一个基于相关存储的模式识别方法

周景洲
(深圳大学信息工程学院,广东深圳 518060)

摘 要: 应用相关的存储技术实现模式识别可缩减存储大小,但由此带来的饱和问题严重影响系统正常工作,为解决此问题,本文在分析现有系统的基础上给出了基于一系列逻辑操作的识别技术.该技术的实质是用一形式化的方法增加输入模式间的线性无关性.通过论证与实例验证,该方法是有效而实用的.

关键词: 模式识别; 神经网络; 相关存储; 逻辑操作
中图分类号: O235 **文献标识码:** A **文章编号:** 0372-2112 (2002) 12A-2146-03

A Method of Pattern Recognition Based on Associative Matrix

ZHOU Jing-zhou
(*Faculty of Information Engineering, Shenzhen University, Shenzhen, Guangdong* 518060, *China*)

Abstract: To reduce memory size and to solve the saturation problem arising from using the associative matrix in the pattern recognition process, on the basis of analyzing the conventional system, we introduce some developments based on logical operations. The aim of these operations is to enhance the linear independence between the input patterns. By presenting examples, we can illustrate that these methods are more efficient and pragmatic.

Key words: pattern recognition; neural network; associative matrix; logical operation

1 引言

神经网络的一个主要功能是模式识别,即对给定的输入模式产生一个与之相关的输出模式.本文提出的方法是在分析布尔神经网络(Boolean neural network) 和相关存储 (associative memory) 基础上的一个行之有效的技术.为此,本节首先概述这两个系统的基本部分,并在适当的分析基础上给出它们之间的联系.在以后的章节中将进一步分析这两个系统的特点和存在的问题,给出一个应用逻辑操作的算法,来解决现有系统所存在的问题,并用实例验证它的有效性和实用性.

布尔神经网络是建立在传统的布尔逻辑门基础上的系统.应用该系统,输入模式(或图像)首先被分为更小的子部分.每一子部分都与一个电子存储设备相连[1].增加这样的分割部分,可使存储空间缩小和使系统更具推广(generation)能力.但这样做同时,系统会趋于饱和(saturation),从而严重地影响输出的正确性.假定每一分割部分有 n 个像素,整个模式的像素个数为 P,则系统所占的存储为:

$$m=\frac{p\times 2^n\times d}{n}$$

这里 d 为区分器(discriminator)的长度,n 的值可取 l 和 p 之间任一数.

由上式可见,当 n 增大时,存储 m 随之增加.与此同时系统的推广能力却随之减弱.另一方面,当 n 减少到 1 或 2 时,存储达到最小值,但系统很快就因为饱和而变成没有任何应用价值.

这里值得一提的是当 $n=0$ 时,我们可以定义存储为

$$m=p\times d$$

在这种情况下,布尔神经网络蜕变为相关存储.相关存储的一个早期例子是学习阵列(learning matrix).Hopfield 网络[2]和双向相关存储(bidirectional associative memory)就是以这样阵列为核心的系统.由于所处理的模式本身可以是任何数据(或信号),这些系统具有很广泛的应用价值.

和布尔神经网络相比,相关存储具有存储小且在某种程度上能克服饱和(saturation)问题的特点,其所用的学习规则[3]为:

$$w=\eta[x]*[y] \tag{1}$$

这里 X 和 Y 分别为输入模式阵列和输出模式阵列.为使问题简单起见,我们通常令 $\eta=1$.阵列 W 的行数等于输入模式的长度.在训练和测试过程中,阵列中的行可由输入模式中的 1 的位置来确定 (寻址).显而易见该阵列地址比布尔神经网络的地址少得多.而且由于该学习过程只涉及一次计算,相比其他算法所用时间也短.

