

感知器神经网络加法器的理论研究与设计

徐陈靖^{1,2} 贺勤斌²

XU Chenjing HE Qinbin

摘要

加法器作为数字电路中的核心组件,其运算效率对处理器的整体性能起着至关重要的作用。针对传统加法器存在的效率和延迟方面的瓶颈,文章提出一种基于感知器神经网络的加法器设计方法。首先提出一种全连接的网络结构并设计基于该网络的布尔函数实现算法,然后通过分析加法器的逻辑需求,设计“和”与“进位”功能的网络结构,最终构建出具备完整功能的感知器神经网络加法器架构。实验结果表明,基于感知神经网络的电路设计方法不仅具有可行性,还可以进一步推广应用于一般的逻辑电路设计领域。

关键词

神经网络;感知器;加法器;全连接结构;电路设计

doi: 10.3969/j.issn.1672-9528.2025.02.025

0 引言

传统的逻辑电路主要由“与”“或”和“非”3种基本门电路构成。现代芯片集成度极高,一块芯片中可能包含数十亿至数百亿晶体管以及数百万条线路。全球最先进的CPU制程技术已达到3 nm级别^[1],而硅基芯片的极限精度大约为1 nm。随着科学技术的不断发展,芯片设计领域正面临着巨大的挑战,并进入瓶颈期^[2-3]。因此,研究人员积极探索新工艺、新材料及新设计理念,以突破现有限制。与传统逻辑电路相比,基于感知神经网络的电路设计具备更快的电路响应速度和更小的电路规模。鉴于这些显著优势,感知神经网络电路实现方法有望成为一个全新的芯片设计思路与方法。

本文提出了一种感知器神经网络全连接结构,以及基于该结构的一种布尔函数实现算法,并且完成了基于感知器神经网络的加法器设计。

1 基本概念

1.1 感知器神经网络

神经网络的概念源于生物的神经系统,其运作机制模拟了生物神经元的交互方式。在生物神经系统中,神经元通过树突接收来自其他神经元的信号,并在细胞体中进行加工处理,随后通过轴突将处理后的信号传递给其他神经元。无数神经元相互连接交互作用,构成了一个复杂的网络结构^[4]。1943年,Mcculloch等人^[5]根据生物神经系统的运作模式,抽象出一个著名“M-P模型”,其拓扑结构就是现代神经网络

中的一个神经元。1958年,Rosenblatt^[6]提出了一种可以模拟人类感知能力的机器——感知器,并将其描述为能够在其输入空间中创建分割超平面的数学模型。这些超平面将 n 维空间划分为两个区域,分别对应感知器的两种输出状态:0和1^[7]。单个感知器模型如图1所示。

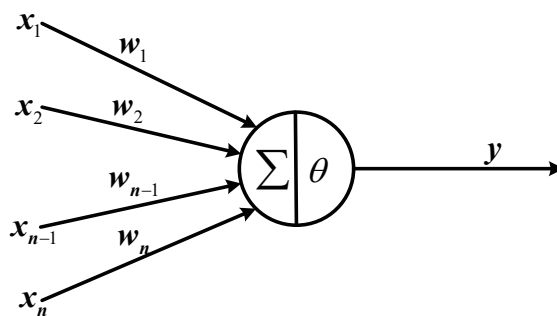


图1 感知器模型

图1为一个 n 输入的单个感知器模型。其中, $(x_1, x_2, \dots, x_n)$ 为输入向量的各个分量, $(w_1, w_2, \dots, w_n)$ 为各个分量的连接权重。其数学表达式为:

$$y = f\left(\sum_{i=1}^n w_i x_i + \theta\right) \quad (1)$$

式中: f 为激活函数(阶跃函数)。

$$f(x) = \text{sign}(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases} \quad (2)$$

单个感知器由于其固有的线性特性,能够轻松实现“与”“或”“非”等简单的逻辑运算。然而,对于非线性可分布尔函数的实现问题,例如异或、同或等逻辑运算,单个感知器则无法实现^[8]。为有效处理这类问题,Rumelhart等人^[9]提出了多层感知器(multi-layer perception, MLP)的结构。

