语音信号处理

深度神经网络在语音处理中的应用

李军锋

中国科学院声学研究所
中科院语言声学与内容理解重点实验室

提纲

- 基本概念
- 梯度下降算法
- BP 神经网络
- 递归神经网络
- 卷积神经网络
- 相关应用

神经网络（Neural Network, NN）

- **生物神经网络**主要是指人脑的神经网络，它是人工神经网络的技术原型。人脑是人类思维的物质基础，思维的功能定位在大脑皮层，后者含有大约1011个神经元，每个神经元又通过神经突触与大约103个其它神经元相连，形成一个高度复杂高度灵活的动态网络。作为一门学科，生物神经网络主要研究人脑神经网络的结构、功能及其工作机制，意在探索人脑思维和智能活动的规律。

- **人工神经网络**是生物神经网络在某种简化意义下的技术复现，作为一门学科，它的主要任务是根据生物神经网络的原理和实际应用的需要建造实用的人工神经网络模型，设计相应的学习算法，模拟人脑的某种智能活动，然后在技术上实现出来用以解决实际问题。因此，生物神经网络主要研究智能的机理；人工神经网络主要研究智能机理的实现，两者相辅相成。

分类问题

- 裙子到底是“白金”还是

NEURAL NETWORKS

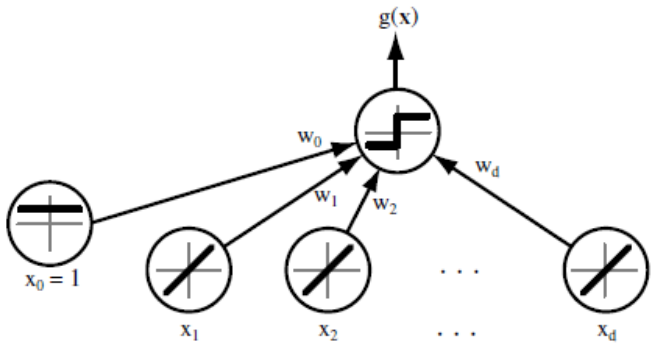
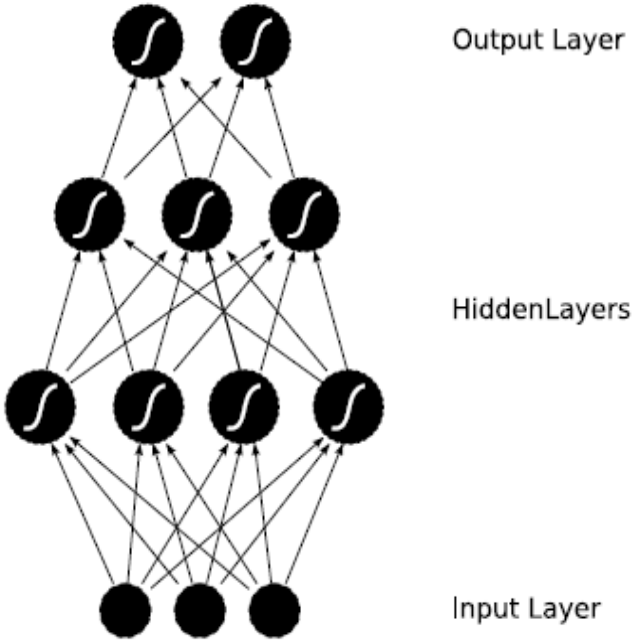
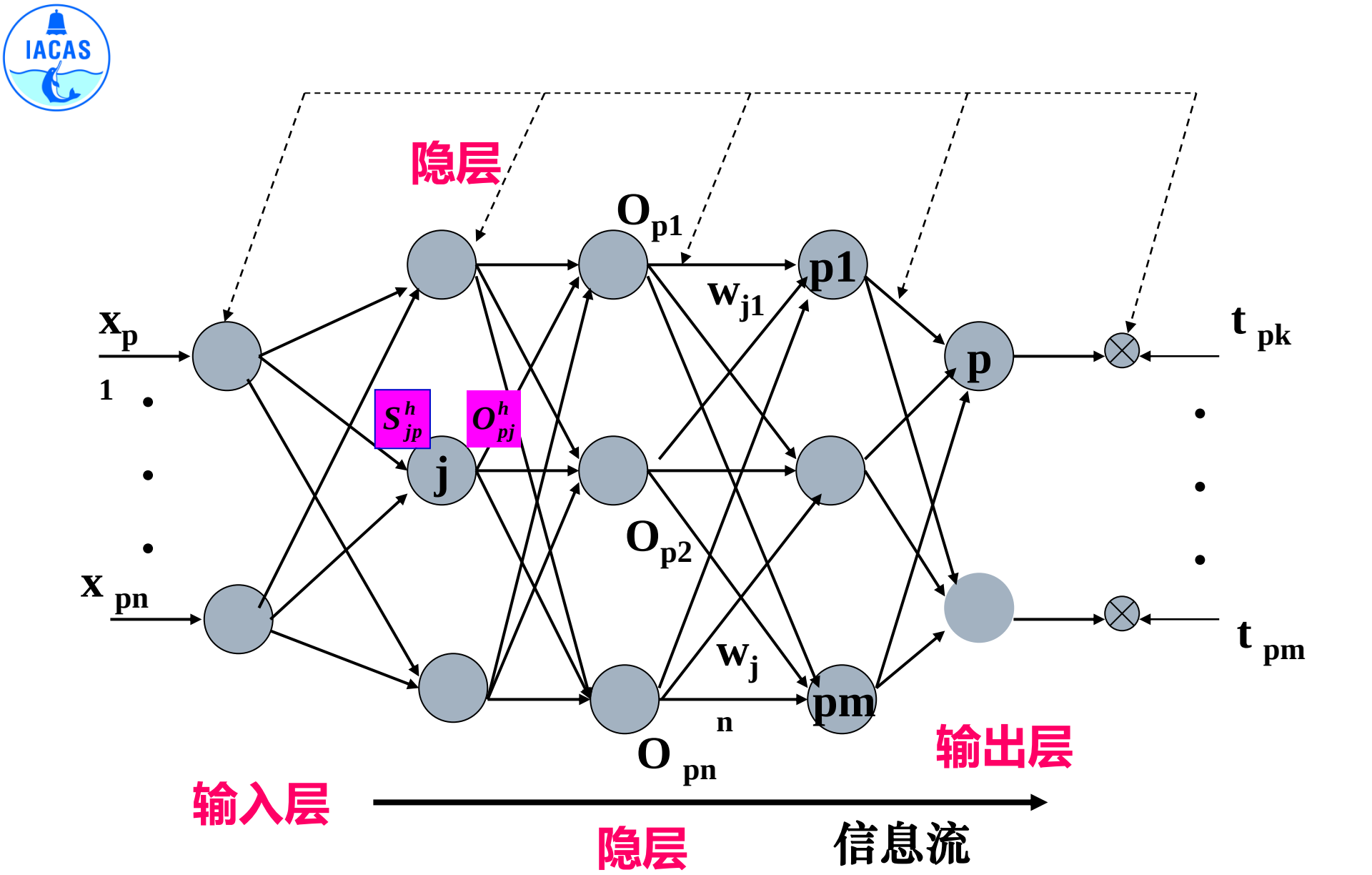


Figure 5.1: A simple linear classifier having d input units, each corresponding to the values of the components of an input vector. Each input feature value x_i is multiplied by its corresponding weight w_i; the output unit sums all these products and emits a $+1$ if $\mathbf{w}^t\mathbf{x} + w_0 > 0$ or a -1 otherwise.




隐层
O_{p1}
w_{j1}
p1
x_p
1
x_{pn}
S^h_{jp}
O^h_{pj}
j
p
t_{pk}
O_{p2}
w_j
n
pm
t_{pm}
O_{pn}
输出层
输入层
隐层
信息流

激活函数：S函数

神经网络构成的基本原理

神经元

神经网络是由大量简单处理单元组成，通过可变权值连接而成的并行分布式系统。神经元是人工神经网络的基本处理单元，它是一个多输入-单输出的非线性器件

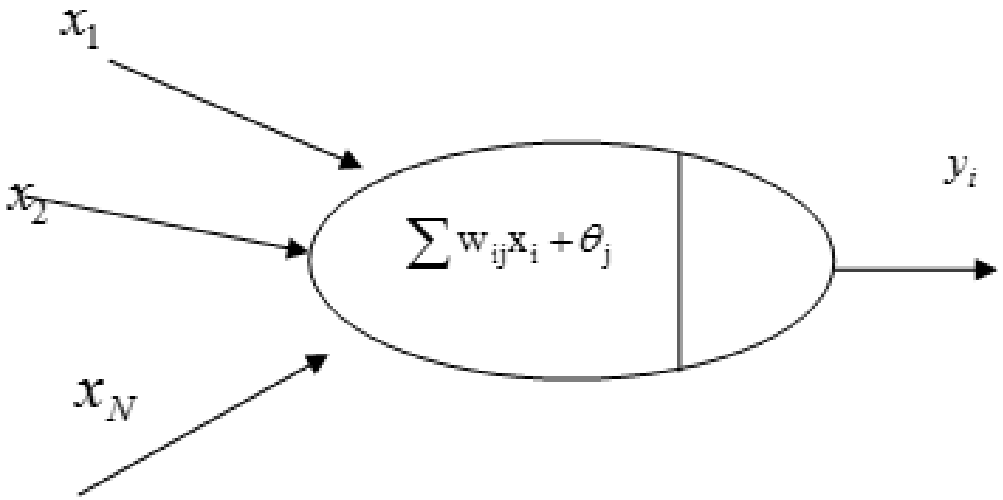


图 2·1 神经元的一般描述

神经网络构成的基本原理

图中，x_i为输入信号，w_{ij}表示从第 i 个神经元到第 j 个神经元的连接权值，θ_j为第 j 个神经元的阈值。设s_j为外部输入信号，y_j为输出信号，在上述模型中第 j 个神经元的变换可描述为

$$y_j = f\left(\sum w_{ij} x_i - \theta_j + s_j\right) \qquad (2\cdot1)$$

这里采用的非线性函数 f(x)可以是阶跃函数、分段函数及 Sigmoid 型函数。

神经网络构成的基本原理

连接权值

人工神经网络的处理单元间相互连接，所有的连接构成一有向图。每一连接对应于一个实数，称为连接权值，或称为权重。权值的集合可看作是长期记忆。我们可以用权矩阵W来表示网络中的连接模式，W中的元素是w_{ij}。连接权值的类型一般分为激发和抑制形式，正的权值表示激发连接，相反，负的权值表示抑制连接。连接权值的连接方式是人工神经网络的特征描述。

神经网络构成的基本原理

神经网络状态

在时刻t，每一个神经元都有一个实数值，称之为神经元状态，也叫做神经元的激励值，用xi表示神经元uj的状态，用X(t)表示神经网络的状态空间。在各种不同的神经网络类型中，状态空间可以作各种不同的假设。状态空间可能是续的，也可能是离散的；可能是有界的，也可能是无界的；可能在一个实数区间上取值，也可能取有限值；最常见的情形是取二值，即0和1两种状态，或-1和1两种状态，亦或是取连续实数值。