2 对应用相关存储进行模式识别所存在的问题分析

应用上节给出的学习规则(1),在某种程度上饱和问题可以得到解决.但解决这些问题的同时,又会引发其他问题,既导致错误的识别结果.例如:

收稿日期:2002-06-07;修回日期:2002-09-19

令　$\boldsymbol{x}=\begin{bmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{bmatrix}=\begin{bmatrix}0&1&1&1\\0&1&0&1\end{bmatrix}$

则　$\boldsymbol{w}=[\boldsymbol{x}]^T*[\boldsymbol{x}]=\begin{bmatrix}0&0&0&0\\0&2&1&2\\0&1&1&1\\0&2&1&2\end{bmatrix}$

再计算相关的阈值 $\boldsymbol{T}$:

$$\boldsymbol{T}=\begin{bmatrix}0&0\\1&1\\1&0\\1&1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}=\begin{bmatrix}0\\2\\1\\2\end{bmatrix}$$

当输入模式为 $\boldsymbol{x}_2=[0\ \ 1\ \ 0\ \ 1]$时,则有

$$[0\ \ 1\ \ 0\ \ 1]\begin{bmatrix}0&0&0&0\\0&2&1&2\\0&1&1&1\\0&2&1&2\end{bmatrix}=[0\ \ 4\ \ 2\ \ 4]$$

应用阈值 $\boldsymbol{T}$,结果得到模式:

$$[0\ \ 1\ \ 1\ \ 1]$$

显然该结果不是我们所预期的模式 $\boldsymbol{x}_2$.其主要原因是需要满足对于输入模式间线性无关(linearly independent)的要求.

上面给出的例子是自相关存储(auto associative memory)的例子.其意义在于当给出一个不完整或缺损的模式时,系统应能恢复并产生一个完整正确的输出模式.对于异相关存储(hetero associative memory),我们还没见到一个能直接计算阈值的方法.此外,在 Hopfield 网络中,为避免出现不需要的稳定状态,通常需要增加神经元的个数.对于 n 个神经元的网络而言,通常只能存储大约 $p(p=0.15n)$个模式.而且当在模式中通过增加额外的比特(bits)来增加 n 的值时,必须以增加模式间的差异为原则来选择这些比特.其目的是增加输入模式间的线性无关性.然而,对此我们仍没有现成的规则可循.此外是对存储模式的要求,一个纯随机模式集合是不太可能使系统给出相应的正确的响应的.

3　相关存储转换

为解决前面所分析的问题,本文给出了一个基于逻辑操作的方法.首先仍应用前述学习规则产成一个相关存储阵列.然后对于阵列中的成分(权值)进行转换,使之成为反温度计编码(reverse thermometer code)对于权值 i 和它的反温度计编码的定义如下:

i	reverse thermometer code
0	0 0 0 0
1	0 0 0 1
2	0 0 1 1
3	0 1 1 1
4	1 1 1 1

在此基础上将应用一系列操作来响应系统的输入.最后再将这种反温度计编码还原为正常的模式编码作为系统的输出.其还原原理简单,即对于包含有 1 的反温度计编码转换为模式中的一个 1 ;不含有 1 的编码在用模式中用一个 0 来代替.此方法与作者以前的工作[5]相比,极大地节省了存储空间.

4　转换后的相关存储操作

对转换后的相关存储的处理发生在编码阵列的行之间,即对行与行相对应的编码间进行操作.用输入模式中出现的 1 的位置来确定行,然后执行如下操作:

(1)执行由输入模式中 1 的位置所确定的相应编码的逻辑与(AND)操作;

(2)执行由输入模式中 0 位置所确定行中所有编码的逻辑非(NOT)操作;

(3)在由第一步和第二步所得到的结果之间执行逻辑与(AND)操作;

(4)把如上操作结果还原为正常模式形式.

上面第一个操作的目的在于对任一给定输入模式,确定在学习后的相关存贮中对应的信息,第二个操作旨在确定与该输入不相关的信息,第三、四操作是前两操作的合成,从而以一形式化的方法达到增加输入模式间的线性无关性.进行编码转换和执行其后的操作,其作用相当于为系统增加神经元.最后的结果则为所需的输出模式.

为说明如上过程,这里再看第二节中所述的例子.对阵列转换后,执行相应操作,从而得到正确结果.首先把阵列

$$\begin{bmatrix}0&0&0&0\\0&2&1&2\\0&1&1&1\\0&2&1&2\end{bmatrix}$$

转换为:

$$\begin{bmatrix}0&0&0&0\\0&11&01&11\\0&01&01&01\\0&11&01&11\end{bmatrix}$$

当给定的输入模式 $\boldsymbol{x}_2=[0\ 1\ 0\ 1]$,对上述已转换阵列的第二和第四行的对应编码间实施逻辑与操作得到如下编码序列:

$$[0\ \ 11\ \ 01\ \ 11]$$

阵列的第三行取反得:

$$[11\ \ 10\ \ 10\ \ 10]$$

对如上两编码序列执行逻辑与操作得到:

$$[00\ \ 10\ \ 00\ \ 10]$$

最后把此编码序列还原,得到模式[0 1 0 1],即得到正确的 $\boldsymbol{x}_2$ 模式.从而解决了由传统方法导致错误问题.

