# 二进制神经网络分类问题的几何学习算法*

朱大铭 马绍汉

(山东大学计算机系 济南 250100)
(中国科学院计算技术研究所 北京 100080)

**摘要** 分类问题在前向神经网络研究中占有重要位置. 本文利用几何方法给出一个二进制神经网络 $K(\geqslant 2)$ 分类问题的新学习算法. 算法通过训练点的几何位置与类别分析,建立一个四层前向神经网络,实现网络输入向量分类. 本文算法的优点在于:保证学习收敛且收敛速度快于BP算法及已有的其他一些前向网络学习算法;算法可以确定神经网络的结构且能实现精确的向量分类. 另外,算法所建神经网络由线性阈值单元组成,神经元突触权值和阈值均为整数,特别适合于集成电路实现.

**关键词** 神经网络,算法,收敛,训练,几何.
**中图法分类号** TP18

给定一 $n$ 维向量集合,按其固有性质可将集合中的向量分为 $K(\geqslant 2)$ 类,分类结果由训练样本给出. 欲根据所有训练样本建立一个前向神经网络,识别 $n$ 维向量的类别,该问题称为神经网络 $K$ 分类问题. 若分类问题的神经网络输入和输出向量均为二进制数,则又称为二进制映射分类问题. BP 算法[1]是目前应用广泛且研究较深的前向网络学习算法,该算法的缺陷在于收敛速度慢(甚至不收敛)且学习不精确,用于二进制映射学习则弱点更为明显. BP 算法学习不精确的根本原因在于学习中无法确定隐层神经元个数. 在前向网络学习算法研究中,隐层神经元个数的确定是一个有待探索的重要问题. STONE-WEISTRASS 定理[2]只给出了定性结论. 对于二分问题,目前已有卡若图法[3]和几何方法[4]. 但这两种方法均不能有效地直接用于一般二进制映射 $K(>2)$ 分类问题. 另外,神经网络顺序生成方法[5]虽可以实现二进制映射 $K$ 分类问题学习,但该方法仍有若干参数需要人凭经验选择,其收敛速度和隐层神经元学习的优化程度均受到人为选择的影响.

本文给出二进制映射 $K(\geqslant 2)$ 分问题的一个几何学习算法. 算法通过几何学习得到一个计算给定 $K$ 分问题的4层神经网络. 第1层为输入层,输入神经元个数等于输入二进制向量位数;第2层为隐层,隐层神经元个数由隐层几何学习算法首先确定;第3层和第4层又分别称为第1输出层和第2输出层,神经元个数分别固定为 $K$ 和 $[\log_2 K]$. 2个输出层向

* 本文研究得到国家自然科学基金、国家863高科技项目基金和山东省自然科学基金资助. 作者朱大铭,1964年生,博士生,副教授,主要研究领域为神经网络,算法分析与设计. 马绍汉,1938年生,教授,博士导师,主要研究领域为算法设计与分析,人工智能.

本文通讯联系人:朱大铭,北京 100080,中国科学院计算技术研究所

本文1996-09-17收到修改稿

量可分别独立地表达输入向量的类别.该算法的关键在于隐层神经元的确定方法:通过几何分析产生一组超平面,将具有不同目标输出的 $K$ 类训练点分割在不同区域中,每个超平面对应确定一个隐层神经元.该算法保证学习收敛性且学习复杂度大大低于文献[4]的方法.另外,算法确定的网络神经元均为线性阈值单元[6],特别适合于集成电路实现.

本文几何算法与BP算法比较,其优势在于极快的收敛速度和学习的精确性.在学习中产生的隐层神经元个数也是较理想的.