神经网络构成的基本原理

神经网络输出

对于每一个神经元，都有一个输出，并通过连接权值将输出传送给其相连的处理单元，输出信号直接依赖于处理单元的状态或激励值。这种依赖性通过输出变换函数f_j对于处理单元u_j的作用来表示。假如我们用$z_j(t)$来定义t时刻神经元的u_i输出那么

$$z_j(t)=f_j(x_j(t)) \qquad (2\cdot 2)$$

或写成向量的形式

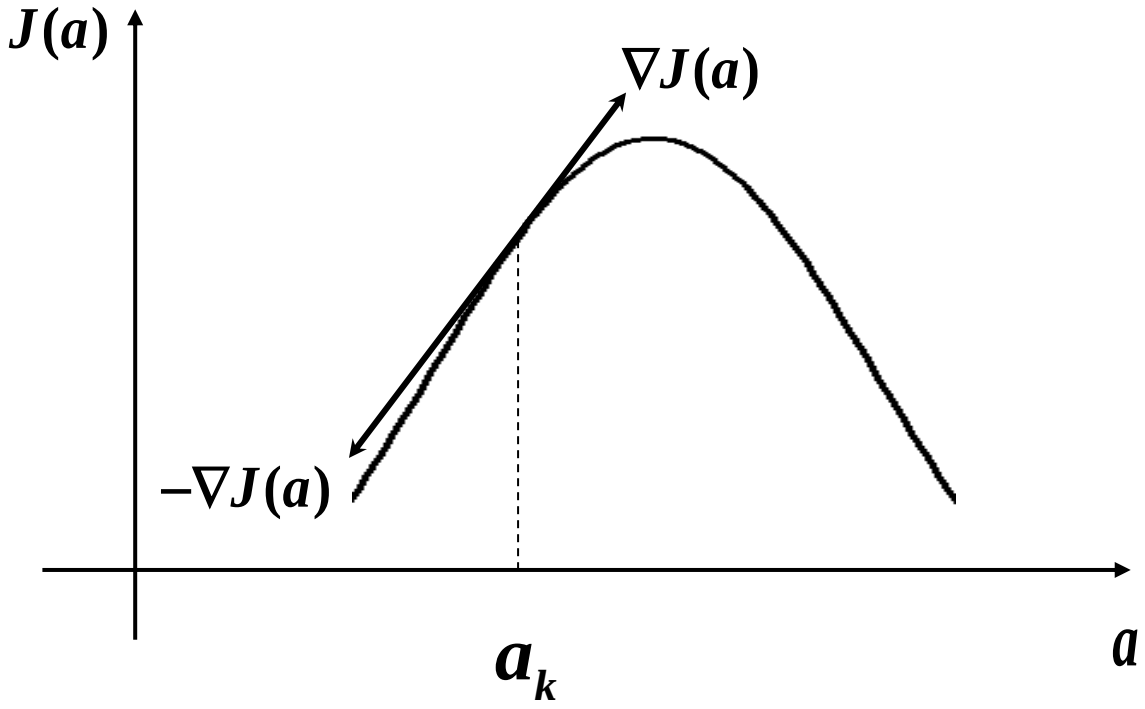
$$Z(t)=f(X(t)) \qquad (2\cdot 3)$$

这里，Z(t)是神经网络的输出向量，f定义为状态向量与每一个分量的对应函数。一般是在区间（0，1）上的有界函数。

梯度下降算法

梯度下降法又称最速下降法。函数$J(a)$在某点a_k的梯度$\nabla J(a_k)$是一个向量，其方向是$J(a)$增长最快的方向。显然，负梯度方向是$J(a)$减少最快的方向。

在梯度下降法中，求某函数极大值时，沿着梯度方向走，可以最快达到极大点；反之，沿着负梯度方向走，则最快地达到极小点。


J(a)
∇J(a)
−∇J(a)
a_k
a

求函数$J(a)$极小值的问题，可以选择任意初始点a_0,从a_0出发沿着负梯度方向走，可使得$J(a)$下降最快。

$$s^{(0)} = -\nabla J(a_0)$$

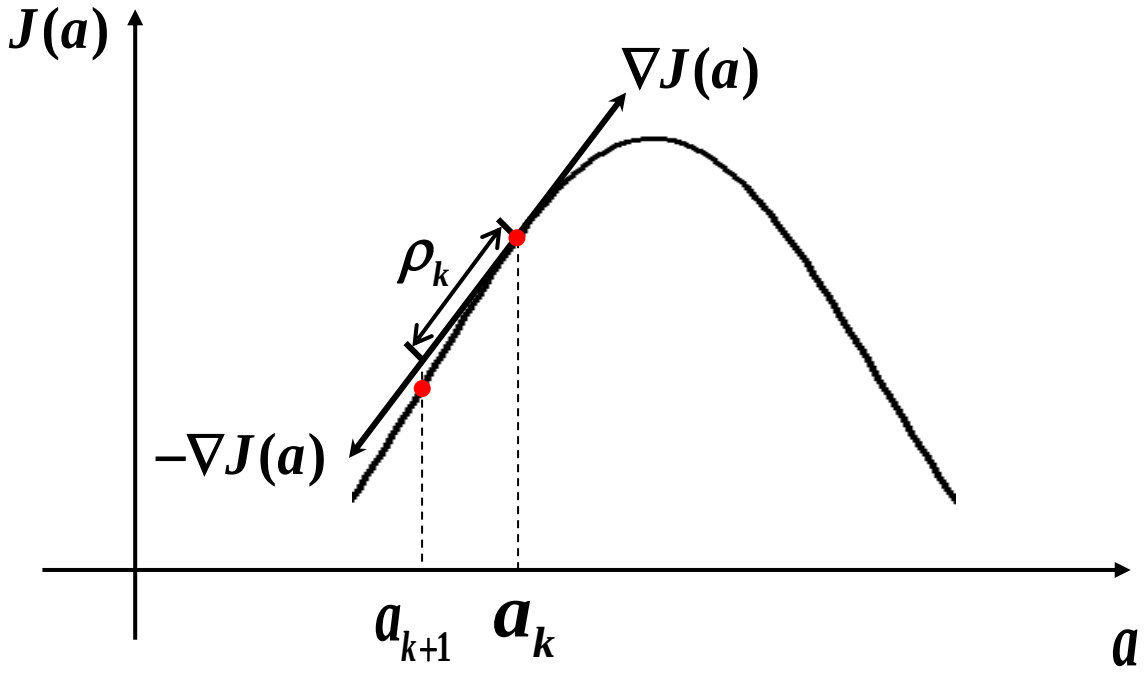
$s^{(0)}$：点a_0的搜索方向。

对于任意点a_k，可以定义a_k点的负梯度搜索方向的单位向量为:

$$\hat{s}^{(k)} = -\frac{\nabla J(a_k)}{\|\nabla J(a_k)\|}$$

从a_k点出发，沿着 $\hat{s}^{(k)}$方向走一步，步长为 ρ_k ,得到新点a_{k+1},表示为:

$$a_{k+1} = a_k + \rho_k \hat{s}^{(k)}$$


J(a)
∇J(a)
ρ_k
−∇J(a)
a_{k+1}
a_k
a

因此，在新点a_{k+1}，函数$J(a)$的函数值为：

$$J(a_{k+1}) = J(a_k + \rho_k \hat{s}^{(k)})$$

所有的a_k组成一个序列，该序列由迭代算法生成

$$a_0, a_1, a_2, \ldots, a_k, a_{k+1}, \ldots$$

该序列在一定条件下收敛于使得$J(a)$最小的解a^*

迭代算法公式：$a_{k+1} = a_k + \rho_k \hat{s}^{(k)}$

迭代算法公式：

$$\boldsymbol{a}_{k+1} = \boldsymbol{a}_k + \rho_k \hat{\boldsymbol{s}}^{(k)}$$

关键问题：如何设计步长 ρ_k

如果选得太小，则算法收敛慢，如果选得太大，可能会导致发散。

梯度法的迭代过程：

1) 选取初始点a_0，给定允许误差$\alpha>0,\beta>0$,并令$k=0$。

2) 计算负梯度 $s^{(k)}=-\nabla J(a_k)$ 及其单位向量 $\hat{s}^{(k)}$ 。

3) 检查是否满足条件$\left\|s^{(k)}\right\|\le\alpha$，若满足则转8，否则继续。

4) 计算最佳步长 ρ_k^* 。

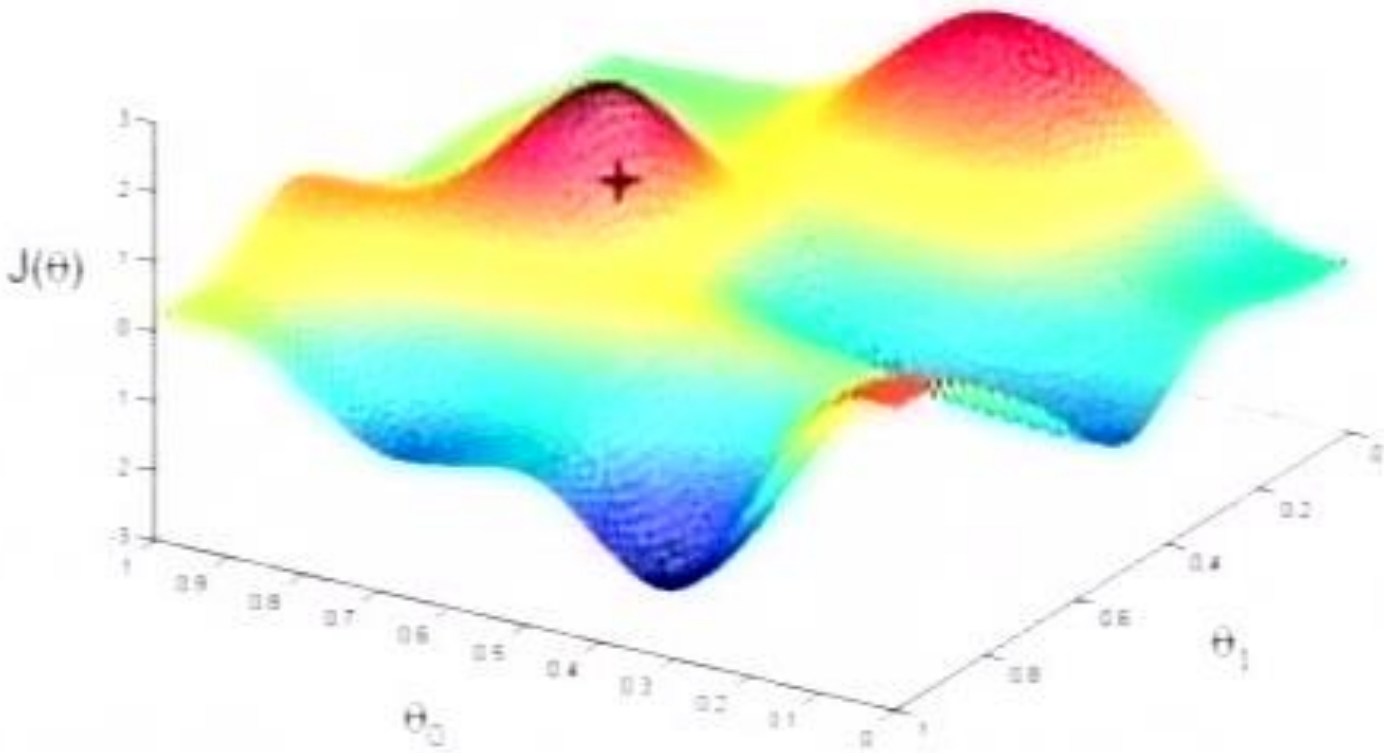
5) 令：$a_{k+1}=a_k+\rho_k^*\hat{s}^{(k)}$

