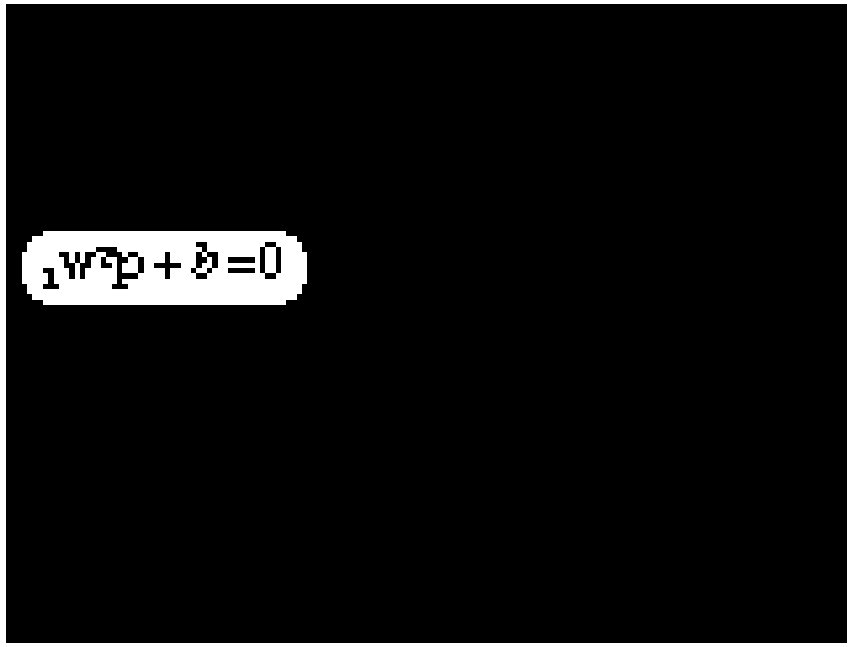
- 非监督式学习

仅仅根据网络的输入调整网络的权值和偏置值，大多数这种算法是要完成某项聚类功能。

单神经元的感知机



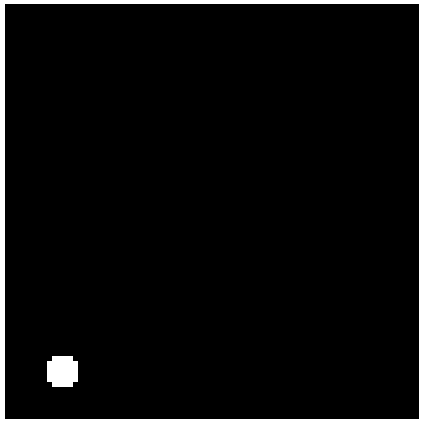
$w_{1,1} = 1 \qquad w_{1,2} = 1 \qquad b = -1$



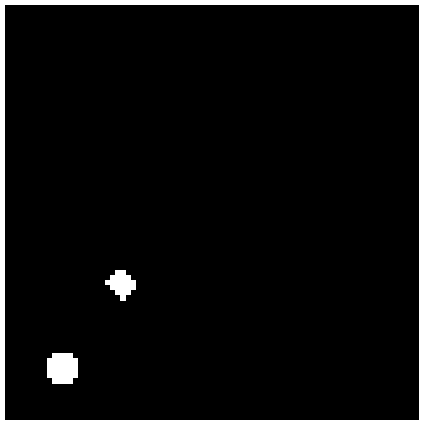
$$a = hardlim({}_1\mathbf{w}^{T}\mathbf{p} + b) = hardlim(w_{1,1}p_1 + w_{1,2}p_2 + b)$$

实现逻辑运算- OR 的例子

$$\left\{\mathbf{p}_1 = \begin{bmatrix}0\\0\end{bmatrix}, t_1 = 0\right\} \quad \left\{\mathbf{p}_2 = \begin{bmatrix}0\\1\end{bmatrix}, t_2 = 1\right\} \quad \left\{\mathbf{p}_3 = \begin{bmatrix}1\\0\end{bmatrix}, t_3 = 1\right\} \quad \left\{\mathbf{p}_4 = \begin{bmatrix}1\\1\end{bmatrix}, t_4 = 1\right\}$$



实现网络



权值向量垂直于判定边界

$$_1\mathbf{w} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

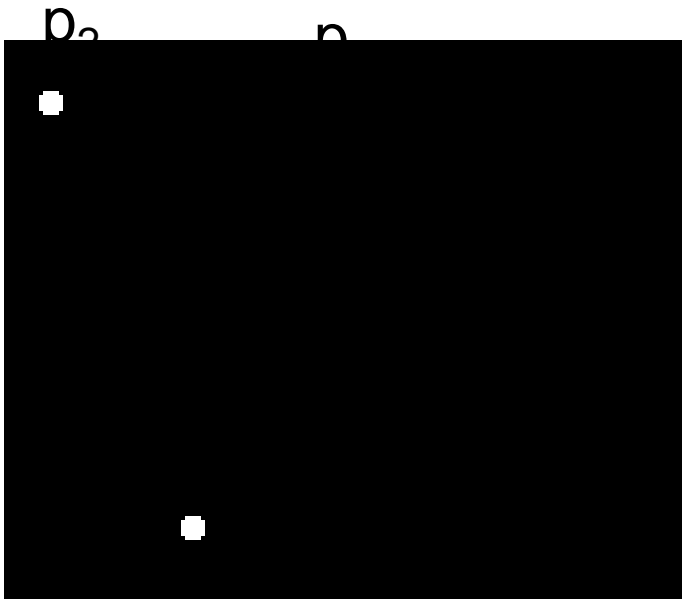
在判定边界上选择一点，帮助确定偏置b.

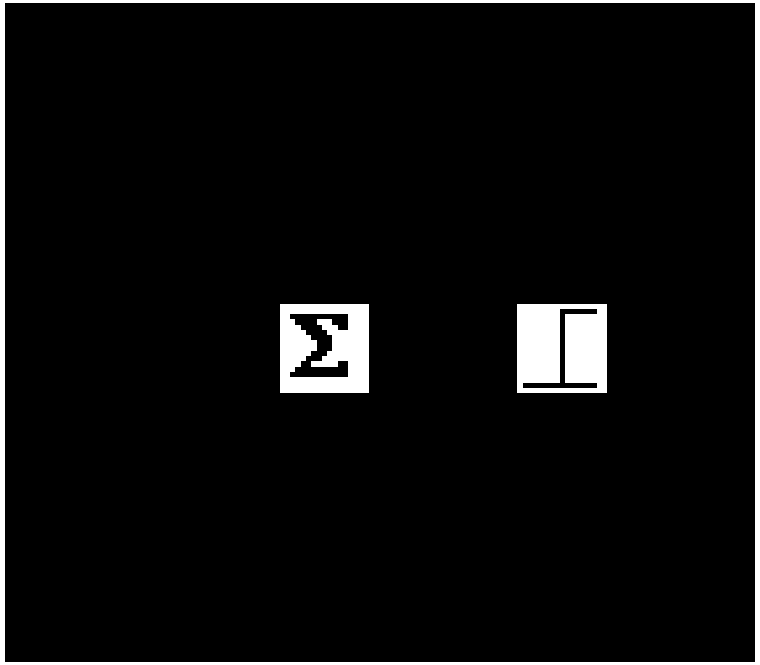
- $_1\mathbf{w}^T\mathbf{p} + b = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \begin{bmatrix} 0.9 \\ 0 \end{bmatrix} + b = 0.45 + b \Rightarrow b = -0.45$