## 1 问题简述

### 1.1 训练集合与二进制映射 K(≥2)分类

二进制映射 $K$ 分类问题由训练集合给出 $U=\{u_1,\ldots,u_N\}=U_1+\ldots+U_K, U_i\neq\varnothing, U_i\cap U_j=\varnothing, 1\leqslant i,j\leqslant K$,其中训练样本 $u_i=(v_i,o_i)$ 由二进制输入向量 $v_i=(v_{i1},\ldots,v_{in})^T$ 和标志该向量所属类别的输出向量 $o_i=(o_{i1},\ldots,o_{im})^T$ 组成. $v_{ij},o_{il}\in\{0,1\}, 1\leqslant i\leqslant N, 1\leqslant j\leqslant n, 1\leqslant l\leqslant m$.一般输出向量表达输入向量类别有2种方式,分别满足条件:(a) $m=K$ 且 $o_i\in\{\delta_1,\ldots,\delta_K\}, 1\leqslant i\leqslant N$,其中 $\delta_j=(\delta_{j1},\ldots,\delta_{jK})^T$, $\delta_{jj}=1, \delta_{jk}=0, j\neq K, 1\leqslant j,k\leqslant K$.若 $u_i\in U_j$,则 $o_i=\delta_j$. (b) $m=\lceil\log_2 K\rceil, o_i\in\{0,1\}^m$.若 $u_i\in U_j$,则 $B(o_{i1},\ldots,o_{im})=j-1, 1\leqslant i\leqslant N, 1\leqslant j\leqslant K$.其中 $B(o_{i1},\ldots,o_{im})$ 表示由 $o_{i1},\ldots,o_{im}$ 形成的二进制数.

在下文讨论中,样本 $u_i$ 的输入向量也称为 $u_i$ 的训练点,记为 $V(u_i)$,并将任意训练样本子集 $U_a\subseteq U$ 中样本的输入向量集合称为 $U_a$ 的训练点集,记为 $V(U_a)$.若 $u_i\in U_j$,则称 $u_i$ 为第 $j$ 类训练样本,$V(u_i)$ 为第 $j$ 类训练点.为方便起见,亦将输入和输出向量简记为 $v_i=v_{i1}\ldots v_{in}, o_j=o_{j1}\ldots o_{jm}$.

### 1.2 神经网络

下面将实现二进制映射 $K$ 分类问题的前向神经网络称为该问题的目标神经网络.给定一个二进制映射 $K$ 分类问题,本文算法建立由线性阈值单元组成,实现给定 $K$ 分类问题的3层或4层目标网络.线性阈值单元的功能描述如图1,其中线性阈值单元的作用函数为

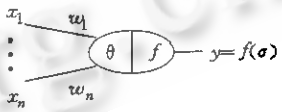

图1 线性阈值单元

$$y=f(\sigma)=\begin{cases}1, \sigma=\sum_{i=1}^{n}w_ix_i-\theta\geqslant 0\\ 0, \sigma=\sum_{i=1}^{n}w_ix_i-\theta<0\end{cases} \tag{1}$$

任一 $n-1$ 维超平面对 $n$ 维空间中点的分离作用可由一个 $n$ 输入线性阈值单元实现.线性阈值单元将超平面的一边赋予0,另一边赋予1,超平面则定义了两类顶点之间的一个边界.一个 $K$ 分类问题至少需要 $K-1$ 个超平面才能将 $K$ 类训练点分开.因此目标神经网络隐层至少需要 $K-1$ 个神经元.一般仅有 $K-1$ 个超平面并不能将 $K$ 类训练点完全分开,本文算法首先通过对训练点的几何分析确定一组超平面将 $K$ 类训练点分开,从而确定了全部隐层神经元.隐层神经元的产生顺序及其对训练点的分离特征决定了输出层神经网络的学习方法.为下文叙述方便,给出空间点集的线性可分性定义.

**定义1.** 设 $V_1,V_2$ 均是 $n$ 维非空点集,且 $V_1\cap V_2=\varnothing$.若存在 $n-1$ 维超平面 $\sum_{i=1}^{n}w_ix_i-\theta=0$,使 $\forall (y_1,\ldots,y_n)^T\in V_1, \sum_{i=1}^{n}w_iy_i-\theta\geqslant 0$;且 $\forall (y_1,\ldots,y_n)^T\in V_2, \sum_{i=1}^{n}w_iy_i-\theta<0$,则

称 $V_1$ 和 $V_2$ 是线性可分的 $n$ 维点集.