6) 计算并检验另一判据：$J(a_{k+1})-J(a_k)\le\beta$，满足转8，否则继续。

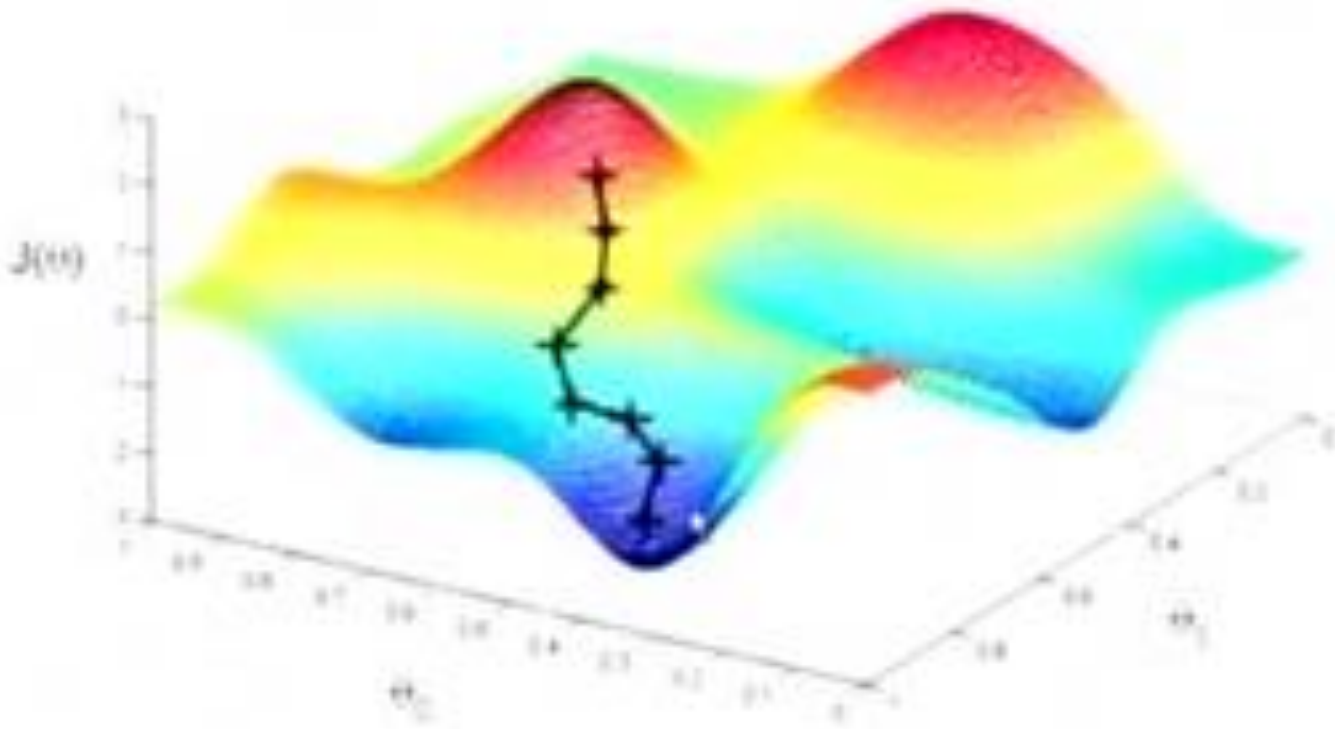
7) 令$k=k+1$,转2。

8) 输出结果，结束。

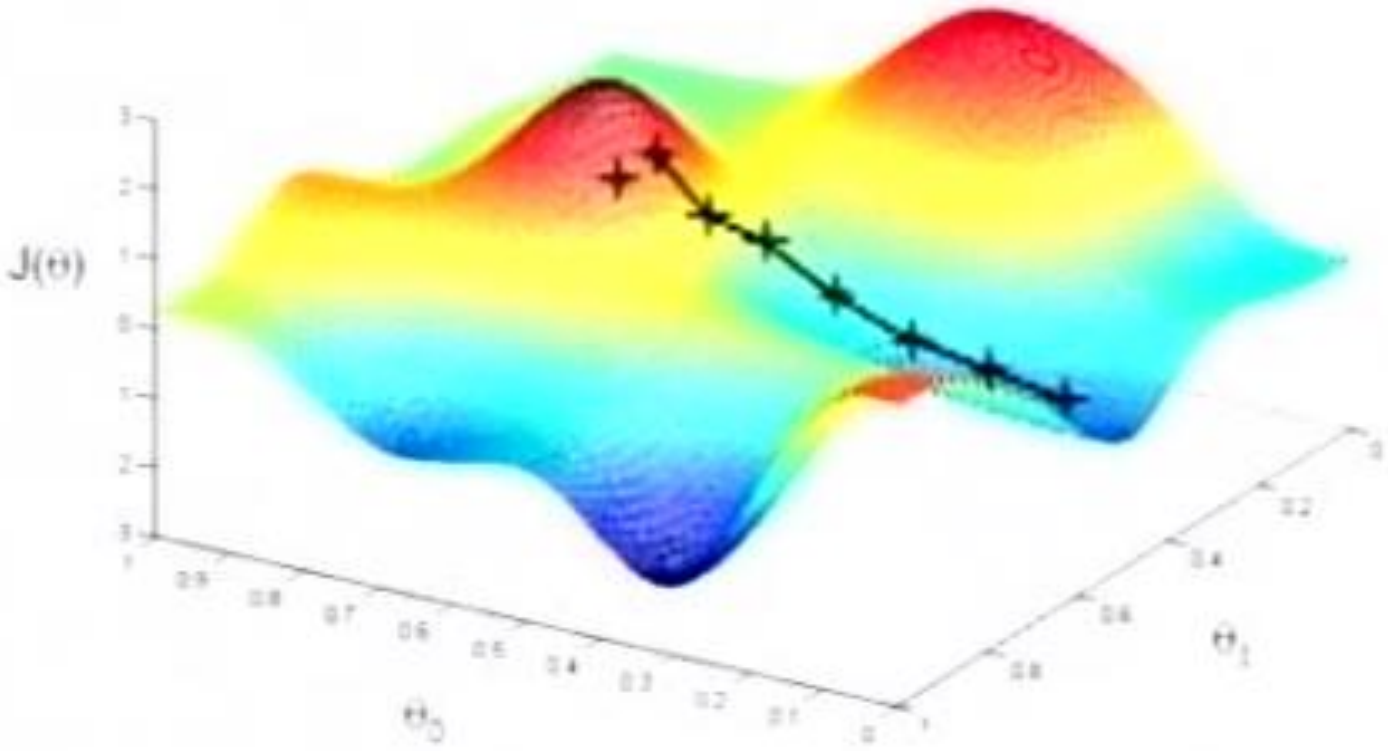
Gradient Descent



Gradient Descent



Gradient Descent



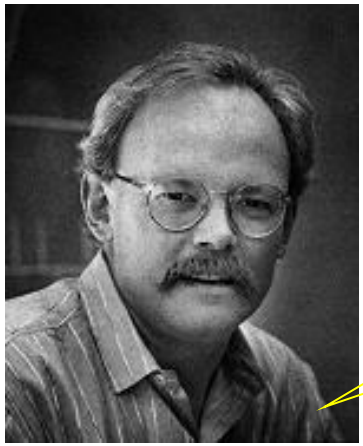
BP网络

概述

Back—Propagation Network，由于其权值的调整采用反向传播（Backpropagation）的学习算法，因此被称为BP网络。

- Rumelhart，McClelland于1985年提出了BP网络的误差反向后传BP(Back Propagation)学习算法

David Rumelhart

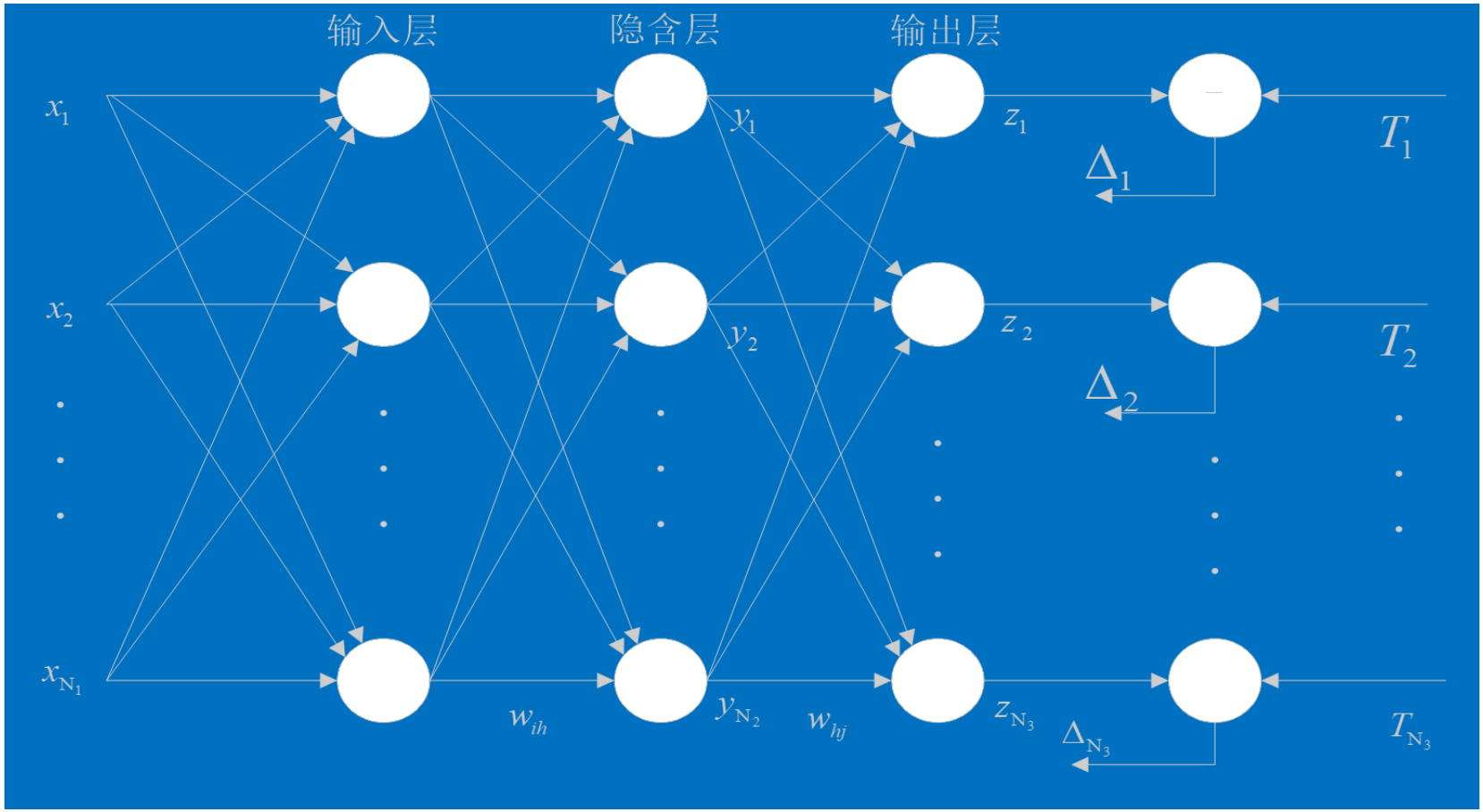


J. McClelland

BP算法基本原理

- 利用输出后的误差来估计输出层的直接前导层的误差，再用这个误差估计更前一层的误差，如此一层一层的反传下去，就获得了所有其他各层的误差估计。
- 是一种单向传播的多层前向网络
- 其神经元的变换函数是S型函数，因此输出量为0到1之间的连续量
- 它可以对非线性可微分函数进行权值训练，从而实现输入到输出的任意的非线性映射。
- 网络中心思想是梯度下降法通过梯度搜索技术，使网络实际输出值与期望输出值的误差均方值最小。
- 网络的学习过程是一种误差边向后传播边修正权系数的过程

三层BP网络


输入层
隐含层
输出层
x_1
x_2
x_{N_1}
y_1
y_2
y_{N_2}
z_1
z_2
z_{N_3}
w_{ih}
w_{hj}
Δ_1
Δ_2
Δ_{N_3}
T_1
T_2
T_{N_3}