学习感知机权值

$$\{\mathbf{p}_1, \mathbf{t}_1\}, \{\mathbf{p}_2, \mathbf{t}_2\}, \ldots, \{\mathbf{p}_Q, \mathbf{t}_Q\}$$

$$\left\{\mathbf{p}_1 = \begin{bmatrix}1\\2\end{bmatrix}, t_1 = 1\right\} \quad \left\{\mathbf{p}_2 = \begin{bmatrix}-1\\2\end{bmatrix}, t_2 = 0\right\} \quad \left\{\mathbf{p}_3 = \begin{bmatrix}0\\-1\end{bmatrix}, t_3 = 0\right\}$$

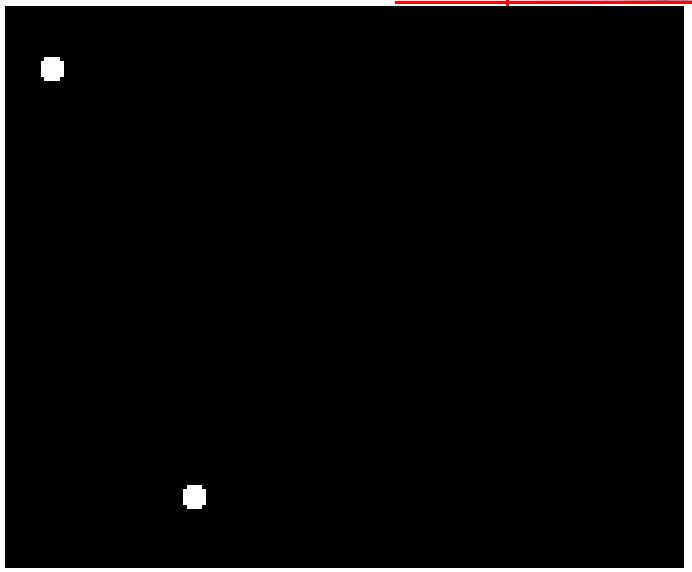




初始权值

随机初始化权值:

$$ {}_1\mathbf{w} = \begin{bmatrix} 1.0 \\ -0.8 \end{bmatrix} $$



$$ \left\{ \mathbf{p}_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, t_1 = 1 \right\} $$

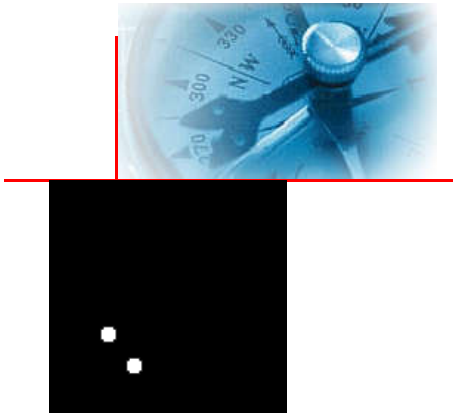
输入 $\mathbf{p}_1$ 到网络:

$$ a = hardlim({}_1\mathbf{w}^T \mathbf{p}_1) = hardlim\left(\begin{bmatrix} 1.0 & -0.8 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right) $$

$$ a = hardlim(-0.6) = 0 $$

非正确的分类

试探性的权值调整方法

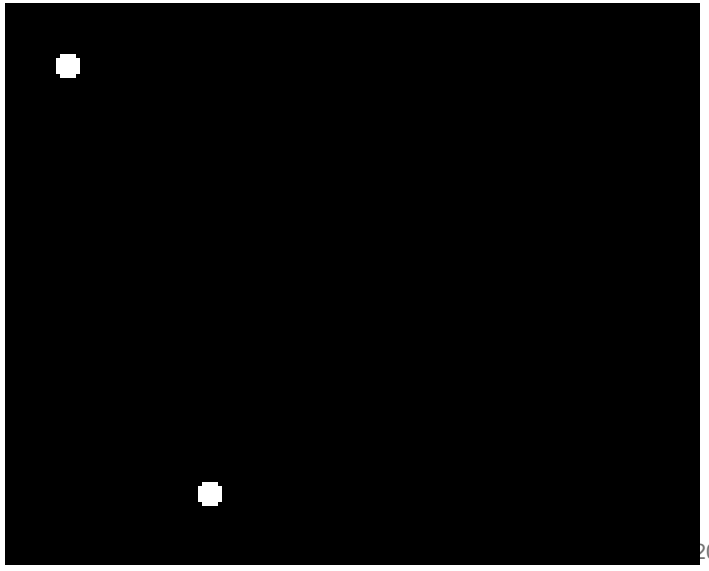


- Set ${}_1\mathbf{w}$ to $\mathbf{p}_1$ ×
  - 不稳定
- Add $\mathbf{p}_1$ to ${}_1\mathbf{w}$ √

试探规则: If $t = 1$ and $a = 0$, then ${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} + \mathbf{p}$

$$\left\{\mathbf{p}_1 = \begin{bmatrix}1\\2\end{bmatrix}, t_1 = 1\right\}$$

$${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} + \mathbf{p}_1 = \begin{bmatrix}1.0\\-0.8\end{bmatrix} + \begin{bmatrix}1\\2\end{bmatrix} = \begin{bmatrix}2.0\\1.2\end{bmatrix}$$



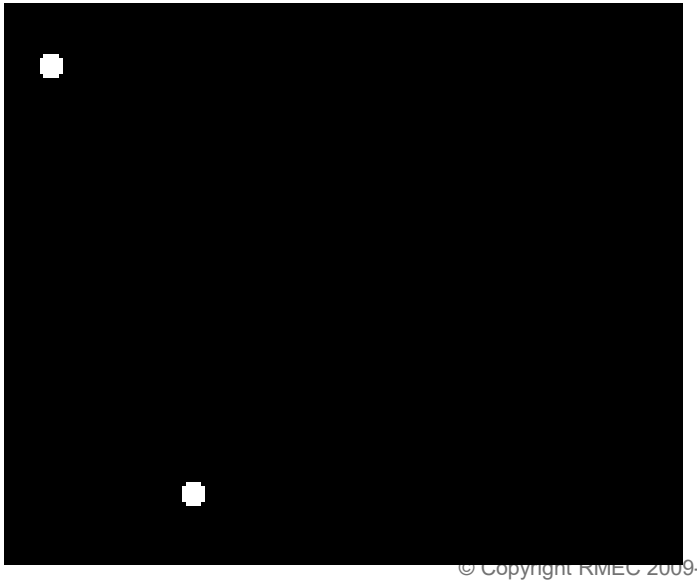
第二个输入向量 $\left\{\mathbf{p}_2 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}, t_2 = 0\right\}$

$$a = hardlim({}_1\mathbf{w}^{\mathrm{T}}\mathbf{p}_2) = hardlim\left(\begin{bmatrix} 2.0 & 1.2 \end{bmatrix}\begin{bmatrix} -1 \\ 2 \end{bmatrix}\right)$$

$$a = hardlim(0.4) = 1 \quad \text{(非正确分类)}$$

更改规则: If $t = 0$ and $a = 1$, then ${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} - \mathbf{p}$

$${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} - \mathbf{p}_2 = \begin{bmatrix} 2.0 \\ 1.2 \end{bmatrix} - \begin{bmatrix} -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 3.0 \\ -0.8 \end{bmatrix}$$