## 2 隐层网络学习

隐层网络学习过程是隐层神经元的逐次产生过程. 在学习过程中需选择一类样本进行训练,这类样本称为真子类样本,其余样本均称为假子类样本.

### 2.1 真子类与核心顶点选择

开始,算法任意选择一类样本形成真子类样本集 $U_t=U_i$, $1\leqslant i\leqslant K$.

**定义** 2. 给定二进制映射 $K$ 分类问题训练集合 $U$. 若 $U_t$ 是真子类样本集,则称 $V(U_t)$ 中的训练点为真顶点;若 $U_t$ 由选择第 $i$ 类样本形成,则记 $T(U_t)=i$. 设 $V_t\subseteq V(U)$ 是某真顶点集,若 $V_t$ 和 $V(U)-V_t$ 是线性可分的,则称 $V_t$ 是真顶点分离集. 即不属于 $V(U_t)$ 也不属于 $V_t$ 的训练点称为假顶点.

最初选定真子类样本集 $U_t$,真顶点分离集 $V_t=\varnothing$. 算法在 $V(U_t)$ 中选择真顶点 $v_c$ 作为核心顶点. 第 1 个核心顶点可根据文献[7]的 $k$ 最近相临算法选择,也可任意选择. $V_t=\{v_c\}$ 成为第 1 个真顶点分离集. 设 $v_c=(v_{c1},\ldots,v_{cn})^T$,分离 $V_t$ 的超平面为 $\sum_{i=1}^{n}(2v_{ci}-1)x_i-\sum_{i=1}^{n}v_{ci}=0$.

### 2.2 真顶点分离集的扩展

不断选择新真顶点进入 $V_t$,形成新真顶点分离集,该过程称为真顶点分离集的扩展过程. 下面讨论真顶点分离集的扩展方法. 首先引入 2 个符号:

设 $I=\{i_1,\ldots,i_m\}$ 是正整数集合,其中 $m\leqslant n$, $1\leqslant i_j\leqslant n$, $1\leqslant j\leqslant m$. 则对任意 $n$ 维二进制向量 $x=(x_1,\ldots,x_n)^T\in\{0,1\}^n$,记 $x(I)=(x_{i_1},\ldots,x_{i_m})^T\in\{0,1\}^m$. 另设 $x,y\in\{0,1\}^n$ 均为 $n$ 维二进制向量,则 $x$ 和 $y$ 之间的 HAMMING 距离记为 $H(x,y)$.

**定理** 1. 给定二进制映射 $K$ 分类问题的训练集 $U$. 设 $1\leqslant d\leqslant m\leqslant n$, $I=\{i_1,\ldots,i_m\}$ 是整数集合, $1\leqslant i_j\leqslant n$, $1\leqslant j\leqslant m$,则训练点集 $V_t=V_a\cup V_b\cup V_c$ 是真顶点分离集. 其中 $V_a=\{v_c\}$; $V_b=\{x\mid x\in V(U),H(x,v_c)\leqslant d-1\}$; $V_c=\{x\mid x\in V(U),H(x(I),v_c(I))=H(x,v_c)=d\}$.