- 激活函数
  - 必须处处可导
    - 一般都使用S型函数
- 使用S型激活函数时BP网络输入与输出关系
  - 输入

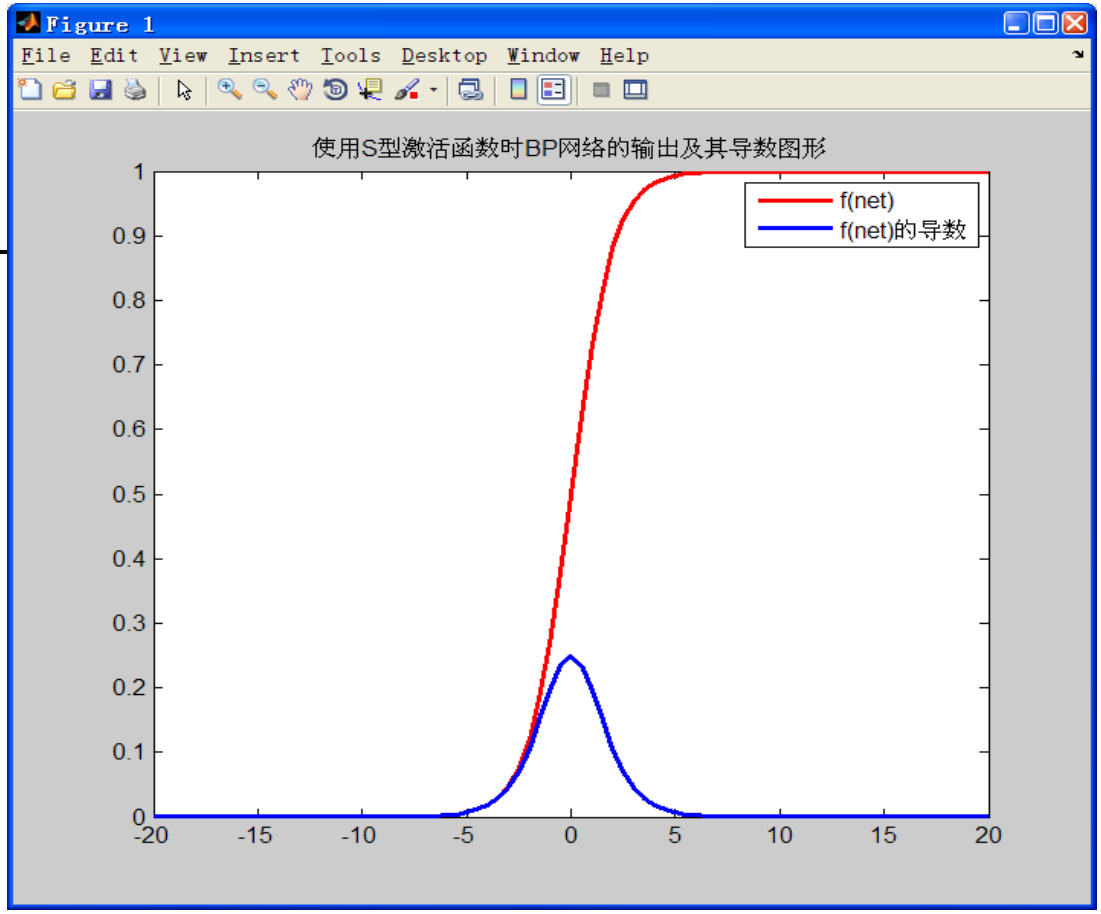
$$net = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n$$

  - 输出

$$y = \mathrm{f}(net) = \frac{1}{1 + \mathrm{e}^{-net}}$$

- 输出的导数

$$f'(net) = \frac{1}{1+e^{-net}} -$$



根据S型激活函数的图形可知，对神经网络进行训练，应该将net的值尽量控制在收敛比较快的范围内

- 学习的过程：
  - 神经网络在外界输入样本的刺激下不断改变网络的连接权值,以使网络的输出不断地接近期望的输出。
- 学习的本质：
  - 对各连接权值的动态调整
- 学习规则：
  - 权值调整规则，即在学习过程中网络中各神经元的连接权变化所依据的一定的调整规则。

BP算法是由两部分组成：信息的正向传递与误差的反向传播。

在正向传播过程中，输入信息从输入经隐含层逐层计算传向输出层，每一层神经元的状态只影响下一层神经元的状态。

如果在输出层没有得到期望的输出，则计算输出层的误差变化值，然后转向反向传播，通过网络将误差信号沿原来的连接通路反传回来修改各层神经元的权值直至达到期望目标。

算法思想

- 学习的类型：有导师学习
- 核心思想：
  - 将输出误差*以某种形式*通过隐层向输入层逐层反传

将误差分摊给各层的所有单元 - - - 各层单元的误差信号 → 修正各单元权值

- 学习的过程：
  - 信号的正向传播 → 误差的反向传播

学习过程

- 正向传播:
  - 输入样本－－－输入层－－－各隐层－－－输出层
- 判断是否转入反向传播阶段:
  - 若输出层的实际输出与期望的输出（教师信号）不符
- 误差反传
  - 误差以某种形式在各层表示－－－－修正各层单元的权值

- 网络输出的误差减少到可接受的程度
  进行到预先设定的学习次数为止

- 网络结构
  - 输入层有n个神经元，隐含层有p个神经元，输出层有q个神经元
- 变量定义
  - 输入向量; $x=(x_1,x_2,\cdots,x_n)$
  - 隐含层输入向量; $hi=(hi_1,hi_2,\cdots,hi_p)$
  - 隐含层输出向量; $ho=(ho_1,ho_2,\cdots,ho_p)$
  - 输出层输入向量; $yi=(yi_1,yi_2,\cdots,yi_q)$
  - 输出层输出向量; $yo=(yo_1,yo_2,\cdots,yo_q)$
  - 期望输出向量; $d_o=(d_1,d_2,\cdots,d_q)$

- 输入层与中间层的连接权值: w_{ih}
- 隐含层与输出层的连接权值: w_{ho}
- 隐含层各神经元的阈值: b_h
- 输出层各神经元的阈值: b_o
- 样本数据个数: $k = 1, 2, \cdots m$
- 激活函数: $\mathrm{f}(\cdot)$
- 误差函数: $e = \frac{1}{2}\sum_{o=1}^{q}(d_o(k) - yo_o(k))^2$

第一步，网络初始化

给各连接权值分别赋一个区间（-1，1）内的随机数，设定误差函数e，给定计算精度值ε和最大学习次数M。

第二步,随机选取第k个输入样本及对应期望输出

$$x(k)=\left(x_1(k),x_2(k),\cdots,x_n(k)\right)$$

$$d_o(k)=\left(d_1(k),d_2(k),\cdots,d_q(k)\right)$$

第三步，计算隐含层各神经元的输入和输出

$$hi_h(k) = \sum_{i=1}^{n} w_{ih} x_i(k) - b_h \qquad h = 1, 2, \cdots, p$$

$$ho_h(k) = \mathrm{f}(hi_h(k)) \qquad h = 1, 2, \cdots, p$$

$$yi_o(k) = \sum_{h=1}^{p} w_{ho} ho_h(k) - b_o \qquad o = 1, 2, \cdots q$$

$$yo_o(k) = \mathrm{f}(yi_o(k)) \qquad o = 1, 2, \cdots q$$

第四步，利用网络期望输出和实际输出，计算误差函数对输出层的各神经元的偏导数$\delta_o(k)$。

$$\frac{\partial e}{\partial w_{ho}} = \frac{\partial e}{\partial yi_o}\frac{\partial yi_o}{\partial w_{ho}} \qquad \frac{\partial yi_o(k)}{\partial w_{ho}} = \frac{\partial(\sum_h^p w_{ho} ho_h(k) - b_o)}{\partial w_{ho}} = ho_h(k)$$

$$\frac{\partial e}{\partial yi_o} = \frac{\partial(\frac{1}{2}\sum_{o=1}^{q}(d_o(k) - yo_o(k)))^2}{\partial yi_o} = -(d_o(k) - yo_o(k))yo_o'(k)$$

$$= -(d_o(k) - yo_o(k))\mathrm{f}'(yi_o(k)) = -\delta_o(k)$$

第五步，利用隐含层到输出层的连接权值、输出层的$\delta_o(k)$和隐含层的输出计算误差函数对隐含层各神经元的偏导数$\delta_h(k)$。

$$\frac{\partial e}{\partial w_{ho}} = \frac{\partial e}{\partial yi_o}\frac{\partial yi_o}{\partial w_{ho}} = -\delta_o(k)ho_h(k)$$

$$\frac{\partial e}{\partial w_{ih}} = \frac{\partial e}{\partial hi_h(k)}\frac{\partial hi_h(k)}{\partial w_{ih}}$$

$$\frac{\partial hi_h(k)}{\partial w_{ih}} = \frac{\partial(\sum_{i=1}^{n} w_{ih}x_i(k) - b_h)}{\partial w_{ih}} = x_i(k)$$

$$\frac{\partial e}{\partial hi_h(k)} = \frac{\partial(\frac{1}{2}\sum_{o=1}^{q}(d_o(k)-yo_o(k))^2)}{\partial ho_h(k)}\frac{\partial ho_h(k)}{\partial hi_h(k)}$$

$$= \frac{\partial(\frac{1}{2}\sum_{o=1}^{q}(d_o(k)-\mathrm{f}(yi_o(k)))^2)}{\partial ho_h(k)}\frac{\partial ho_h(k)}{\partial hi_h(k)}$$

$$= \frac{\partial(\frac{1}{2}\sum_{o=1}^{q}((d_o(k)-\mathrm{f}(\sum_{h=1}^{p}w_{ho}ho_h(k)-b_o)^2))}{\partial ho_h(k)}\frac{\partial ho_h(k)}{\partial hi_h(k)}$$

$$= -\sum_{o=1}^{q}(d_o(k)-yo_o(k))\,\mathrm{f}'(yi_o(k))w_{ho}\frac{\partial ho_h(k)}{\partial hi_h(k)}$$

$$= -(\sum_{o=1}^{q}\delta_o(k)w_{ho})\,\mathrm{f}'(hi_h(k)) = -\delta_h(k)$$

第六步，利用输出层各神经元的 $\delta_o(k)$ 和隐含层各神经元的输出来修正连接权值 $w_{ho}(k)$。

$$\Delta w_{ho}(k) = -\mu \frac{\partial e}{\partial w_{ho}} = \mu \delta_o(k) ho_h(k)$$

$$w_{ho}^{N+1} = w_{ho}^{N} + \eta \delta_o(k) ho_h(k)$$

第七步，利用隐含层各神经元的$\delta_h(k)$和输入层各神经元的输入修正连接权。

$$\Delta w_{ih}(k) = -\mu \frac{\partial e}{\partial w_{ih}} = -\mu \frac{\partial e}{\partial hi_h(k)} \frac{\partial hi_h(k)}{\partial w_{ih}} = \mu \delta_h(k) x_i(k)$$

$$w_{ih}^{N+1} = w_{ih}^{N} + \eta \delta_h(k) x_i(k)$$

第八步，计算全局误差

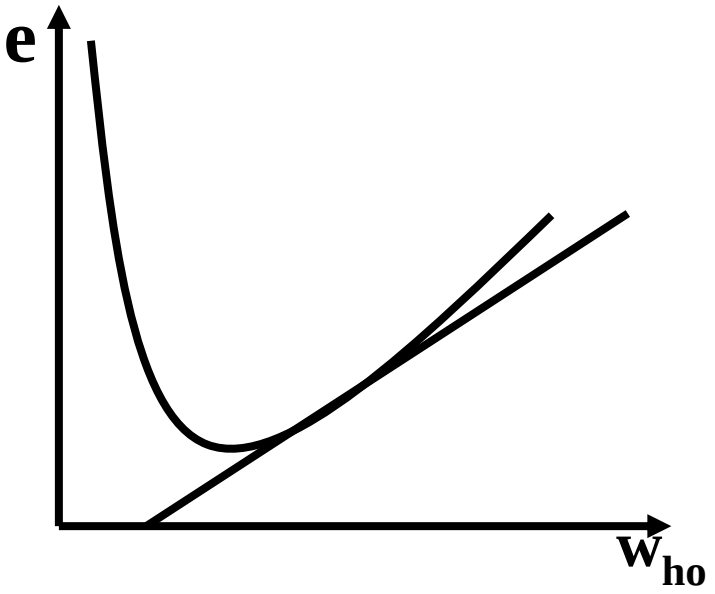
$$E=\frac{1}{2m}\sum_{k=1}^{m}\sum_{o=1}^{q}(d_o(k)-y_o(k))^2$$

第九步，判断网络误差是否满足要求。当误差达到预设精度或学习次数大于设定的最大次数，则结束算法。否则，选取下一个学习样本及对应的期望输出，返回到第三步，进入下一轮学习。