1. 湖州师范学院信息工程学院 浙江湖州 313000

2. 台州学院电子与信息工程学院 浙江台州 318000

该结构的特点是,包括一个输入层、一个或多个隐层以及一个输出层。MLP 结构由于其灵活性和强大的功能,已经被应用于众多领域^[10-11]。

1.2 传统加法器结构

加法器是算术逻辑单元(ALU)的核心组件,专门负责执行二进制加法运算。在传统逻辑电路设计中,加法器通常由全加器和半加器构成。半加器的基本功能是处理两个二进制数的加法运算,输出一个“和”以及一个“进位”。根据半加器的真值表,可以得到其电路结构。然而,半加器仅适用于两位二进制数的简单相加,在处理多位加法时,由于进位的存在,需要使用全加器来处理。相比于半加器,全加器不仅可以处理两个二进制位的相加,还能加上来自前一位的进位,1位全加器真值如表1所示。全加器的逻辑结构如图2所示。

表1 全加器真值表

A	B	低位进位 C_{in}	和 S	高位进位 C_{out}
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

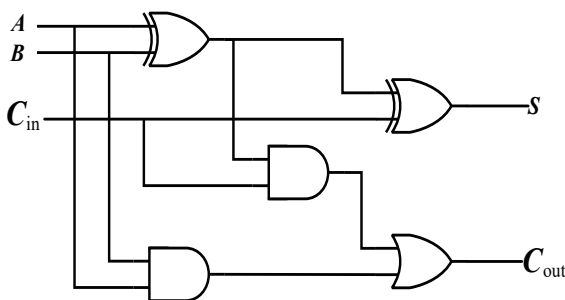


图2 全加器结构图

由全加器真值表可以得到逻辑表达式为:

$$\begin{cases} S = A \oplus B \oplus C_{in} \\ C_{out} = A \cdot B + (A + B) \cdot C_{in} \end{cases} \quad (3)$$

根据图2可知,全加器由两个异或门(用于生成和S),两个与门以及一个或门(用于生成进位 C_{out})组成。

当需要完成 n 位二进制加法时,通常会采用 n 个全加器进行组合。例如行波进位加法器(ripple-carry adder, RCA)以及超前进位加法器(carry-lookahead adder, CLA)^[12]。然而,当处理位数较多时,这些方法在设计难度以及硬件成本方面

会遇到不少问题^[13]。

1.3 布尔函数

一个 n 位的布尔函数(Boolean function)可以看作是集合 $\{0,1\}^n$ 到集合 $\{0,1\}$ 的映射,具体表达式可以写作: $F(U)=v$,其中 $U=(u_1, u_2, \dots, u_n)$ 输入, v 是输出,且 $u_i \in \{0,1\}$, $v \in \{0,1\}$ 。一个 n 输入的布尔函数,全部的输入组合将会有 2^n 个,布尔函数总数有 2^{2^n} 个。可以用输出集 $[v_0, v_1, \dots, v_{2^n-1}]$ 来表示某一个函数 F ,例如二进制表示布尔函数 $F=[01101001]$ 。由于 $v \in \{0,1\}$,因此也可以表示为十进制 $N = \sum_{i=0}^{2^n-1} v_i \cdot 2^i$,例如 $F=150$ 。

按照线性可分性,布尔函数可以被分为线性可分布尔函数(LSBF)和非线性可分布尔函数(non-LSBF)。布尔函数的总数目将随着输入位数 n 的增加,呈指数级增长。目前已知的 $n \leq 9$ 的线性可分布尔函数数目如表2所示^[14-15]。线性可分布尔函数可以使用单个感知器来实现,而非线性可分布尔函数的实现则较为复杂,必须包含至少一个隐层^[16]。但是在设计网络结构来实现各种布尔函数时,隐层的具体规模很难直接确认。因此,给出一种全连接的网络结构来实现所有的布尔函数。

表2 线性可分布尔函数与布尔函数总数的计数

n	线性可分函数数目	布尔函数总数目
1	4	$2^2 = 4$
2	14	$2^{2^2} = 16$
3	104	$2^{2^3} = 256$
4	1882	$2^{2^4} = 65\,536$
5	94\,572	$2^{2^5} = 4\,294\,967\,296$
6	15\,028\,134	$2^{2^6} \approx 1.8 \times 10^{19}$
7	8\,378\,070\,864	$2^{2^7} \approx 3.4 \times 10^{38}$
8	17\,561\,539\,552\,946	$2^{2^8} \approx 1.16 \times 10^{77}$
9	144\,130\,531\,453\,121\,108	$2^{2^9} \approx 1.34 \times 10^{154}$