证明:不失一般性设 $I=\{1,2,\ldots,m\}$. 考虑超平面 $net(x,\theta)=\sum_{k=1}^{n}w_kx_k-\theta=0$,其中

$$w_k=\begin{cases}\alpha, v_{ck}=1, k\in I\\ -\alpha, v_{ck}=0, k\in I\\ \beta, v_{ck}=1, k\notin I\\ -\beta, v_{ck}=0, k\notin I\end{cases},\quad \theta=\sum_{k=1}^{n}w_kv_{ck}-\gamma$$

任取 $x=(x_1,\ldots,x_n)^T\in\{0,1\}^n$,易证:

$$net(x,\theta)=-\alpha H(x(I),v_c(I))-\beta H(x(I_1),v_c(I_1))+\gamma \tag{2}$$

其中 $I_1=\{m+1,\ldots,n\}$. 若 $d=1$,则取 $\alpha=1,\beta=2,\gamma=1$;若 $d\geqslant 2$,则取 $\alpha=d-1,\beta=d,\gamma=d(d-1)$. 不难验证,对于 $x\in V_t$,有 $net(x,\theta)\geqslant 0$;而对于 $x\notin V_t$ 有 $net(x,\theta)<0$.

根据定理 1 可给出扩展真顶点分离集 $V_t$ 的方法 1:若距核心顶点 $v_c$ HAMMING 距离小于 $d$ 的训练点均为真顶点,则同时进入真顶点分离集 $V_t$;若距 $v_c$ HAMMING 距离小于 $d$ 的训练点均已进入 $V_t$ 且距 $v_c$ HAMMING 距离等于 $d$ 的训练点非全部真顶点,则选择真顶点集 $V_c=\{x\mid x\in V(U),H(x(I),v_c(I))=H(x,v_c)=d\}$ 并入真顶点分离集 $V_t$($V_t\leftarrow V_t$

$\cup V_c$). 分离超平面由定理 1 的证明过程立即给出. 方法 1 可用来将真顶点分离集同时扩展多个顶点，但只适用于核心顶点选定后的首次扩展. 下面讨论扩展真顶点分离集的更一般方法.

**定义 3.** 给定 $n$ 位二进制向量集合 $V$，对于 $x=(x_1,\ldots,x_n)^T\in\{0,1\}^n$，将 $x$ 与 $V$ 中所有向量的 HAMMING 距离之和称为 $x$ 到 $V$ 的距离，记为 $d_H(x,V)=\Sigma_{y\in V}H(x,y)$.

**定理 2.** 给定二进制映射 $K$ 分类问题的训练集 $U$. 设 $V_t$ 是真顶点集，$x=(x_1,\ldots,x_n)^T$. 若 $\max\limits_{x\in V_t}\{d_H(x,V_t)\}<\min\{d_H(x,V_t)\mid x\in V(U)-V_t\}$，则 $V_t$ 是真顶点分离集，分离超平面为

$$net(x,\theta)=\Sigma_{i=1}^{n}(2c_i-c_0)x_i-\theta=0 \tag{3}$$

其中 $c_0=|V_t|$，$c_i=\Sigma_{x\in V_t}x_i$，$\theta=\Sigma_{x\in V_t}\Sigma_{i=1}^{n}x_i-\max\{d_H(x,V_t)\mid x\in V_t\}$.

证明：任给 $x=(x_1,\ldots,x_n)^T\in\{0,1\}^n$，有

$$\begin{aligned}net(x,\theta)&=\Sigma_{y\in V_t}\Sigma_{i=1}^{n}(2y_i-1)x_i-\theta\\&=\Sigma_{y\in V_t}\Sigma_{i=1}^{n}(2y_i-1)y_i-d_H(x,V_t)-\theta\\&=\Sigma_{y\in V_t}\Sigma_{i=1}^{n}y_i-d_H(x,V_t)-\theta\end{aligned} \tag{4}$$

若 $x\in V_t$，则
$$net(x,\theta)=\max_{y\in V_t}\{d_H(y,V_t)\}-d_H(x,V_t)\geqslant 0 \tag{5}$$

若 $x\in V(U)-V_t$，则
$$net(x,\theta)=\max_{y\in V_t}\{d_H(y,V_t)\}-d_H(x,V_t)<0 \tag{6}$$

故 $V_t$ 是真顶点分离集.