BP算法直观解释

情况一直观表达

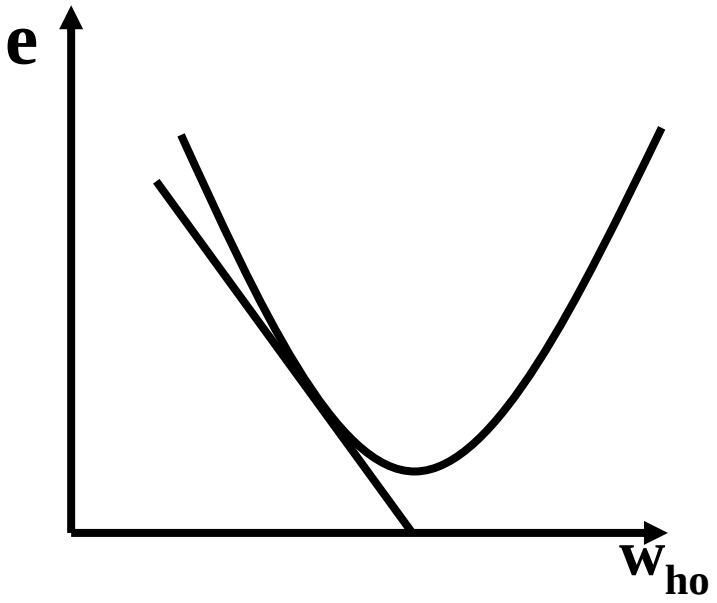
当误差对权值的偏导数大于零时，权值调整量为负，实际输出大于期望输出，权值向减少方向调整，使得实际输出与期望输出的差减少。



>0，此时$\Delta w_{ho}<0$

BP算法直解释

- 情况二直观表达
  - 当误差对权值的偏导数小于零时，权值调整量为正，实际输出少于期望输出，权值向增大方向调整，使得实际输出与期望输出的差减少。



<0, 此时$\Delta w_{ho}>0$

BP网络结构特点：

1. BP网络具有一层或多层隐含层，与其他网络模型除了结构不同外，主要差别表现在激活函数上。
2. BP网络的激活函数必须是处处可微的，所以它就不能采用二值型的阈值函数{0，1}或符号函数{—1，1}，BP网络经常使用的是S型的对数或正切激活函数和线性函数。

3、只有当希望对网络的输出进行限制，如限制在0和1之间，那么在输出层应当包含S型激活函数，在一般情况下，均是在隐含层采用S型激活函数，而输出层采用线性激活函数。

4、输入和输出是并行的模拟量；

5、网络的输入输出关系是各层连接的权因子决定，没有固定的算法；

6、权因子是通过学习信号调节的，这样学习越多，网络越聪明；

7、隐含层越多，网络输出精度越高，且个别权因子的损坏不会对网络输出产生大的影响

BP网络的设计

1 网络的层数:

理论上已经证明：具有偏差和至少一个S型隐含层加上一个线性输出层的网络，能够逼近任何有理函数。

增加层数主要可以更进一步的降低误差，提高精度，但同时也使网络复杂化，从而增加了网络权值的训练时间。

一般情况下，应优先考虑增加隐含层中的神经元数。

能不能仅用具有非线性激活函数的单层网络来解决问题呢？结论是：没有必要或效果不好。

2 隐含层的神经元数

网络训练精度的提高，可以通过采用一个隐含层，而增加其神经元数的方法来获得。这在结构实现上，要比增加更多的隐含层要简单得多。

在具体设计时，比较实际的做法是通过对不同神经元数进行训练对比，然后适当地加上一点余量。

3、初始权值的选取

一般取初始权值在(−1，1)之间的随机数。

由于每次训练时都对权值进行随机初始化，所以每次训练得到的网络权值都是不一样的。

4 学习速率

学习速率决定每一次循环训练中所产生的权值变化量。

大的学习速率可能导致系统的不稳定。

小的学习速率导致较长的训练时间，可能收敛很慢，不过能保证网络的误差值不跳出误差表面的低谷而最终趋于最小误差值。

所以在一般情况下，倾向于选取较小的学习速率以保证系统的稳定性。学习速率的选取范围在0．01—0．8之间。

5　期望误差的选取

在设计网络的训练过程中，期望误差值也应当通过对比训练后确定一个合适的值。

这个所谓的“合适”，是相对于所需要的隐含层的节点数来确定，因为较小的期望误差值是要靠增加隐含层的节点，以及训练时间来获得的。

一般情况下，作为对比，可以同时对两个不同期望误差值的网络进行训练，最后通过综合因素的考虑来确定采用其中一个网络。

BP网络用途

1)函数逼近：用输入矢量和相应的输出矢量训练一个网络逼近一个函数；

2)模式识别：用一个特定的输出矢量将它与输入矢量联系起来；

3)分类：把输入矢量以所定义的合适方式进行分类；

4)数据压缩：减少输出矢量维数以便于传输或存储。

BP网络的局限与不足

(1)需要较长的训练时间

因为涉及到求导的运算，需要的时间较长

(2)训练瘫痪问题

通常为了避免这种现象的发生，一是选取较小的初始权值，二是采用较小的学习速率，但这又增加了训练时间。

小结

1)反向传播法可以用来训练具有可微激活函数的多层前向网络以进行函数逼近，模式分类等工作；

2)反向传播网络的结构不完全受所要解决的问题所限制。网络的输入神经元数目及输出层神经元的数目是由问题的要求所决定的，而输入和输出层之间的隐含层数以及每层的神经元数是由设计者来决定的；

3)已证明，两层S型线性网络，如果S型层有足够的神经元，则能够训练出任意输入和输出之间的有理函数关系；

4)反向传播法沿着误差表面的梯度下降，使网络误差最小，网络有可能陷入局部极小值；

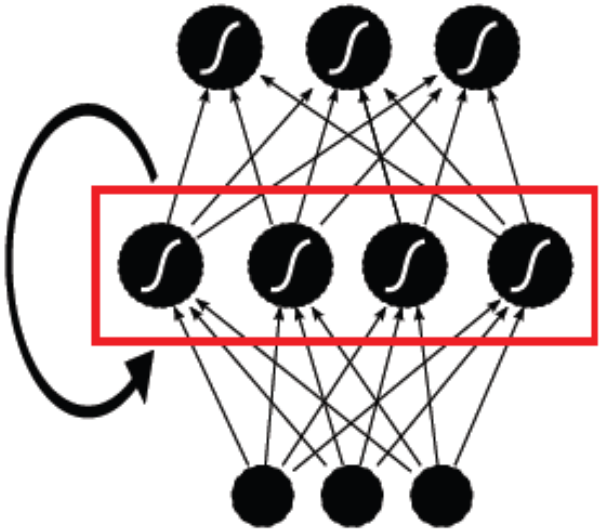
5)太大的学习速率导致学习的不稳定，太小值又导致极长的训练时间。自适应学习速率通过在保证稳定训练的前提下，达到了合理的高速率，可以减少训练时间；

6)80％—90％的实际应用都是采用反向传播网络的。改进技术可以用来使反向传播法更加容易实现并需要更少的训练时间。

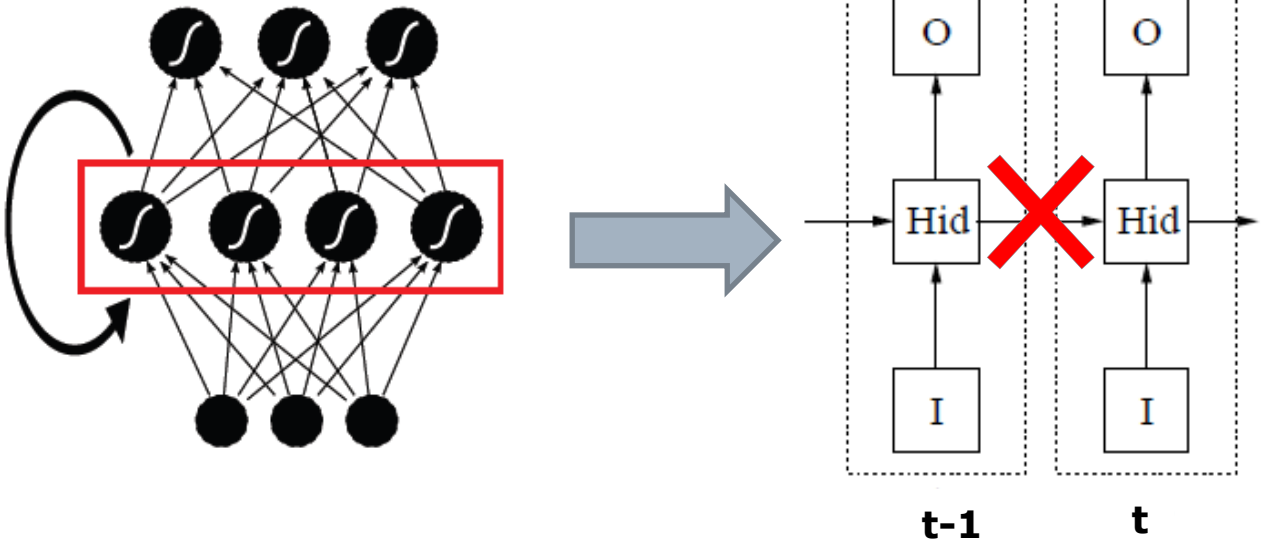
递归神经网络（RNN）

Recurrent Neural Network (RNN)

- FNN仅能建立*当前*输入特征到输出目标的映射关系
- RNN能够建立整个*历史*输入特征到输出目标的映射关系（理论上）
- 关键在于RNN有对历史的“记忆功能”，从而影响网络的输出



RNN （FW）



- 结点激励公式：

$$a_h^t = \sum_{i=1}^{I} w_{ih} x_i^t + \sum_{h'=1}^{H} w_{h'h} b_{h'}^{t-1} \qquad b_h^t = \theta_h(a_h^t)$$

- 迭代计算：

$$b_i^0 = 0$$

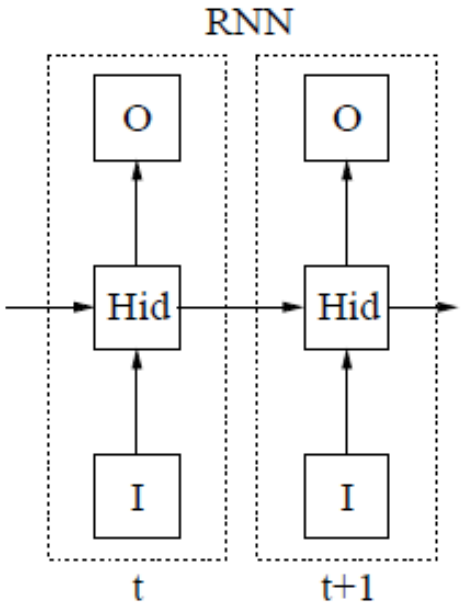
$$t = 1, 2 \ldots\ldots T$$

RNN （BW）

$$\delta_h^t = \theta'(a_h^t)\left(\sum_{k=1}^{K}\delta_k^t w_{hk} + \sum_{h'=1}^{H}\delta_{h'}^{t+1} w_{hh'}\right)$$

where

$$\delta_j^t \overset{\text{def}}{=} \frac{\partial O}{\partial a_j^t}$$



- 整句话：

$$\frac{\partial O}{\partial w_{ij}} = \sum_{t=1}^{T}\frac{\partial O}{\partial a_j^t}\frac{\partial a_j^t}{\partial w_{ij}} = \sum_{t=1}^{T}\delta_j^t b_i^t$$