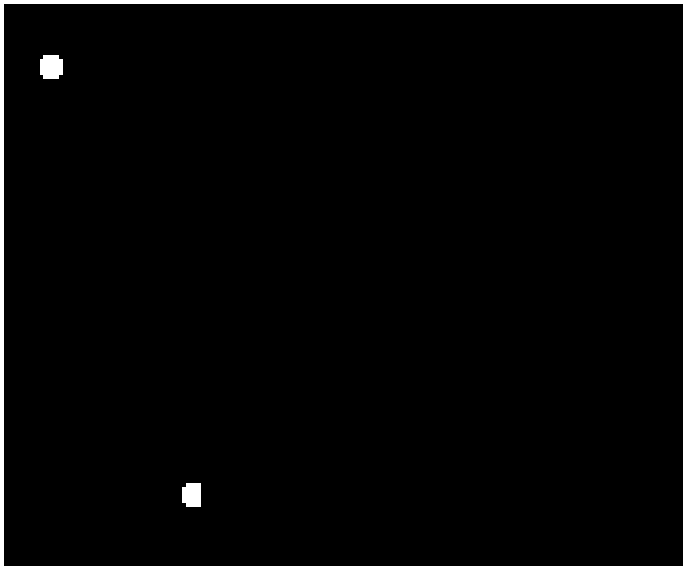
第三个输入向量

$$\left\{\mathbf{p}_3 = \begin{bmatrix}0\\-1\end{bmatrix}, t_3 = 0\right\}$$

$$a = hardlim({}_1\mathbf{w}^T\mathbf{p}_3) = hardlim\left(\begin{bmatrix}3.0 & -0.8\end{bmatrix}\begin{bmatrix}0\\-1\end{bmatrix}\right)$$

$$a = \text{hardlim}(0.8) = 1 \qquad \text{(非正确分类)}$$

$${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} - \mathbf{p}_3 = \begin{bmatrix}3.0\\-0.8\end{bmatrix} - \begin{bmatrix}0\\-1\end{bmatrix} = \begin{bmatrix}3.0\\0.2\end{bmatrix}$$



继续迭代训练，最终将可以进行正确模式识别

If $t = a$, then ${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old}$.

统一的学习规则

If $t = 1$ and $a = 0$, then ${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} + \mathbf{p}$

If $t = 0$ and $a = 1$, then ${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} - \mathbf{p}$

If $t = a$, then ${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old}$

$$e = t - a$$

If $e = 1$, then ${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} + \mathbf{p}$

If $e = -1$, then ${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} - \mathbf{p}$

If $e = 0$, then ${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old}$

$${}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} + e\mathbf{p} = {}_1\mathbf{w}^{old} + (t-a)\mathbf{p}$$

$$b^{new} = b^{old} + e$$

偏置可以看做一个输入总为1的权值

多神经元感知器

更新权值矩阵的第i行:

$$_i\mathbf{w}^{new} = {}_i\mathbf{w}^{old} + e_i\mathbf{p}$$

$$b_i^{new} = b_i^{old} + e_i$$

矩阵的形式:

$$\mathbf{W}^{new} = \mathbf{W}^{old} + \mathbf{e}\mathbf{p}^T$$

$$\mathbf{b}^{new} = \mathbf{b}^{old} + \mathbf{e}$$

香蕉和苹果

训练集

$$\left\{\mathbf{p}_1 = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}, t_1 = \begin{bmatrix} 1 \end{bmatrix}\right\} \qquad \left\{\mathbf{p}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}, t_2 = \begin{bmatrix} 0 \end{bmatrix}\right\}$$

初始权值

$$\mathbf{W} = \begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix} \qquad b = 0.5$$

初次迭代

$$a = hardlim(\mathbf{W}\mathbf{p}_1 + b) = hardlim\left(\begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix}\begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} + 0.5\right)$$

$$a = hardlim(-0.5) = 0 \qquad e = t_1 - a = 1 - 0 = 1$$

$$\mathbf{W}^{new} = \mathbf{W}^{old} + e\mathbf{p}^T = \begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix} + (1)\begin{bmatrix} -1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} -0.5 & 0 & -1.5 \end{bmatrix}$$

$$b^{new} = b^{old} + e = 0.5 + (1) = 1.5$$

第二次迭代

$$a = hardlim\left(\mathbf{W}\mathbf{p}_2 + b\right) = hardlim\left(\begin{bmatrix} -0.5 & 0 & -1.5 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} + (1.5)\right)$$

$$a = hardlim\,(2.5) = 1$$

$$e = t_2 - a = 0 - 1 = -1$$

$$\mathbf{W}^{new} = \mathbf{W}^{old} + e\mathbf{p}^T = \begin{bmatrix} -0.5 & 0 & -1.5 \end{bmatrix} + (-1)\begin{bmatrix} 1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} -1.5 & -1 & -0.5 \end{bmatrix}$$

$$b^{new} = b^{old} + e = 1.5 + (-1) = 0.5$$

检查

$$a = hardlim(\mathbf{W}\mathbf{p}_1 + b) = hardlim\left(\begin{bmatrix}-1.5 & -1 & -0.5\end{bmatrix}\begin{bmatrix}-1\\1\\-1\end{bmatrix} + 0.5\right)$$

$$a = hardlim(1.5) = 1 = t_1$$

$$a = hardlim(\mathbf{W}\mathbf{p}_2 + b) = hardlim\left(\begin{bmatrix}-1.5 & -1 & -0.5\end{bmatrix}\begin{bmatrix}1\\1\\-1\end{bmatrix} + 0.5\right)$$

$$a = hardlim(-1.5) = 0 = t_2$$

感知器的能力

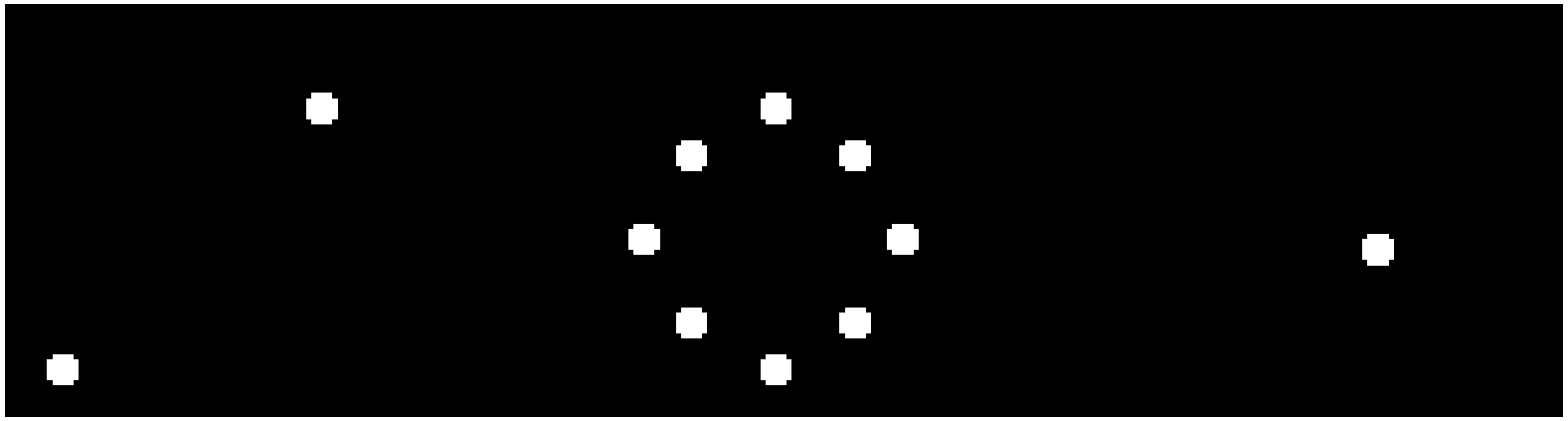
只要问题的解存在，那么感知器的学习规则就一定能够在有限步数后收敛到问题的一个解。