于是根据定理 2 可给出扩展真顶点分离集 $V_t$ 的方法 2：在 $V(U_t)-V_t$ 中选择 $v_t$，使 $d_H(v_t,V_t)=\min\{d_H(x,V_t)\mid x\in V(U_t)-V_t\}$. 利用定理 2 判定 $V_t+\{v_t\}$ 是否真顶点分离集，若是，则 $v_t$ 进入 $V_t$ 形成新真顶点分离集；否则 $V_t$ 扩展失败. 其中 $U_t$ 是真子类样本集. 核心顶点选定后，先按方法 1 扩展 $V_t$，再按方法 2 继续扩展 $V_t$，直至扩展失败，算法由此得到第 1 个分离超平面，对应得到第 1 个隐层神经元.

例 1：设 $n=3$，真顶点集为{000,001,010,011,111}，假顶点集为{100,101,110}，选定核心顶点 $v_c=000$，利用方法 1 得到 $V_t=\{000,001,010\}$，再利用方法 2 两次扩展 $V_t$ 得到 $V_t=\{000,001,010,011,111\}$，分离超平面为 $-3x_1+x_2+x_3+1=0$.

### 2.3　真子类替换与真顶点分离集再扩展

进入真顶点分离集 $V_t$ 的训练点对应的样本视为已被训练过的样本. 真顶点分离集扩展的目标是使所有样本均得到训练. 为进一步寻求分离超平面，算法选择另一类样本代替原来的真子类样本，方法如下：(1)将不在 $V_t$ 中的真顶点恢复为假顶点. $U_t$ 中的样本不再视为真子类样本. (2)在 $V(U)-V_t$ 中选择距 $V_t$ 最近的训练点 $v_t$，使 $d_H(v_t,V_t)=\min\{d_H(x,V_t)\mid x\in V(U)-V_t\}$. 设 $v_t\in V(U_j)$，$1\leqslant j\leqslant K$，则重新选择新真子类样本集 $U_t=U_j$，即使 $T(U_t)=j$.

真子类替换后，$V_t+V(U_t)$ 中的训练点均为真顶点. 真顶点分离集 $V_t$ 得以继续扩展. $V_t$ 扩展再次失败时，算法得到第 2 个分离超平面，对应第 2 个隐层神经元. 这一过程可一直进行下去. 若所有训练点均已进入真顶点分离集中，则隐层神经元全部产生，结束. 若真子类替换后，真顶点分类集 $V_t$ 仍不能继续扩展，则算法进入阻塞状态. 设阻塞状态下的真顶点分离集 $V_t=V(U_z)$，$U_z\subset U$. 此时 $U_z$ 中的训练样本成为无关样本，$V_t$ 中的训练点成为无关顶点. 因此若遇阻塞状态，则清除 $U_z$：$U\leftarrow U-U_z$，并将真顶点分离集清空：$V_t\leftarrow\varnothing$. 于是算法只需

重新选择真子类样本和核心顶点,便可继续真顶点分离集 $V_i$ 的扩展过程.

## 3 输出层网络学习

设隐层学习算法求得 $M$ 个隐层神经元,分别记为 $H_1,\ldots,H_M$,其输出量分别为 $h_1,\ldots,h_M$. 将隐层输出表达为输入向量函数的形式:$h_i=h_i(x)\in\{0,1\}$,$x=(x_1,\ldots,x_n)^T\in V(U)$,$1\leqslant i\leqslant M$.

### 3.1 第1输出层学习

第1输出层网络固定 $K$ 个线性阈值单元. 该层学习确定由隐层到该层的连接权值和该层神经元阈值,使该层输出向量满足输出条件(a). 设该层输出量分别记为 $o_1^1,\ldots,o_K^1$,需有

$$O_i^1(h_1(x),\ldots,h_M(x))=\begin{cases}1, & x\in V(U_i)\\ 0, & x\notin V(U_i)\end{cases},\quad 1\leqslant i\leqslant K \tag{7}$$

**定义4.** 设隐层学习算法得到隐层神经元 $H_j$ 时,$T(U_i)=i$,则称 $H_j$ 为第 $i$ 类隐层神经元,记为 $D(H_j)=i$,$1\leqslant j\leqslant M$,$1\leqslant i\leqslant K$.