这里有两点需要强调的:首先,阵列中的全为 0 的行不参加任何操作.因为原始的输入模式暗示,这些行中没有任何信息.其次,如果对任何一个取反操作后而产生全 0 的行,取消对该行所要执行的操作.因为这意味着原始行中各元素全为 1,即所有的存储模式在该行均有信息.只有在遇到信息不完整的输入模式时才会有这种情况出现.而取反操作的目的是根据该模式与其他模式的差异来分辨该模式.而原始的相应完整模式就此位置而言与其他模式没有差异(均为 0 或 1),

所以不执行该操作.总之,这两种情况通常都出现在对不完整模式的处理过程中.

下面再看系统是如何处理不完整测试模式的.在上例中,如输入模式为原始存储模式 $x_1=[0\ 1\ 1\ 1]$一个缺损或多余模式,即[0 1 1 0]或[1 1 1 1].应用相应操作后,我们仍可得到正确的 x_1 模式,即[0 1 1 1].第一种情况我们不应用最后一行进行操作,因为它导至了全0模式结果.第二种情况不应用第一行进行操作,因为该行为一全0行.

为验证该系统在异相关存储下如何正确工作,让我们看如下例:

$$令\ x=\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}1&1&0&0&0\\1&0&1&0&0\\0&1&1&0&0\end{bmatrix}\quad 和$$

$$y=\begin{bmatrix}y_1\\y_2\\y_3\end{bmatrix}=\begin{bmatrix}0&1&0&1&0\\0&1&0&0&1\\1&0&1&0&0\end{bmatrix}$$

则有

$$W=[x]^T*[y]=\begin{bmatrix}0&2&0&1&1\\1&1&1&1&0\\1&1&1&0&1\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}$$

把 W 转换为编码阵列

$$\begin{bmatrix}00&11&00&01&01\\01&01&01&01&00\\01&01&01&00&01\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}$$

当给出 x_3 为输入模式,阵列中第二、三行进行与操作后得编码序列[01 01 01 00 00],第一行取反后得$\overline{x_1}$[11 00 11 10 10].

如上两编码序列进行的与操作后得到编码序列[01 00 01 00 00].再还原为正常模式,得到模式[1 0 1 0 0],即正确的 y_3 模式.

5 结论

本文中给出的算法和操作在占用很小存储空间的基础上解决了在应用神经网络技术进行模式识别中出现的饱和问题.该算法和操作避免了应用其他方法对所要训练模式本身的要求(线性无关性),以及为增加模式中额外的比特而对这些比特的适应选取,而是采用了能体现出识别测试模式和存储模式中相同和不同的本质的逻辑操作,从而使系统更加有效和实用.

参考文献:

[1] Phil Picton. Neural Networks[M]. palgrave, UK, 2000. 50 - 84.

[2] Hopfield J J. Neurons with graded responses have collective computational properties like those f two-state neurons[A]. Proceedings of the National Academy of Science[C]. United States of America: Biophysics, 1984. 81. 3088 - 3092.

[3] Patrick K S. Artificial Neural Systems[M]. Pergamon Press. USA, 1990. 100 - 150.

[4] John H, Anders K, Richard G P. Introduction to the Theory of Neural Computation[M]. USA: Addison-Wesseley Publishing company, 1990. 43 - 63, 89 - 111.

[5] Zhou J Z. Densely coded matrix model of associative memory [J]. Hildeskeimer Informatik-Beriche, Germany, 1995, 22: 1 - 15.

作者简介:

周景洲 男,1945 年 11 月生于哈尔滨,现任深圳大学信息工程学院教授,1982 年于中科院自动化所获硕士学位,1984 年调入深圳大学,1988 年至 1989 年 6 月于香港从事计算网络方面工作,1989 年 6 月至 11 月于法国从事知识工程领域研究工作.1995 年赴德国从事神经网络研究工作,并在德国发表二篇论文.主要研究领域和完成的科研成果为专家系统和神经网络方面课题.