感知器的局限性

只能判定线性边界

$$_1\mathbf{w}^T\mathbf{p} + b = 0$$

线性可分性



大纲

- 神经网络概述
- 神经元模型和网络结构
- 感知机应用实例
- 感知机学习
- 简单扩展

多神经元感知机

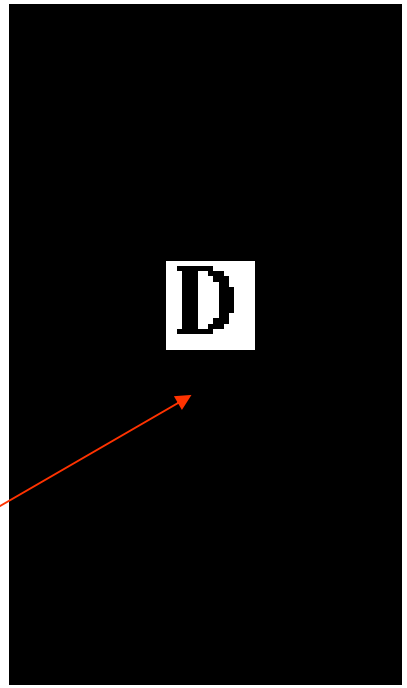
每个神经元都有自己的判定边界。

$$_i\mathbf{w}^T\mathbf{p} + b_i = 0$$

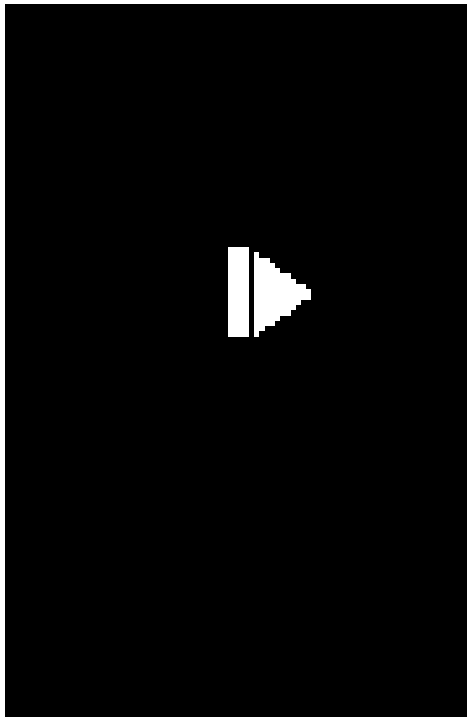
一个独立的神经元可以将输入数据分为两类。

多神经元感知器可以将输入数据划分为 2^S 个类别。

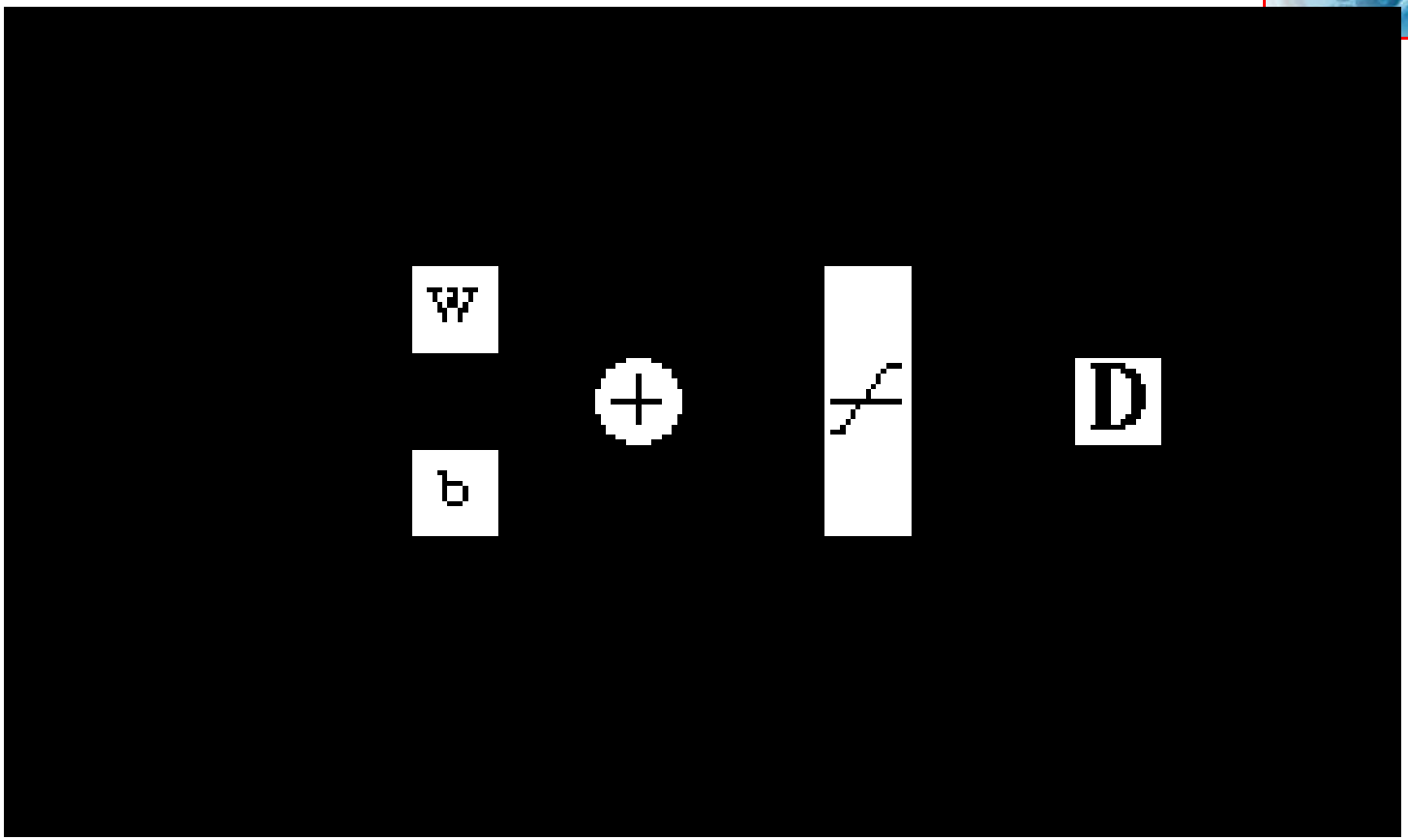
延时模块和积分模块



初始条件



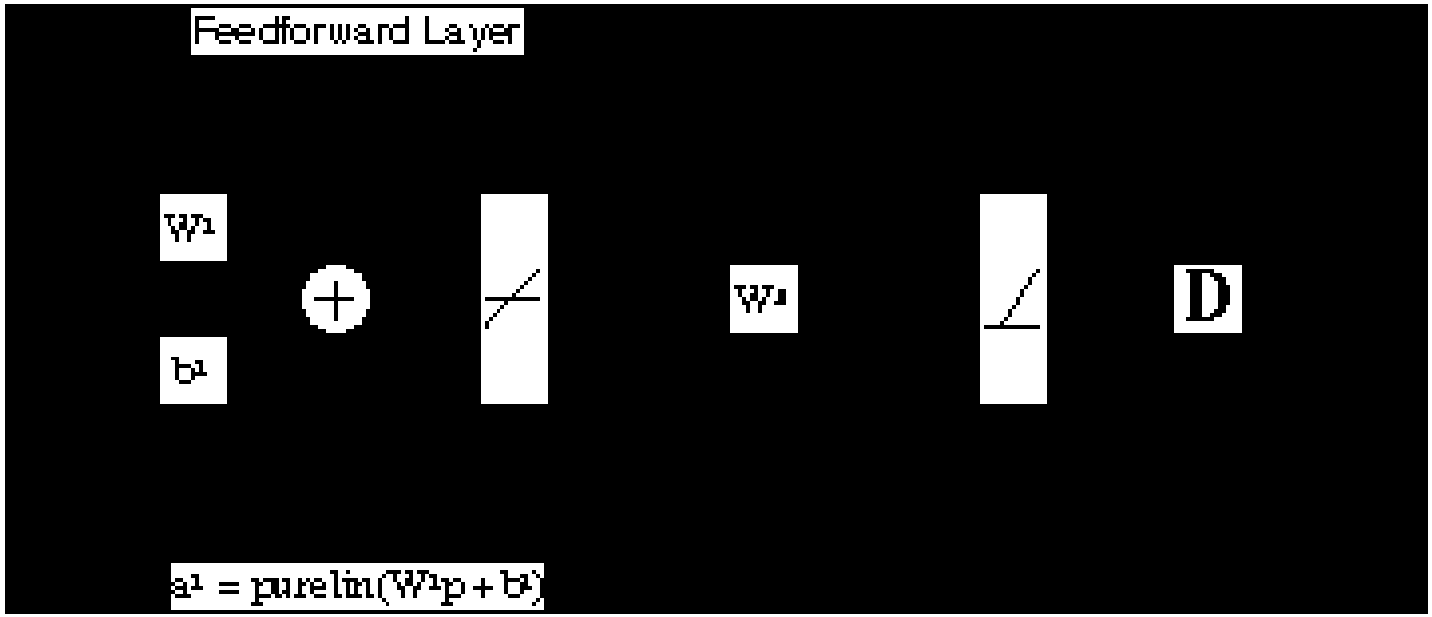
递归网络



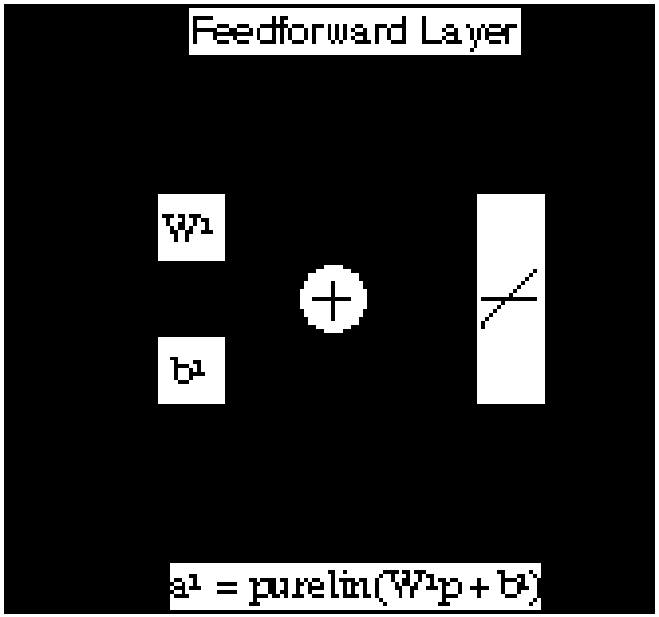
$$\mathbf{a}(1) = \mathbf{satlins}(\mathbf{W}\mathbf{a}(0) + \mathbf{b}) = \mathbf{satlins}(\mathbf{W}\mathbf{p} + \mathbf{b})$$

$$\mathbf{a}(2) = \mathbf{satlins}(\mathbf{W}\mathbf{a}(1) + \mathbf{b})$$

Hamming网络



前馈层



对于香蕉和苹果的识别

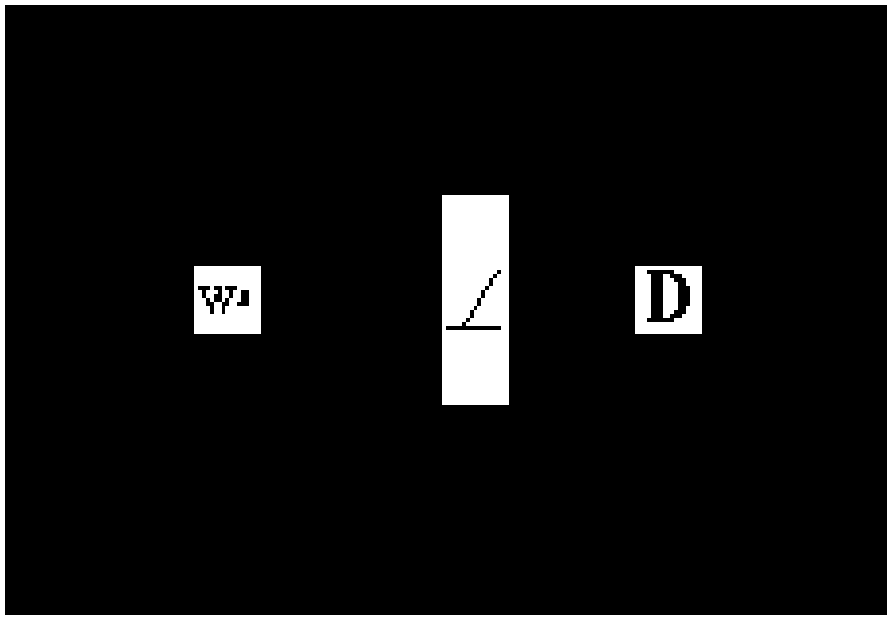
$$S = 2$$

$$\mathbf{W}^1 = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \end{bmatrix} = \begin{bmatrix} -1 & 1 & -1 \\ 1 & 1 & -1 \end{bmatrix}$$

$$\mathbf{b}^1 = \begin{bmatrix} R \\ R \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{a}^1 = \mathbf{W}^1\mathbf{p} + \mathbf{b}^1 = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \end{bmatrix}\mathbf{p} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1^T\mathbf{p} + 3 \\ \mathbf{p}_2^T\mathbf{p} + 3 \end{bmatrix}$$

递归层



$$\mathbf{W}^2 = \begin{bmatrix} 1 & -\varepsilon \\ -\varepsilon & 1 \end{bmatrix} \qquad \varepsilon < \frac{1}{S-1}$$

$$\mathbf{a}^2(t+1) = \mathbf{poslin}\left(\begin{bmatrix} 1 & -\varepsilon \\ -\varepsilon & 1 \end{bmatrix}\mathbf{a}^2(t)\right) = \mathbf{poslin}\left(\begin{bmatrix} a_1^2(t) - \varepsilon a_2^2(t) \\ a_2^2(t) - \varepsilon a_1^2(t) \end{bmatrix}\right)$$

Hamming操作

第一层

输入(Rough 香蕉)

$$\mathbf{p} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

$$\mathbf{a}^1 = \begin{bmatrix} -1 & 1 & -1 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} (1+3) \\ (-1+3) \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