**定理3.** 给定二进制映射 $K$ 分类问题的训练集 $U=U_1+\ldots+U_K$. 由隐层学习得到 $M$ 个隐层神经元 $H_1,\ldots,H_M$,则集合$\{(h_1(x),\ldots,h_M(x))^T|x\in V(U_i)\}$和$\{(h_1(x),\ldots,h_M(x))^T|x\in V(U-U_i)\}$是线性可分的 $M$ 维点集. $1\leqslant i\leqslant K$.

证明:考虑如下构造 $M-1$ 维超平面 $net(x,\theta)=\sum_{k=1}^{M}w_kx_k-\theta=0$ 的算法.

输出层学习算法1:

(1)$w_1=w_2=\ldots=w_M=0$. 若 $D(H_M)=i$,则 $\theta=0$;否则 $\theta=1$.

(2)for $k=M-1$ down to 1 step $-1$

(2.1)若 $D(H_k)=i$,则取 $w_k=\max\{\theta-\sum_{j=k+1}^{M}w_jx_j|x_j\in\{0,1\}\}$.

(2.2)若 $D(H_k)\neq i$,则取 $w_k=\min\{\theta-\sum_{j=k+1}^{M}w_jx_j|x_j\in\{0,1\}\}-1$.

设 $x\in V(U)$. $x$ 进入真顶点分离集后算法产生的第1个隐层神经元为 $H_l$,$1\leqslant l\leqslant M$. 则 $h_l(x)=1$. 且对于 $j$,$1\leqslant j<l$,$h_j(x)=0$. 若 $x\in V(U_i)$,则 $D(H_l)=i$,故

$$\begin{aligned}net(x,\theta)&=\Sigma_{k=1}^{M}w_kh_k(x)-\theta=\Sigma_{k=1}^{M}w_kh_k(x)-\theta\\&=w_i+\Sigma_{k=i+1}^{M}w_kh_k(x)-\theta\geqslant 0\end{aligned} \tag{8}$$

若 $x\in V(U-U_i)$,则 $D(H_l)\neq i$. 故

$$\Sigma_{k=1}^{M}w_kh_k(x)-\theta=w_i+\Sigma_{k=i+1}^{M}w_kh_k(x)-\theta\leqslant -1<0. \tag{9}$$

若隐层学习得到神经元个数 $M=2$,则必然 $K=2$. 此时只需隐层神经元 $H_1$ 即可实现正确分类,而无需输出层神经元参与. 若 $M\geqslant 3$,由定理3易知,只需按照输出层学习算法1分别计算 $K$ 个神经元的输入权值和阈值,则输出向量满足二进制映射 $K$ 分类问题的输出条件(a). 进一步考察算法可知,$H_M$ 只用于第1输出层神经元阈值的计算,而到第1输出层网络的连接权值恒为0. 因此目标分类网络隐层只需由 $H_1,\ldots,H_{M-1}$ 组成即可.

### 3.2 第2输出层学习

第2输出层固定为 $m=\lceil\log_2K\rceil$ 个神经元,该层输出量分别记为 $o_1^2,\ldots,o_m^2$,其中 $o_i^2(x)=o_i^2(o_1^1(x),\ldots,o_K^1(x))\in\{0,1\}$,$1\leqslant i\leqslant m$,$x\in V(U)$. 欲使该层输出向量满足二进制映射 $K$ 分类问题的输出条件(b),须对第 $i$ 类训练点 $x\in V(U_i)$ 满足:$B(o_1^2(x),\ldots,o_m^2(x))=i-1$,$1\leqslant i\leqslant K$. 第1输出层学习完成后,产生第2输出层网络已相对较易.