2 神经网络全连接结构与布尔函数实现算法

本文提出一种三层的神经网络全连接结构。第一层为输入层,由 n 个输入 $(u_1, u_2, \dots, u_n)$ 组成;第二层为隐层,由 m ($m \geq 2$)个 n 输入感知器 $(f_1, f_2, \dots, f_m)$ 组成;第三层为输出层,由一个 m 输入感知器 f_{m+1} 完成结果的输出。网络中每一层的所有感知器均与下一层的每一个感知器存在连接,且不能跨层连接。网络结构图如图3所示。文中所提出的神经网络全连接结构,可以使用 $m+1$ 个感知器实现任意 n 输入的布尔函数。

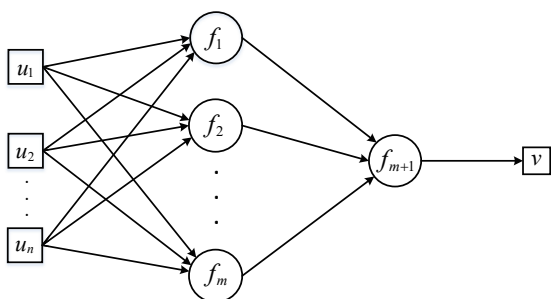


图3 全连接结构

表2中列出了任意 n 输入($n \leq 9$)的布尔函数，进一步建立所有 $n \leq 9$ 的线性可分布尔函数库。利用如图3所提出的全连接结构，可以实现任意 n 输入的非线性可分布尔函数。主要算法过程如下：

首先令 $m=2$ ，然后对隐层和输出层分别遍历所建立的线性可分布尔函数库；若无法实现 n 输入布尔函数 f ，则增加 m 为 $m+1$ ，再次进行遍历，直至实现 n 输入布尔函数 f 。全连接感知器神经网络实现布尔函数算法的伪代码如图4所示。

算法 1: 全连接感知器神经网络实现布尔函数算法

输入: 需要实现的布尔函数 F ，以及布尔函数位数 n

输出: 最终的神经网络结构

$LSBF_lib$ // 线性可分布尔函数库

$num \leftarrow num(LSBF_lib)$ // 库内 $LSBF$ 个数

$m \leftarrow 2$ // m 为隐层感知器个数，初始值设置为 2

$M \leftarrow m + 1$ // M 为感知器总个数

$Neuron[i]$ // $Neuron[i]$ 代表第 i 个感知器的取值编号

$Network_Structure$ // 具体的神经网络结构，返回最终结果

$output$ // 网络结构的结果输出

while True **do**

$flag \leftarrow False$

while True **do**

$output \leftarrow Network_Structure(Neuron)$

if $output$ is correct **then**

 // 得到正确结果

$flag \leftarrow True$

break

end

$Neuron[M] \leftarrow Neuron[M] + 1$

for i in range($M, 1, -1$) **do**

if $Neuron[i] = n$ **then**

$Neuron[i] \leftarrow 0$

$Neuron[i - 1] \leftarrow Neuron[i - 1] + 1$

end

end

if $Neuron[i] = num$ **then**

 // 所有感知器完成遍历

break

end

end

if $flag = True$ **then break;**

else $m \leftarrow m + 1$;

end

图4 全连接感知器神经网络实现布尔函数算法的伪代码

利用文中所提出的算法，可以实现任意非线性可分的布

尔函数。以3输入的布尔函数为例，要实现一个3输入非线性可分布尔函数，只需将目标函数代入到算法中，便可得到所需的感知器，最终确定网络结构。该算法所得到的结果将为构建神经网络加法器提供重要的理论支撑。