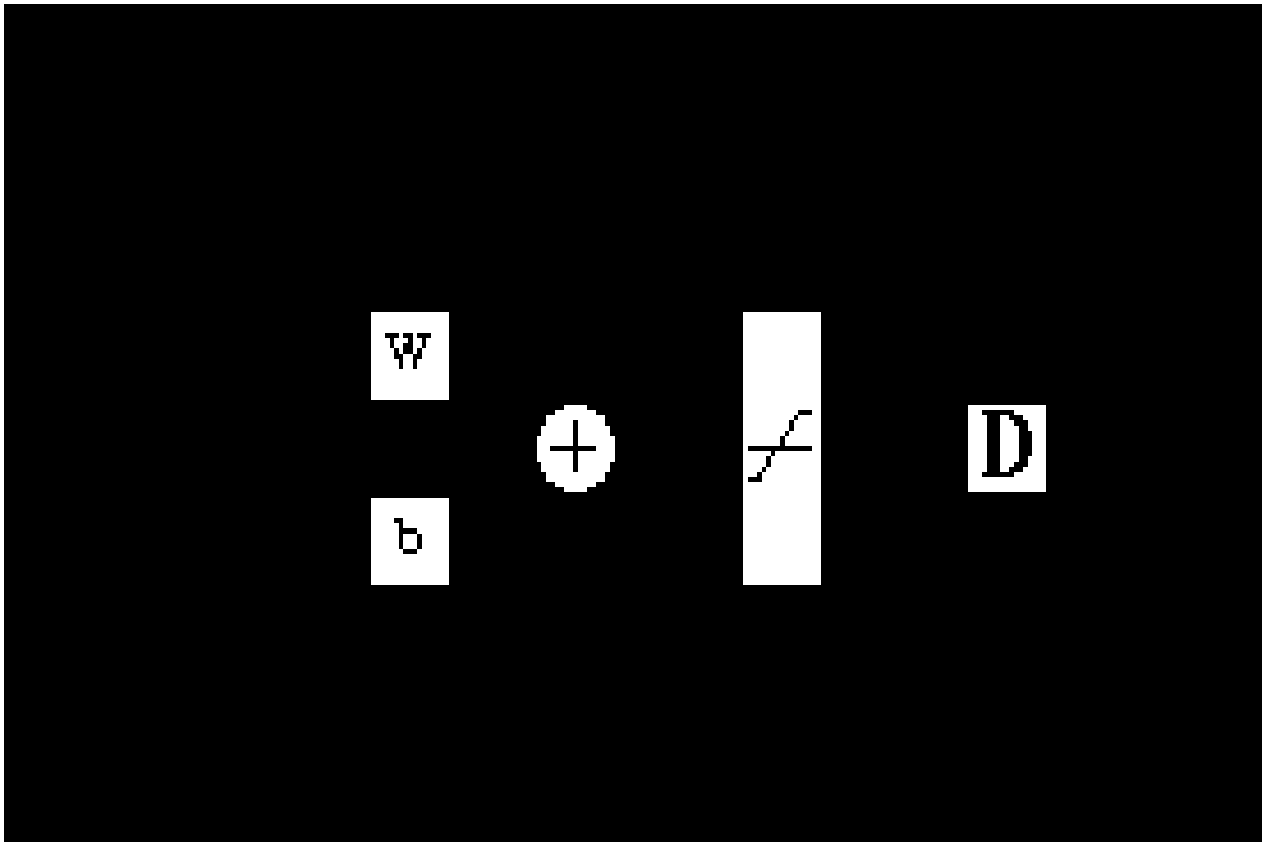
Hamming操作

第二层

$$\mathbf{a}^2(1) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(0)) = \begin{cases} \mathbf{poslin}\left(\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}\begin{bmatrix} 4 \\ 2 \end{bmatrix}\right) \\ \mathbf{poslin}\left(\begin{bmatrix} 3 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix} \end{cases}$$

$$\mathbf{a}^2(2) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(1)) = \begin{cases} \mathbf{poslin}\left(\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}\begin{bmatrix} 3 \\ 0 \end{bmatrix}\right) \\ \mathbf{poslin}\left(\begin{bmatrix} 3 \\ -1.5 \end{bmatrix}\right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix} \end{cases}$$

Hopfield网络



仍然是香蕉和苹果

$$\mathbf{W} = \begin{bmatrix} 1.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 0 \\ 0.9 \\ -0.9 \end{bmatrix}$$