**定理 4.** 给定二进制映射 $K$ 分类问题的训练集 $U=U_1+\ldots+U_K$. 若 $(o_1^1(x),\ldots,o_K^1(x))^T$ 满足输出条件(a)，则由关系式 $o_i^2(x)=f(\Sigma_{j=1}^K w_{ij}o_j^1(x)-\theta_1)$，$1\leqslant i\leqslant m$，$m=\lceil\log_2 K\rceil$，确定的第 2 输出层向量 $(o_1^2(x),\ldots,o_m^2(x))^T$ 必满足输出条件(b)，$x\in V(U)$. 其中 $f(\cdot)$ 表示线性阈值单元的作用函数，$w_{ij}$ 和 $\theta_i$ 由如下算法得到

输出层学习算法 2：

(1)for $i=1$ up to $m$ step 1

　　$\theta_i=1$，$w_{i1}=\ldots=w_{iK}=0$.

(2)for $i=1$ up to $m$ step 1

　　for $j=1$ up to $2^{l-1}$ step 1

　　for $k=1$ up to $2^{m-i}$ step 1

　　if $(l=(2j-1)*2^{m-i}+k\leqslant K)$ then $w_{il}=1$.

证明：注意到第 1 输出层向量满足条件(a)，并分析第 2 输出层向量 $(o_1^2(x),\ldots,o_m^2(x))^T$ 和第 1 输出层向量 $(o_1^1(x),\ldots,o_K^1(x))^T$ 之间的逻辑关系，不难得定理 4 成立.

于是由输出层学习算法 2 立即建立起目标网络的最后一层——第 2 输出层，且该层输出向量满足二进制映射 $K$ 分类问题的输出条件(b).

例 2：设 $n=4$，$K=3$. 欲建立前向网络实现图 2 所示各区域像素点分类识别，可建立训练集合 $U=U_1+U_2+U_3$，其中 $V(U_1)=\{0000,0001,0100,0101\}$，$V(U_2)=\{0010,0110,1000,1001,1010\}$，$V(U_3)=\{0011,0111,1011,1100,1101,1110,1111\}$. 利用本文几何算法得到前向网络如图 3.

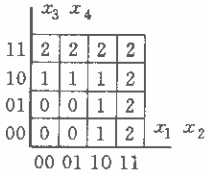


图2

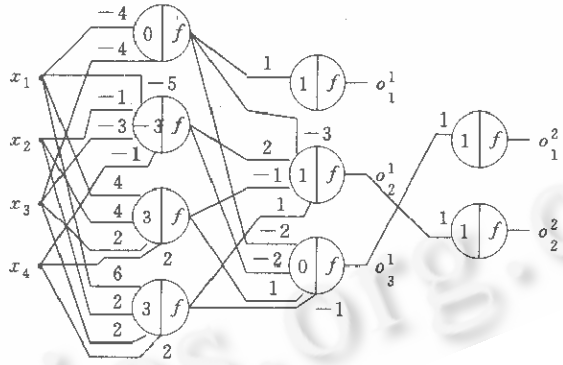


图3

## 4 讨　论

### 4.1 收敛速度

易知在隐层学习过程中，每个训练样本只可能被训练一次. 因训练样本个数总是有限的，故隐层学习必结束. 隐层学习得到有限个神经元，由定理 3 和定理 4 易知，第 1 输出层学习和第 2 输出层学习均必结束且得到正确输出. 故本文算法是收敛的. 学习算法的时间复杂度标志着算法的收敛速度，本文算法的时间复杂度可以确切给出. 这也是本文算法不同于以梯度下降法为学习准则的一类算法的一个突出标志.

考察真顶点分离集扩展过程易知，真顶点分离集每扩展一个顶点最坏复杂度为 $O(N)$，隐层学习最多扩展 $N$ 个训练点. 故隐层学习算法的最坏复杂度为 $O(N^2)$. 分别考察第 1 输出层和第 2 输出层的 2 个学习算法易知，第 1 输出层学习最坏复杂度为 $O(KM)$；第 2 输出层学习最坏复杂度为 $O(K\log_2 K)$. 因 $K$，$M$ 均远远小于样本总数 $N$，故整个算法的复杂度主

要体现为隐层学习时间复杂度$O(N^2)$.