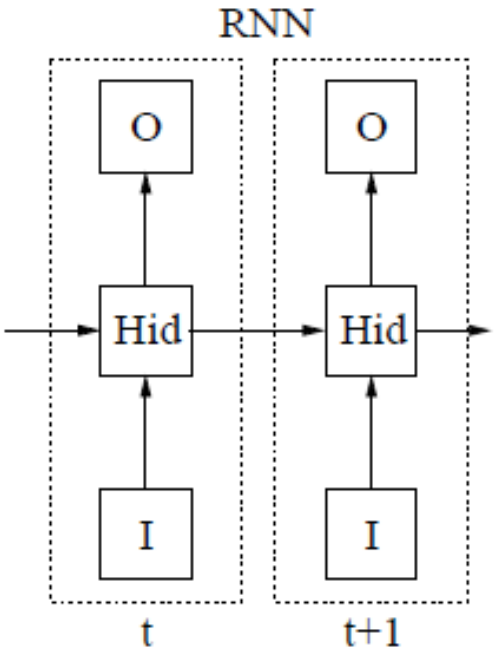
- 迭代计算：

$$\delta_j^{T+1} = 0 \forall j$$

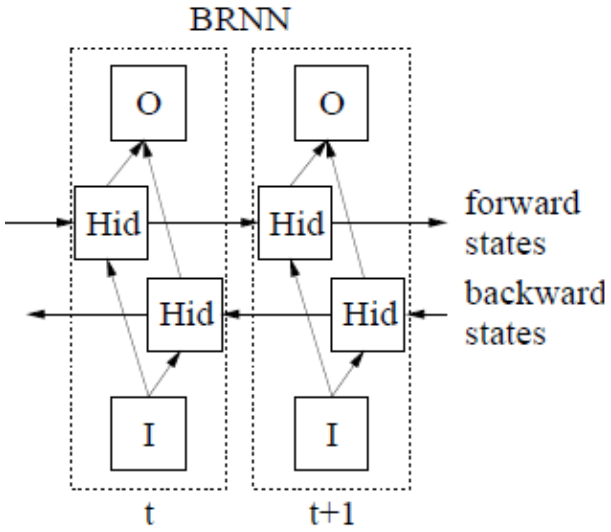
$$t = T, T-1 \ldots\ldots 0$$

Bidirectional RNN

- RNN只考虑历史，没有考虑未来

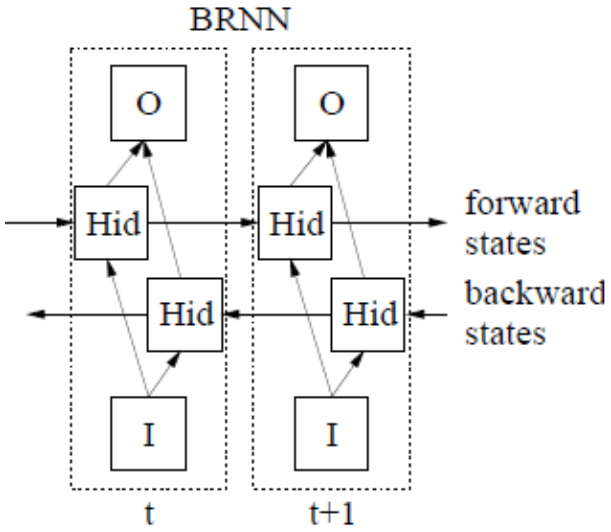


Bidirectional RNN （FW）



for $t = 1$ to T **do**
Do forward pass for the forward hidden layer, storing activations at each timestep
for $t = T$ to 1 **do**
Do forward pass for the backward hidden layer, storing activations at each timestep
for $t = 1$ to T **do**
Do forward pass for the output layer, using the stored activations from both hidden layers

Bidirectional RNN （BW）

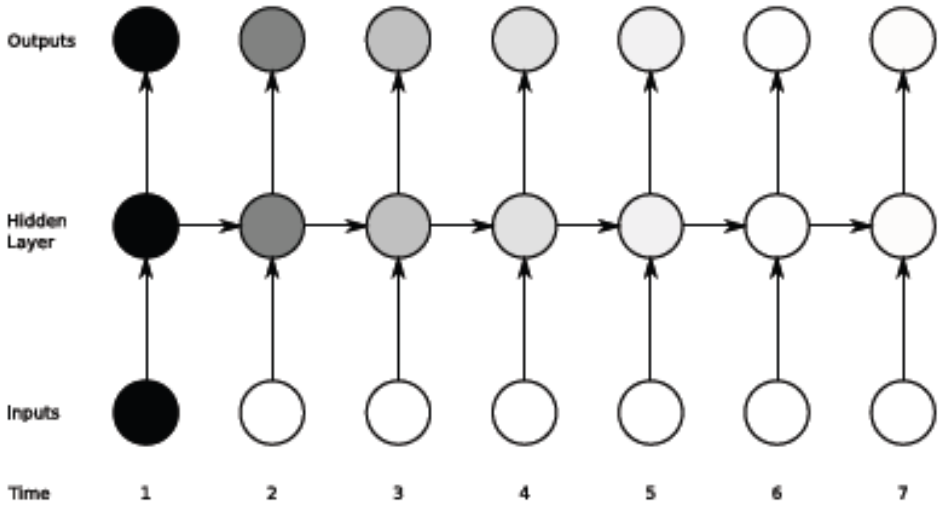


for $t = T$ to 1 **do**
Do BPTT backward pass for the output layer only, storing δ terms at each timestep
for $t = T$ to 1 **do**
Do BPTT backward pass for the forward hidden layer, using the stored δ terms from the output layer
for $t = 1$ to T **do**
Do BPTT backward pass for the backward hidden layer, using the stored δ terms from the output layer

长短时记忆网络（LSTM）

Long Short-Term Memory (LSTM)

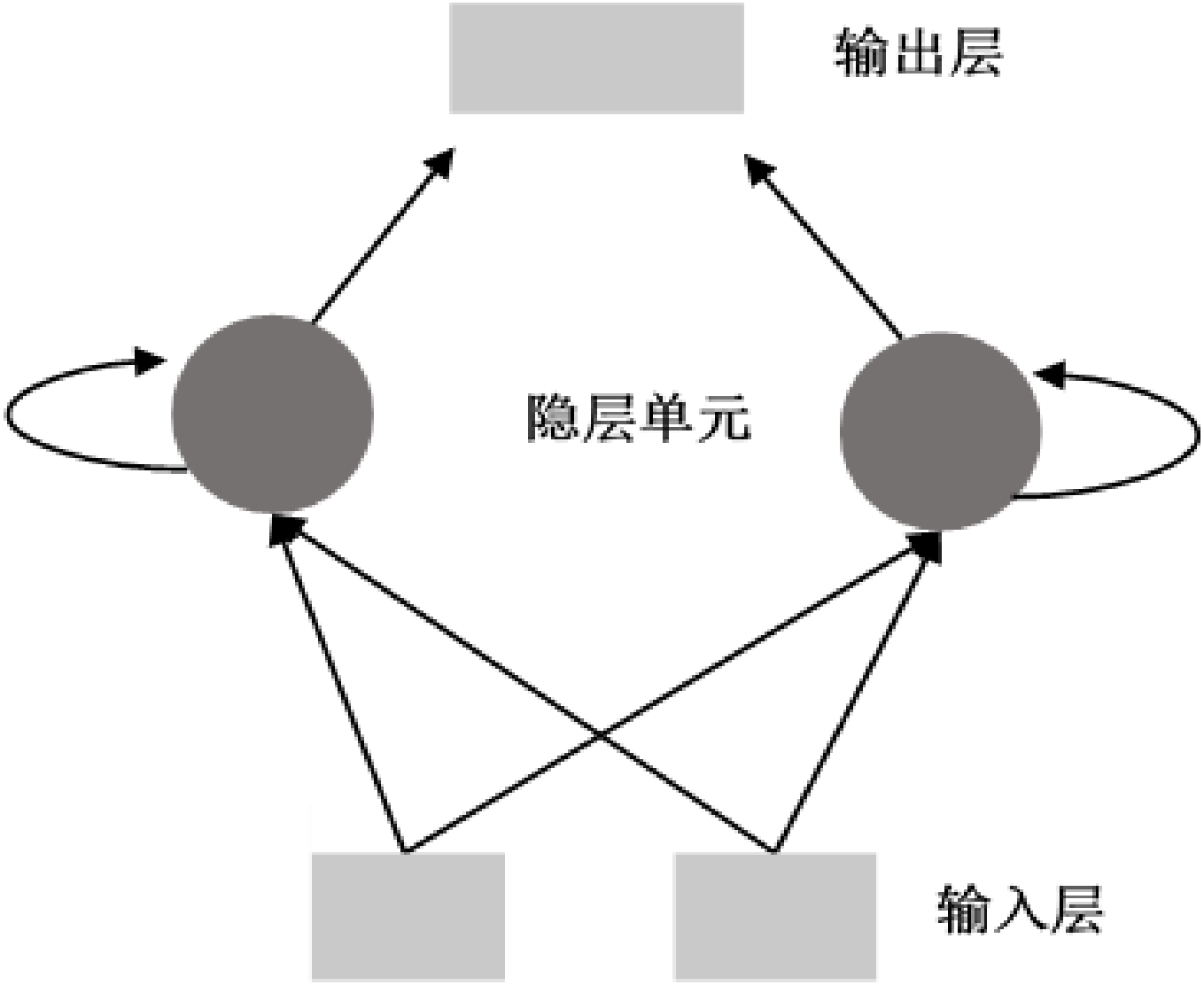
- RNN虽然有记忆功能，但记忆功能退化严重！Vanishing gradient problem
- Vanishing gradient problem makes it hard for an RNN to learn tasks containing delays of more than about 10 timesteps between relevant input and target events!