3 基于神经网络的加法器设计

3.1 神经网络加法器结构

由表1所示的全加器真值表可知，整个全加器可以看作是3输入布尔函数。 A 、 B 、 C_{in} 分别为3个输入值，需要实现的输出有两个，一个为 S ，另一个为进位 C_{out} （十进制表示分别为150、232）。根据文献[17]的定理1可知，加法器的进位对应的布尔函数是一个线性可分布尔函数。表1中，进位 C_{out} 所代表的布尔函数 $F_2=232=[00010111]$ 的确是一个线性可分布尔函数，可以仅使用一个感知器来实现。要实现布尔函数 $F_1=150=[01101001]$ ，根据所提出的全连接感知器神经网络实现布尔函数的算法，得到函数150可由4个感知器组合实现（3个隐层感知器，1个输出层感知器），十进制表示分别为16、8、142、50。最终结构如图5所示。

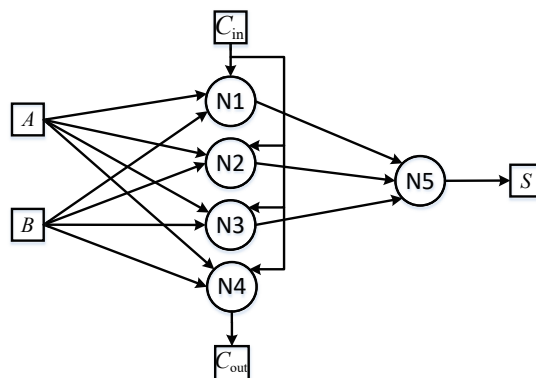


图5 神经网络加法器结构图

图中编号N1~N5分别为5个感知器，整体结构可拆分为两个部分。第一部分由N1、N2、N3、N5号感知器组成，用于实现加法器中和“ S ”。第二部分由一个感知器N4组成，用于实现进位 C_{out} 。最终 S 和 C_{out} 的全连接结构表达如图6和图7所示。程序的运行结果并非唯一。例如N1、N2、N3、N5号感知器的十进制表示还可以为64、212、2、84。

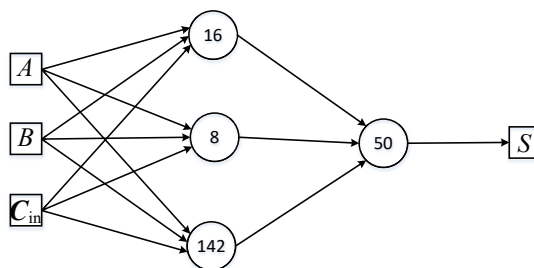


图6 和 S 以及进位 C_{out} 的神经网络表达

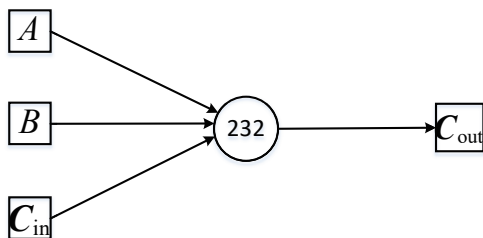


图7 进位 C_{out} 的神经网络表达

由图2的全加器结构图可知，一个全加器模块由5个部分构成。若全部采用与、或、非的基本门电路，由于异或功能需要由两个与门，一个或门以及一个非门构成，总共需要11个门电路才能实现。若采用本文的感知神经网络加法器的设计思想，仅需5个基本元件，即5个神经元就能实现同样的功能。

3.2 感知器的权值和阈值确定

通过文中提出的算法得到了上述所有的感知器，若要进一步进行电路实现，就需要获得其权值以及阈值。这里使用SVM算法确定每个感知器的权值和阈值，如表3所示。

表3 部分感知器的权值和阈值

感知器 (十进制表示)	权值 w_1	权值 w_2	权值 w_3	阈值 θ
8	2	2	-2	-3
16	-2	-2	2	-1
50	2	-4	2	-1
84	-4	2	2	-1
142	2	2	-2	-1
232	2	2	2	-3

4 实验验证

4.1 实验数据准备

实验中所有可能的输入组合有8种，如表4所示。

表4 感知器神经网络加法器的输入数据

输入	1	2	3	4	5	6	7	8
A	0	1	0	1	0	1	0	1
B	0	0	1	1	0	0	1	1
C_{in}	0	0	0	0	1	1	1	1

在前文中，已经通过全连接感知器实现布尔函数算法，得到了感知器神经网络加法器所需具体感知器的编号。此外，还通过SVM算法计算得到了所有感知器的权值以及阈值。接下来将通过具体实验来验证感知器神经网络加法器的正确性。实验中所用到的具体感知器以及权值和阈值如表5所示。