### 4.2 比 较

文献[4]给出二进制映射2分问题学习算法.若以该算法为基础经$K-1$次调用实现$K$($>2$)分学习[4],则必然导致过多的隐层神经元产生且隐层学习复杂度为$O(KN^2)$.这样的算法进行$K$分问题学习显然是不能接受的.本文算法的隐层学习复杂度总是$O(N^2)$,并不随$K$而变化.至于输出层学习复杂度随$K$而增大则是任何算法均不能避免的,且输出层学习复杂度在整个学习复杂度中只占次要地位.另外,多次计算机模拟实验和在语音识别中的应用[8]表明:从学习所得隐层神经元个数考虑,本文算法也是较优的.

本文几何算法可以认为是多层感知器学习的一种特殊情况.BP算法虽然广泛应用于多层感知器学习中,但因BP算法学习是在连续空间中搜索求解,用于二进制映射问题学习则会带来太长的学习时间和太低的学习效率.本文算法较好地解决了这一问题.从分类的可靠精确程度和学习收敛速度考虑,几何算法较BP算法具有很大优越性.本文算法与BP算法的根本区别在于本文算法可以通过学习确定神经网络结构,而BP算法则不能.另外,神经网络顺序生成算法也是一个可以确定网络结构的学习算法[5],但该算法的若干参数仍需人为选择或经多次实验确定,在确定单个神经元时每个样本仍需参与多次训练,算法运算结果的好坏和运行时间长短与人为选择的参数有很大关系.因此仍未完全脱离BP算法的训练模式.本文算法建立的前向网络均由线性阈值单元组成,每个神经元的输入权值和阈值均为整数,因而适于集成电路实现.

## 参考文献


1 Rumelhart D E, McClelland J L. Parallel distributed processing. Cambridge, MA: MIT Press, 1987.

2 Cotter N E. The Stone-weierstrass theorem and its application to neural networks. IEEE Trans. Neural Networks, 1990, 1(12).

3 Gray D L, Michel A N. A training algorithm for binary feed-forward neural network. IEEE Trans. Neural Networks, 1993, 4(3): 176~194.

4 Kim J H, Park S K. The geometrical learning of binary neural networks. IEEE Trans. Neural Networks, 1995, 6(1).

5 Marco Muselli. On sequential construction of binary neural networks. IEEE Trans. Neural Networks, 1995, 6(3): 678~690.

6 焦李成. 神经网络系统理论. 西安:西安电子科技大学出版社, 1991.

7 Fukunaga K. Introduction to statistical pattern recognition. 2nd edition. New York: Academic, 1990.

8 宋兴彬. 基于神经网络的语音识别[硕士论文]. 山东大学, 1995.


# A GEOMETRICAL LEARNING ALGORITHM OF BINARY NEURAL NETWORKS FOR CLASSIFICATION


ZHU Daming MA Shaohan

(*Department of Computer Science Shandong University Ji'nan* 250100)

(*Institute of Computing Technology The Chinese Academy of Sciences Beijing* 100080)

**Abstract**　Binary to binary mapping for classification plays an important role in the researches on feed-forward-neural-network learning. In this paper, the geometrical method is employed to work out a new algorithm to train binary neural networks for classification. By analysis of every training vertex's geometrical location, the algorithm always produces a neural network of four layers for a certain classification problem. The advantages of this algorithm are: it runs with guaranteed convergence and goes to converge much more quickly than BP and some other algorithms; it can determine the structure of the neural networks by learning so that a precise classification is carried out. In addition, every neuron generated by the algorithm employs a hard-limit activation function with integer synaptic weights, which makes the actual implementation by VLSI technology more facilitated.

**Key words**　Neural networks, algorithm, convergence, training, geometry.

**Class number**　TP18