标准向量

香蕉

$$\mathbf{p}_1 = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$$

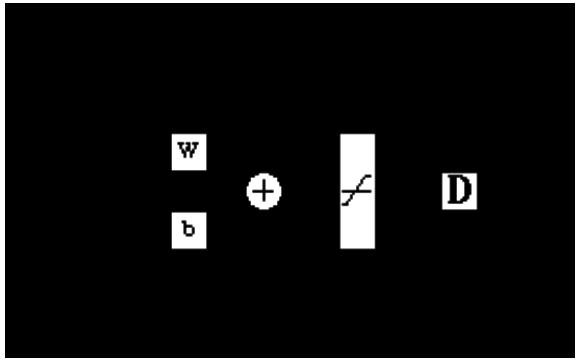
苹果

$$\mathbf{p}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$$

$$a_1(t+1) = satlins(1.2a_1(t))$$

$$a_2(t+1) = satlins(0.2a_2(t) + 0.9)$$

$$a_3(t+1) = \text{satlins}(0.2a_3(t) - 0.9)$$



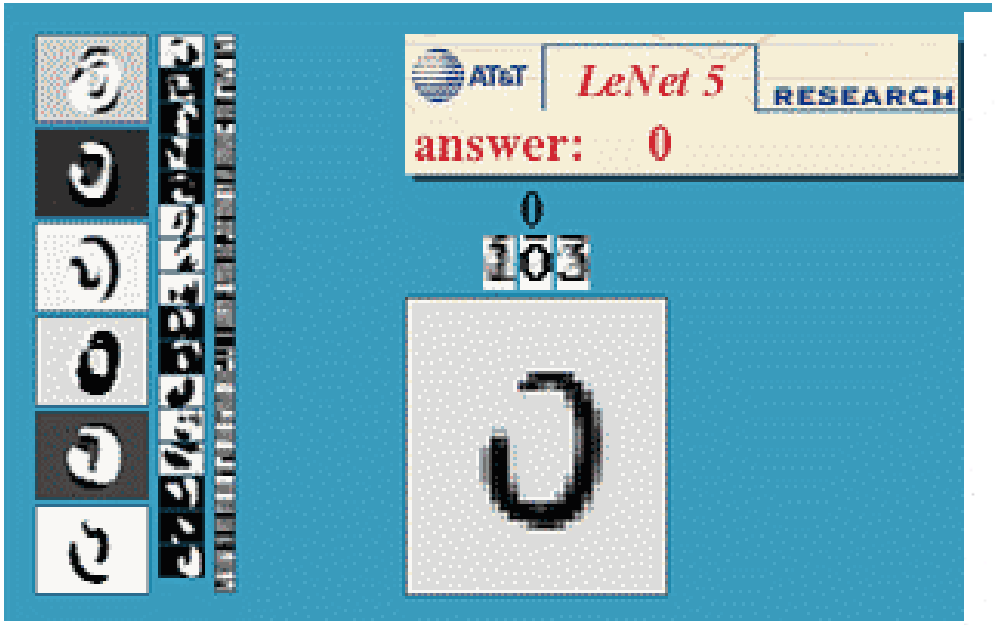
对 “Rough” 香蕉的测试

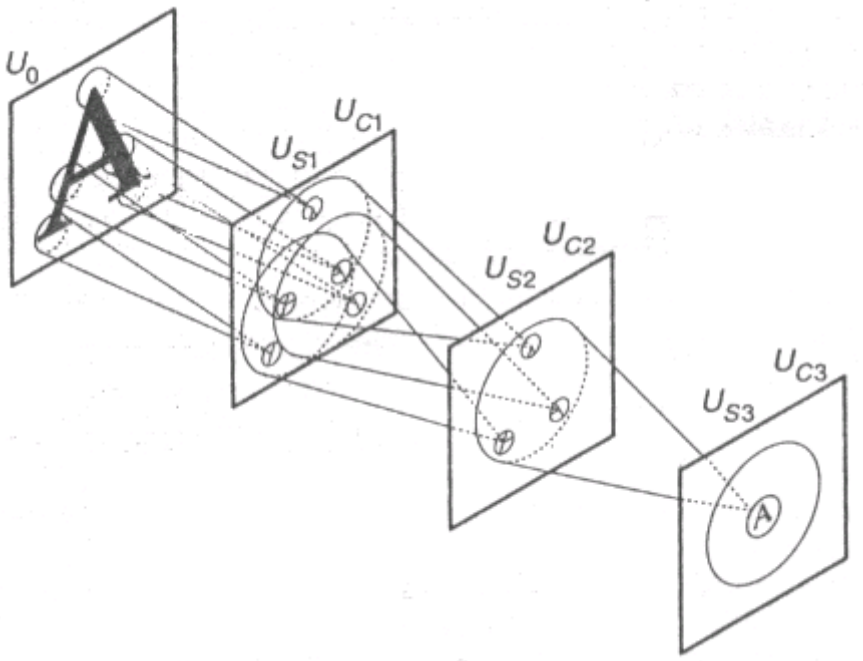
$$\mathbf{a}(0) = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \quad \mathbf{a}(1) = \begin{bmatrix} -1 \\ 0.7 \\ -1 \end{bmatrix} \quad \mathbf{a}(2) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \quad \mathbf{a}(3) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \text{ (香蕉)}$$

扩展

多层感知机可以对问题进行逐层抽象。

深度前馈网络 – 深度扩展





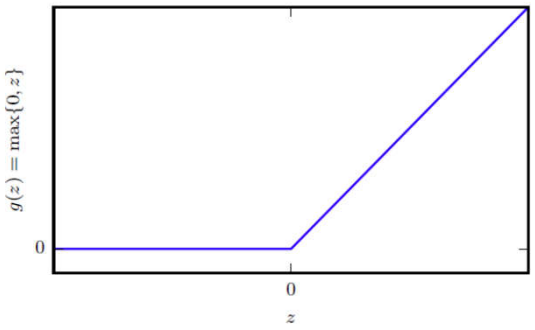
扩展

新的算法

- 反向传播训练

感知机学习（Hebb学习）->Widrow-Hoff学习->反向传播

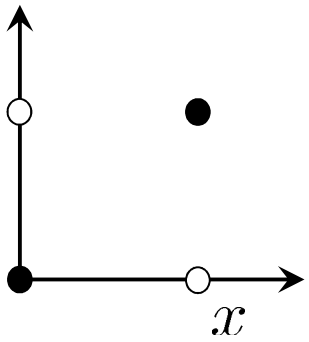
- 现代神经网络中默认推荐的激活函数:ReLu

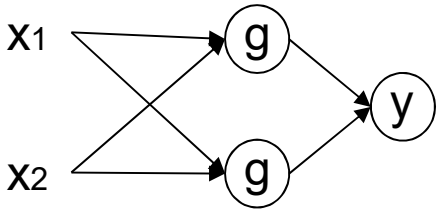


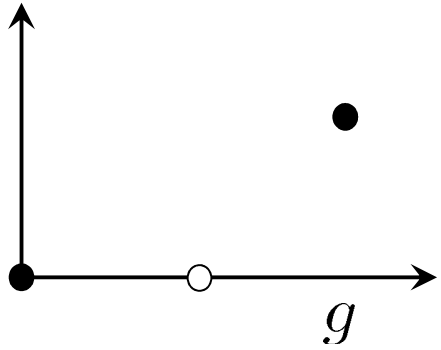
整流线性单元
（rectified linear unit, ReLU）

扩展

XOR学习





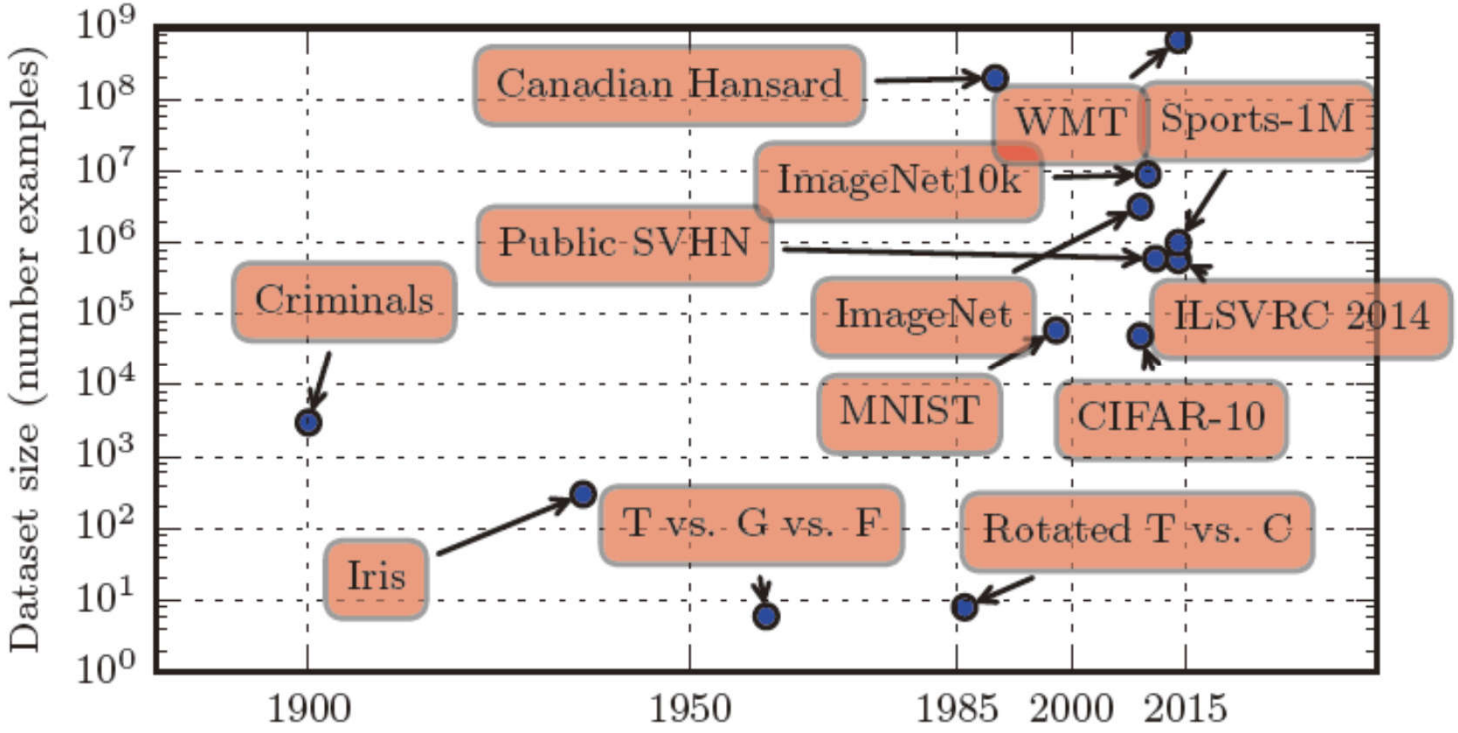


$$\mathrm{f(x;\ W,\ c,\ w,\ b)} = \mathrm{w}^T max\{0, \mathbf{W}^T x + c\} + b$$

$$\mathbf{X} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \quad \mathbf{W} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