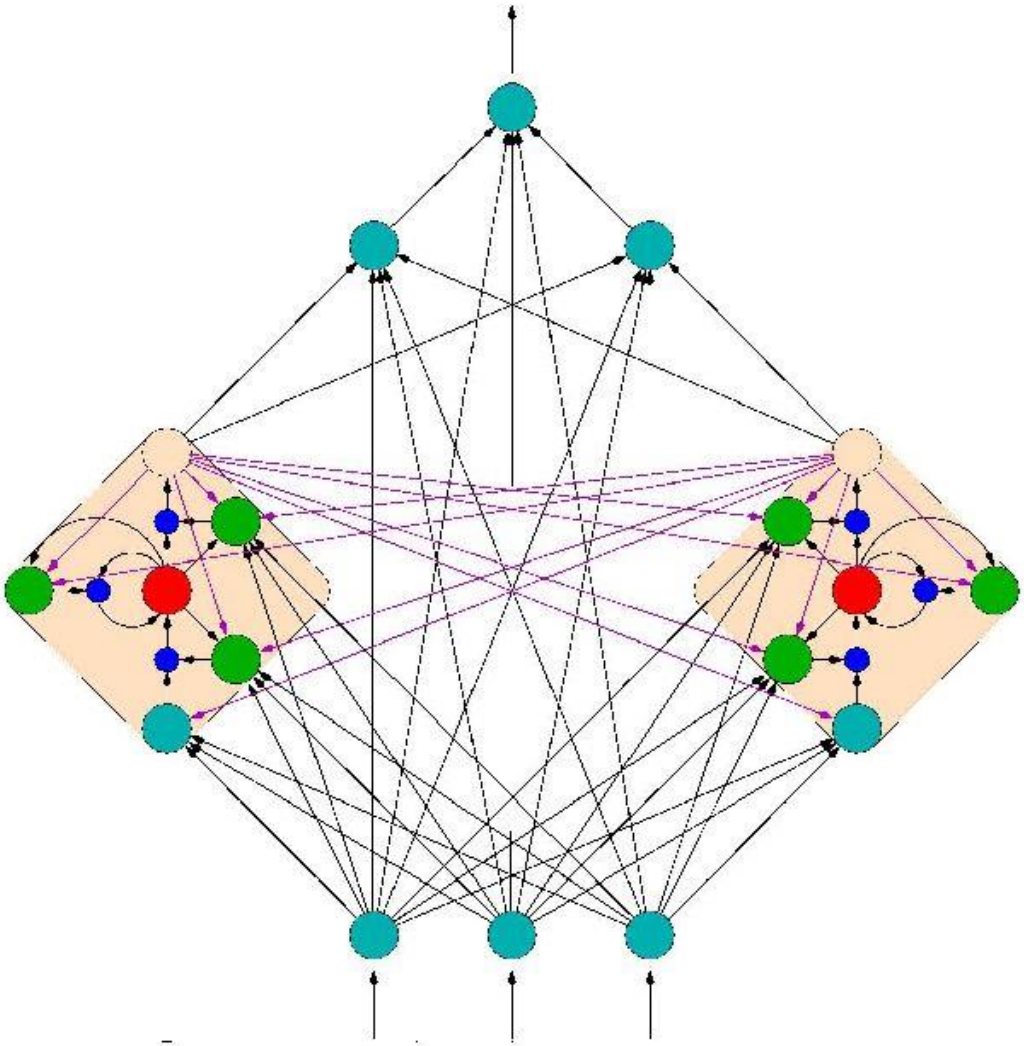
LSTM

LSTM网络由于包含特殊的记忆单元，既可以保留网络中的历史信息，也可以对输出加以判断，因而具有比普通的递归神经网络更优良的性能。

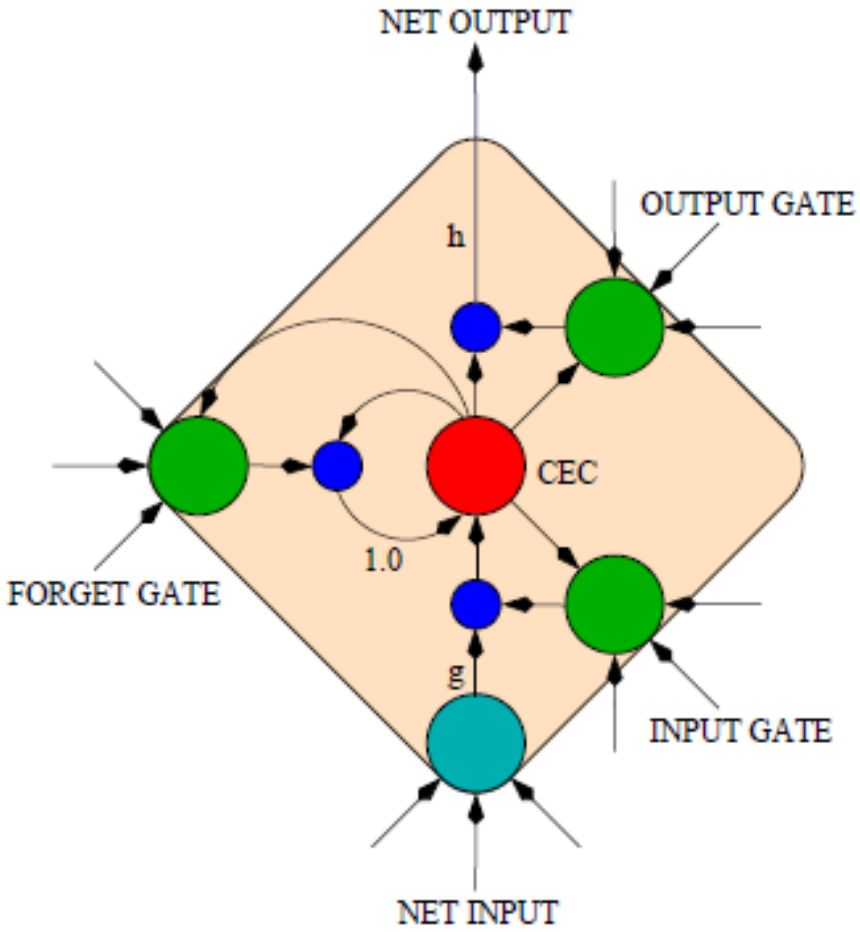
标准的LSTM网络结构由输入层、递归LSTM层以及输出层构成。输入层连接着LSTM层，LSTM层的输入结点直接连接到输出结点，LSTM层的输出结点再连接到整个网络的输出层。LSTM层包含特殊的记忆单元。

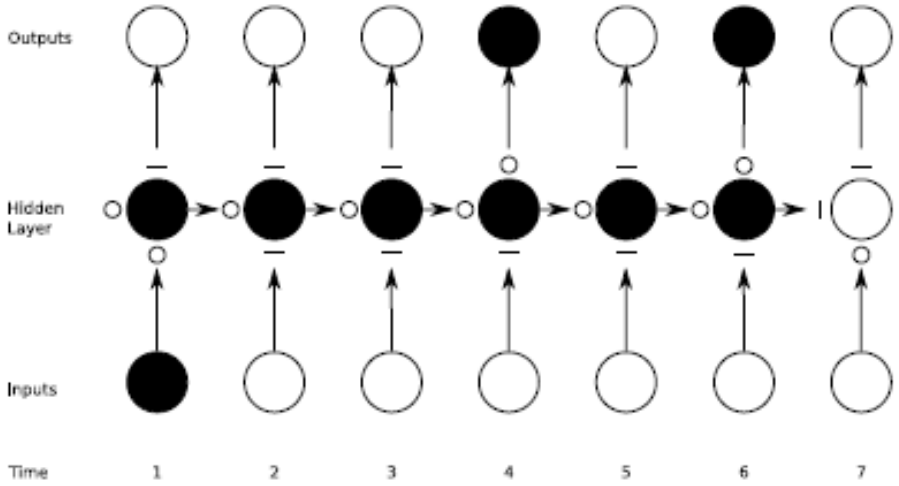

输出层
隐层单元
输入层

LSTM



LSTM


NET OUTPUT
h
OUTPUT GATE
CEC
1.0
FORGET GATE
g
INPUT GATE
NET INPUT


Outputs
Hidden Layer
Inputs
Time
1
2
3
4
5
6
7

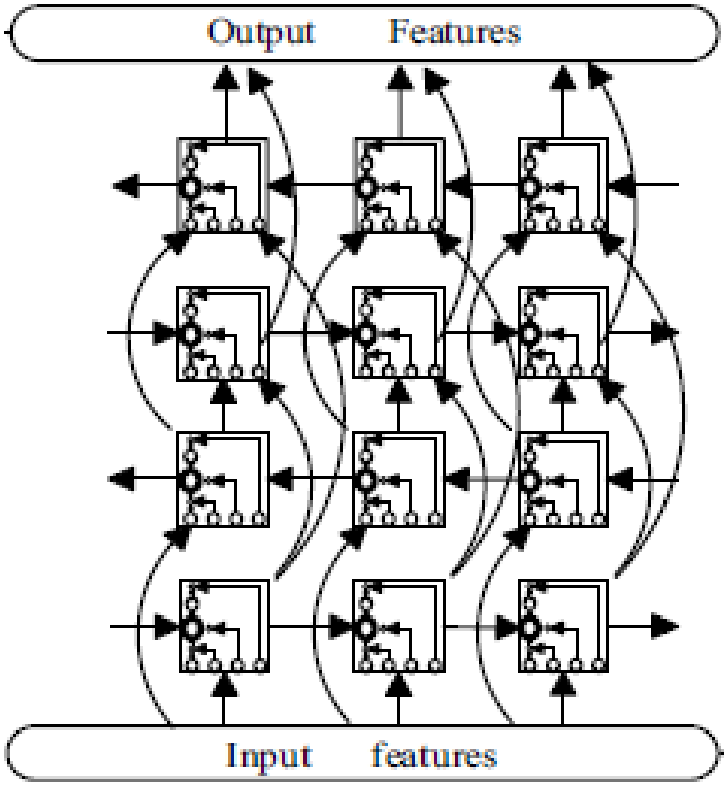
记忆单元包含带有自循环的记忆结点，这些结点可以存储当前网络的状态。每个记忆单元都包含以下几个门：

输入门：控制从输入门流向记忆单元的信息；

输出门：控制从输出结点流向网络其他部分的信息；

忘记门：控制是否将当前结点的状态输出到自循环的回路中，可以自适应地忘记或者重置当前结点记忆信息。

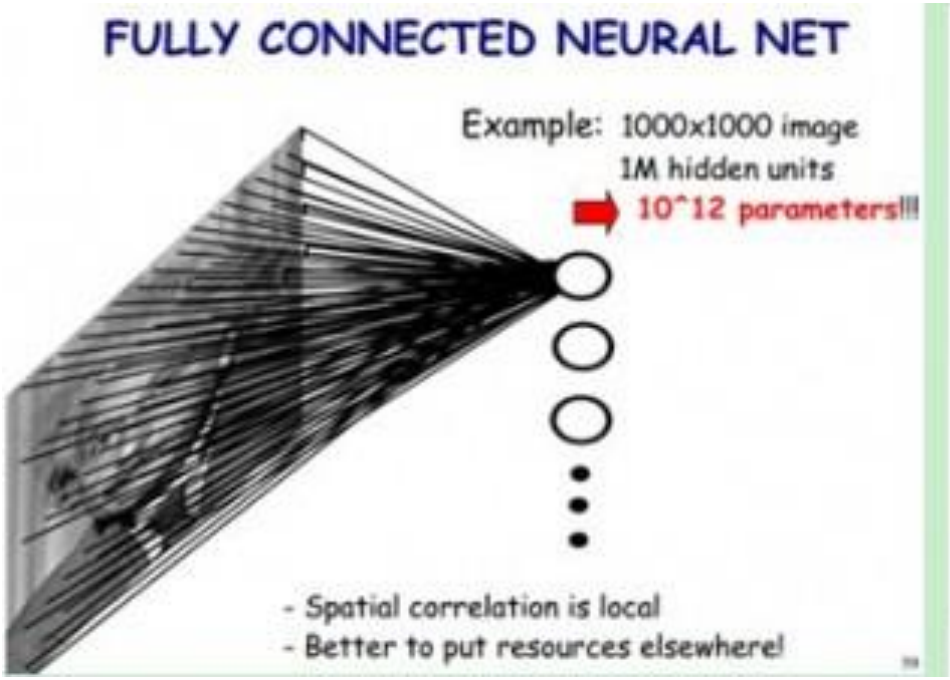
LSTM-DBRNN

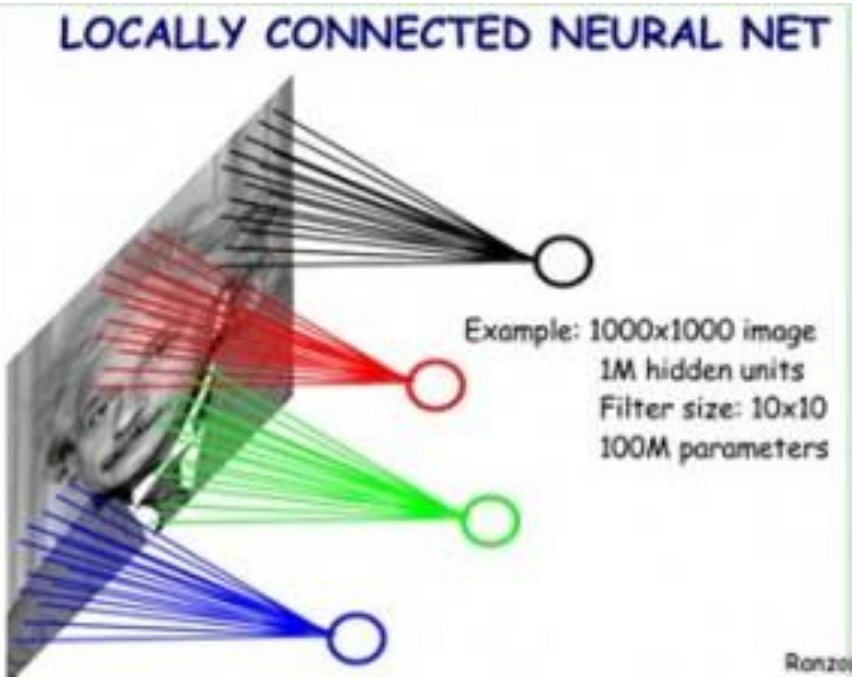


卷积神经网络（CNN）

科学家在研究猫脑皮层中用于局部敏感和方向选择的神经元时发现其独特的网络结构可以有效地降低反馈神经网络的复杂性，继而提出了卷积神经网络（Convolutional Neural Networks-简称CNN）