扩展

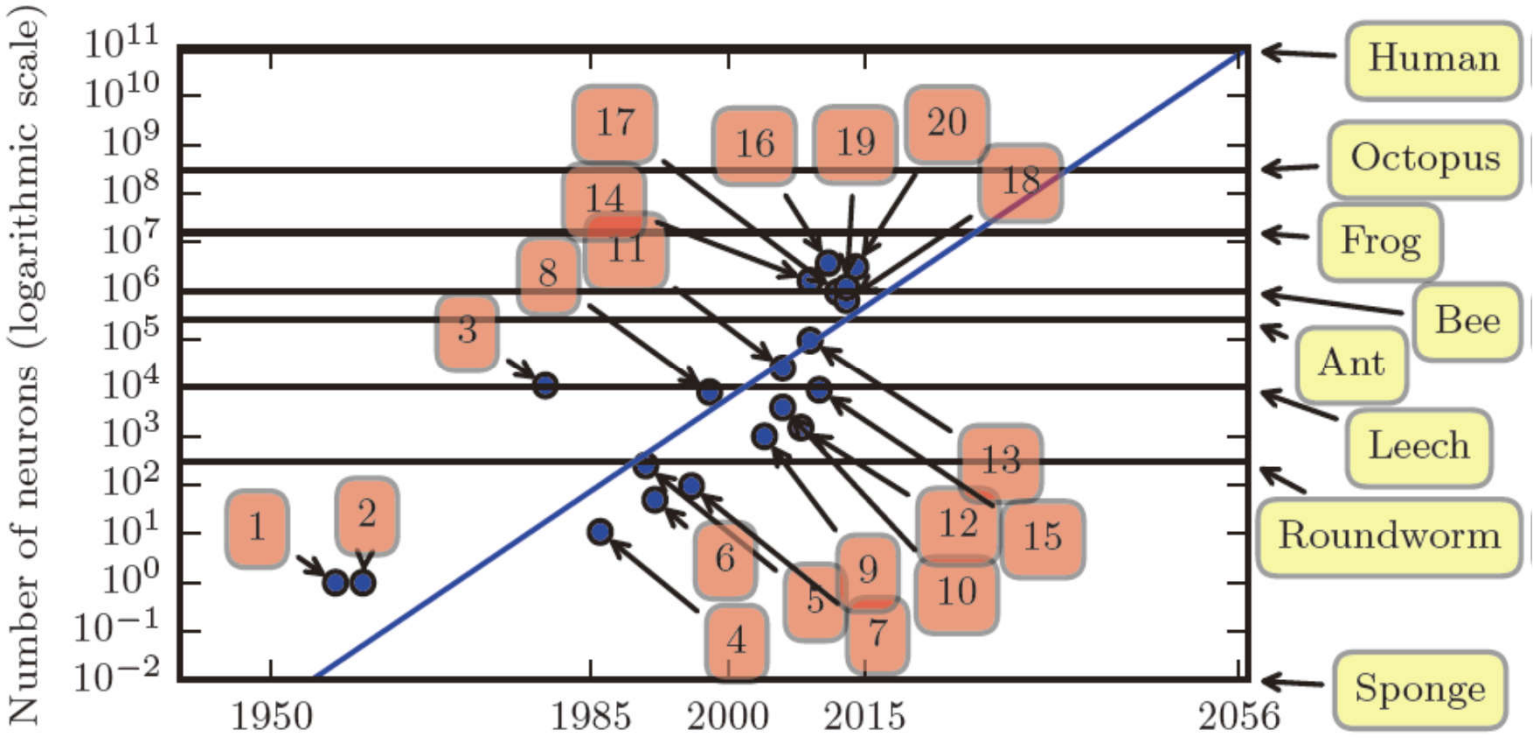
大数据时代使机器学习更加容易



- 样本集的扩展

扩展

计算能力和算法的进步使大规模网络实用化

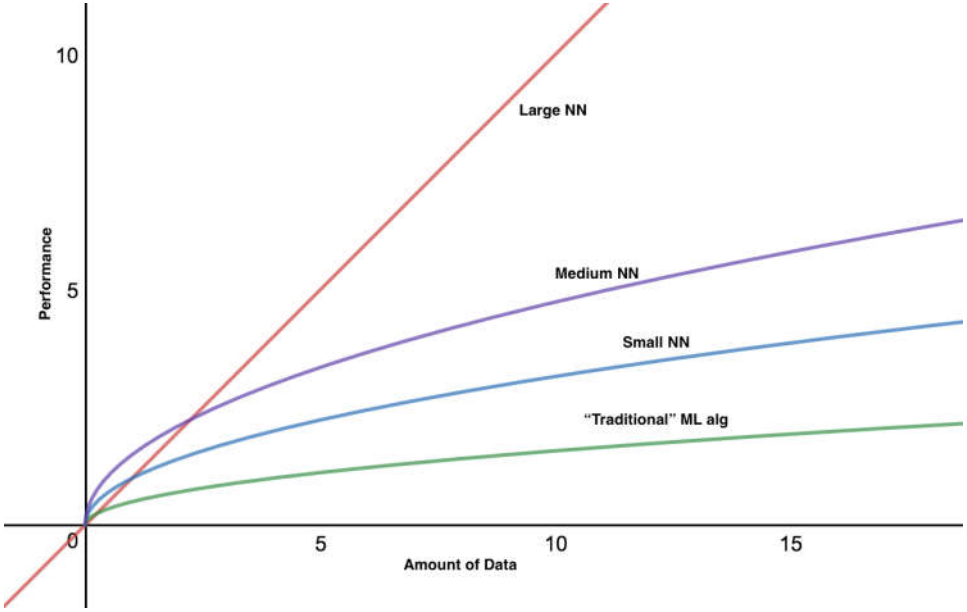


1. 感知机（Rosenblatt, 1958, 1962）

20. GoogleLeNet, Szegedy et al., 2014)

扩展

网络规模扩大和大数据带来明显的性能提升



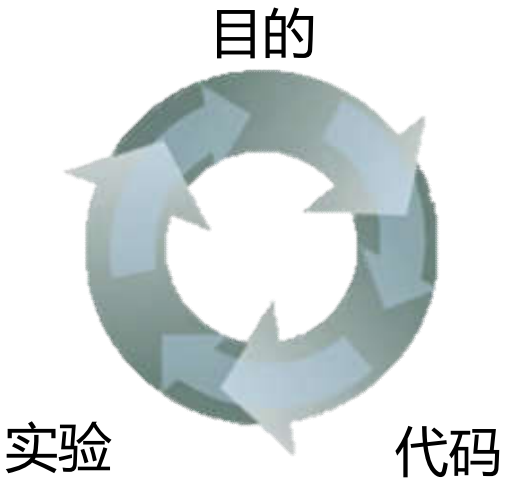
Google大脑

总结

- 感知器
  - 前馈网络
  - 线性的判定边界
  - 每个神经元对应一个决策
- Hamming网络
  - 竞争网络
  - 第一层 – 模式匹配(内积)
  - 第二层 – 竞争 (只余一个大于0的输出)
  - #神经元 = # 标准模式
- Hopfield网络
  - 动态的记忆网络
  - 网络输出收敛于一个标准模式
  - # 神经元 = # 标准模式中的每一个元素

- 增加网络规模和标记数据量可以明显提高性能

- 网络的设计依赖于领域问题和反复调试。



问题?