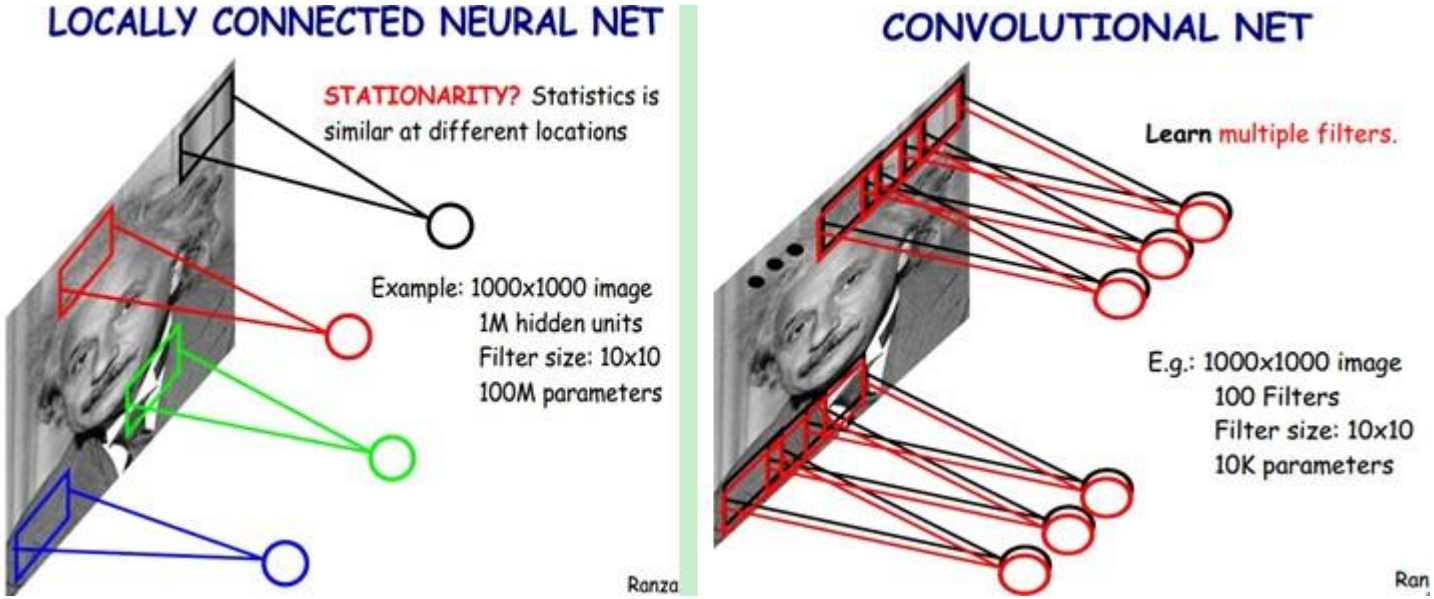
一般认为人对外界的认知是从局部到全局的，而图像的空间联系也是局部的像素联系较为紧密，而距离较远的像素相关性则较弱。因而，每个神经元其实没有必要对全局图像进行感知，只需要对局部进行感知，然后在更高层将局部的信息综合起来就得到了全局的信息。网络部分连通的思想，也是受启发于生物学里面的视觉系统结构。视觉皮层的神经元就是局部接受信息的（即这些神经元只响应某些特定区域的刺激）。如下图所示：左图为全连接，右图为局部连接。

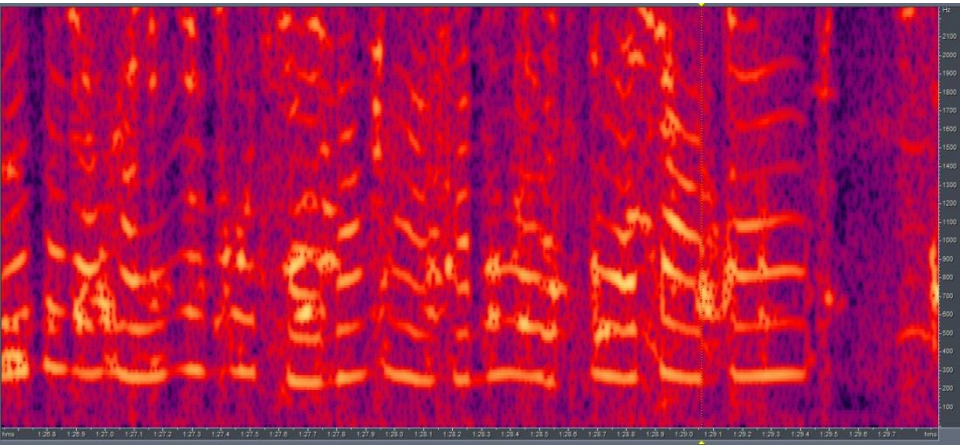

FULLY CONNECTED NEURAL NET
Example: 1000x1000 image
1M hidden units
10^12 parameters!!!
- Spatial correlation is local
- Better to put resources elsewhere!

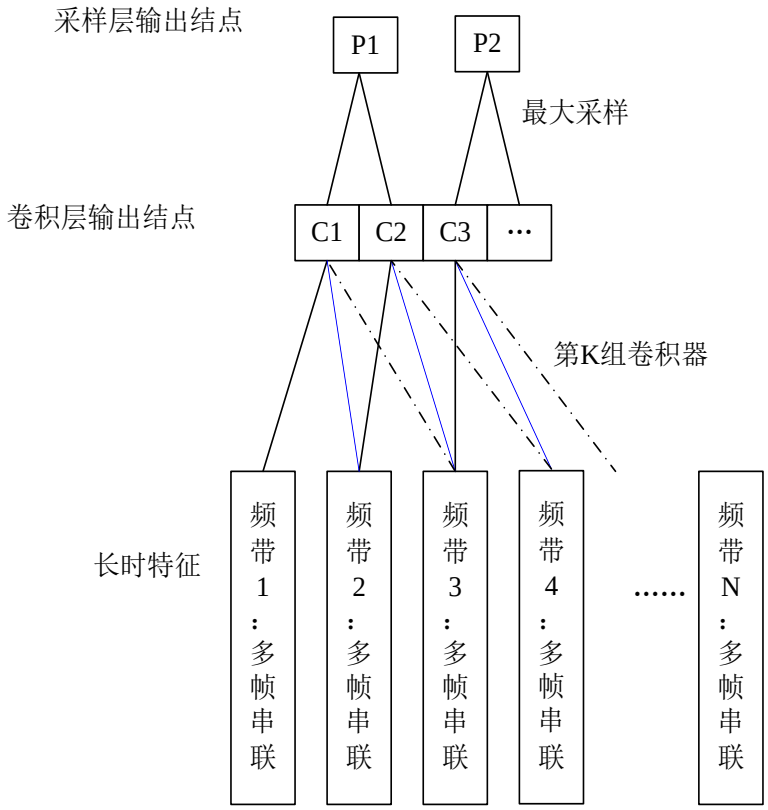

LOCALLY CONNECTED NEURAL NET
Example: 1000x1000 image
1M hidden units
Filter size: 10x10
100M parameters

在上右图中，假如每个神经元只和10×10个像素值相连，那么权值数据为1000000×100个参数，减少为原来的千分之一。而那10×10个像素值对应的10×10个参数，其实就相当于卷积操作。

多卷积核


LOCALLY CONNECTED NEURAL NET
STATIONARITY? Statistics is similar at different locations
Example: 1000x1000 image
1M hidden units
Filter size: 10x10
100M parameters
Ranza
CONVOLUTIONAL NET
Learn multiple filters.
E.g.: 1000x1000 image
100 Filters
Filter size: 10x10
10K parameters
Ran





CNNs中卷积层和最大聚合（采样）层的示例图

实验结果

DNN在声学建模上面的实验结果

中文自然口语电话交谈语音（CTS）识别任务

训练数据：约100小时

DNN结构：616-2048-2048-2048-3399

CTS识别系统不同声学模型的字错误率（%）对比结果

声学模型	训练方式	set1	set2	set3
GMM	MLE	51.7	55.6	63.1
GMM	MPE	48.9	53.1	61.7
DNN	BP	35.3	40.5	49.1
DNN	鉴别性	31.5	37.2	44.6

实验结果

表 4.4: 不同隐含层数目实验结果

隐含层数	参数量	训练样本/参数量	86305	LDC	实网
1	14M	2.85	41.9	46.6	54.5
2	18M	2.2	36.7	41.2	49.7
3	22M	1.81	35.3	40.5	49.1
4	26M	1.53	35.3	39.5	48.1
5	30M	1.33	34.6	39.6	48.1

实验结果

表 汉语普通话大词表非特定人电话自然口语对话系统中**DNN**和**CNN**的对比结果

神经网络	测试集	WER
DNN	86305	48.1%
CNN	86305	47.1%
DNN	HDev04	51.6%
CNN	HDev04	50.7%

相关应用：模式识别领域

神经网络经过训练可有效的提取信号、语言、图像、雷达、声纳等感知模式的特征，并能解决现有启发模式识别系统不能很好解决的不变量测量、自适应、抽象或概括等问题。这方面的主要应用有：图形、符号、手写体及语音识别，雷达及声纳等目标识别，药物构效关系等化学模式信息辨认，机器人视觉、听觉，各种最近相邻模式聚类及识别分类，遥感、医学图像分析，计算机视觉、计算机输入装置等。神经网络可应用于模式识别的各个环节：特征提聚、聚类分析、边缘检测、信号增强、噪音抑制、数据压缩以及各种变换、分类判决等。模式识别是人工神经网络特别适宜求解的一类问题，神经网络模式识别技术在各领域中的广泛应用是神经网络技术发展的一个重要侧面。

相关应用：人工智能领域

专家系统是人工智能领域研究时间最长，应用最成功的技术，但人们在应用专家系统解决诸如语音识别、图像处理和机器人控制等这类似于人脑的形象思维的问题时，却遇到很大的困难。神经网络的问世为人工智能开辟了一条崭新的途径，成为人工智能研究领域中的后起之秀，它具有的自学习能力是传统专家系统望尘莫及的。神经网络技术能对不完整信息进行补全，根据已学会的知识和处理问题的经验对复杂问题作出合理的判断决策，给出较满意的解答，或对未来过程作出有效的预测和估计，从而使之在人工智能领域获得广泛的应用。这个方面的主要应用有：自然语言处理、市场分析、预测估值、系统诊断、事故检查、密码破译、语言翻译、逻辑推理、知识表达、智能机器人、模糊评判等。

相关应用：控制工程领域

神经网络在诸如机器人运动控制、工业生产中的过程控制等复杂控制问题方面有独到之处。较之基于传统数学计算机的离散控制方式，神经网络更适宜于组成快速实施自适应控制系统。这方面的主要应用有：多变量自适应控制、变结构优化控制、并行分布控制、智能及鲁棒控制等。

相关应用：优化计算和联想控制

- 由于并行、分布式的计算结构，神经网络在求解诸如组合优化（NP完备问题）、费心性优化等一系列问题上表现出高速的集体计算能力。在VLSI自动排版、高速通信开关控制、航班分配、货物调度、路径选择、组合编码、排序、系统规划、交通管理以及图论中各类问题的计算等方面得到了成功应用。
- 联想记忆的作用是用一个不完整的模糊的信息联想出储存在记忆中的某个完整、清晰的模式来。如何提高模式储存量和联想质量仍是神经网络的热点之一。目前在这方面的应用又内容寻址器、人脸识别器、知识数据库等。

相关应用：信号处理领域

神经网络的自学习和自适应能力使其成为对各类信号进行多用途加工处理的一种天然工具，主要用于解决信号处理中的自适应和非线性问题。包括自适应均衡、自适应滤波、回拨抵消、自适应波束形成、自适应编码等自适应问题和各种非线性问题，如非线性区域的模式分类、系统辨识和高维非线性系统的检测、估计等问题，还可对病态问题进行求解。神经网络在弱信号检测、通信、自适应滤波等方面的应用尤其引人注目，并已在许多行业得到应用。

语音信号处理中的相关应用

- 基于DNN的声学模型建模
- 基于DNN的语言模型建模
- 基于DNN的语音端点检测
- 基于DNN的情感识别
- 基于DNN的叠音检测
- 基于DNN的语音增强
- ……

虽然神经网络在许多领域内都有成功的应用实例，但神经网络也不是尽善尽美的。目前，神经网络的理论研究和实际用途都在进一步探索之中，相信随着人工神经网络研究的进一步深入，其应用领域会更广，用途会更大。

谢 谢