表5 构成加法器感知器的编号

编号	十进制表示	权值 w_1	权值 w_2	权值 w_3	阈值 θ
N1	64	-2	2	2	-3
N2	212	-2	2	2	-1
N3	2	2	-2	-2	-1
N4	232	2	2	2	-3
N5	84	-4	2	2	-1

4.2 实验过程

根据上文所提出的神经网络加法器结构以及相关数据，模拟了整个结构的运算过程，以表4的第5种输入组合[0, 0, 1]为例，结合图5，算法的具体流程如下：

首先N1、N2、N3、N4感知器接收输入数据，由表5可知，感知器N1权值(-2, 2, 2)和阈值 $\theta=-3$ 。根据这些已知参数，即可完成对输入数据的先行求和操作得到-1。根据式(1)和式(2)，得到输出结果为0。整个过程用公式表示为：

$$y = f\left(\sum_{i=1}^n w_i x_i + \theta\right) = f(-2 \times 0 + 2 \times 0 + 2 \times 1 - 3) = \text{sign}(-1) = 0 \quad (4)$$

即N1的输出为0，同理可得N2和N3的输出分别为1和0。N4的输出为0，也就是加法器的进位输出。

最后，N1, N2, N3感知器的输出[0, 1, 0]将作为N5感知器的输入。根据表5，得到N5的权值(-4, 2, 2)和阈值 $\theta=-1$ 。整个运算过程为：

$$y = f\left(\sum_{i=1}^n w_i x_i + \theta\right) = f(-4 \times 0 + 2 \times 1 + 2 \times 0 - 1) = \text{sign}(1) = 1 \quad (5)$$

即N5的输出为1，也就是加法器的“和”。将表4的8种输入组合全部按上述过程进行计算，得到如表6所示的最终结果。

表6 神经网络加法器结构模拟运算结果

编号	A	B	C_{in}	N5 输出	N4 输出
1	0	0	0	0	0
2	1	0	0	1	0
3	0	1	0	1	0
4	1	1	0	0	1
5	0	0	1	1	0
6	1	0	1	0	1
7	0	1	1	0	1
8	1	1	1	1	1

对照表6中的实验结果与表1的全加器真值表可知，模拟运算的结果与真值表中的值一致。实验结果表明，本文提出的感知器神经网络加法器能够正确有效地实现全加器的功能。

5 结语

本文基于感知器神经网络和布尔函数的相关概念,提出一种全连接结构的感知器神经网络,以及一种基于全连接感知器神经网络的布尔函数实现算法。运用该算法,仅需4个感知器即可实现所有3输入的非线性可分布尔函数。然后,利用全连接结构对加法器进行设计,成功使用5个感知器完成了一位全加器的功能模块,比传统全加器结构所需的基本元件更少。这种思想可以进一步推广到实现更多输入位的复杂布尔函数。例如设计一个64位加法器,总共有128位加上1位进位,129位的输入。若采用全连接的网络结构设计加法器,如果可以用129个隐层感知器,加上输出层的1个感知器组成表示一个和“S”,那么在理想情况下,最多需要 $64 \times 130 = 8320$ 个感知器就能实现该加法器。

本文以加法器的设计为例,介绍了利用感知器神经网络构建加法器的应用。这一设计思路同样适用于其他电路设计,包括复杂的非线性电路设计。对于这些电路,传统设计方法往往难以应对其内在的复杂性和变化性。感知器神经网络通过其独特的结构和算法,能够将电路功能目标转化为神经网络的训练目标,并通过调整神经连接的权值来优化电路参数,显著提高设计的效率和准确性。本文提出的利用神经网络设计逻辑电路的方法提供了新的研究方向,对逻辑电路设计领域具有重要的理论价值和实际应用潜力。

参考文献:

- [1] 邓中翰. 集成电路技术综述[J]. 集成电路与嵌入式系统, 2024, 24(1): 1-12.
- [2] 高彦钊, 邬江兴, 刘勤让, 等. 计算体系架构研究综述与思考[J]. 中国科学: 信息科学, 2022, 52(3): 377-398.
- [3] 张梅娟, 张明月, 杨楚玮, 等. SoC 芯片的 RomCode 设计与 FPGA 验证研究[J]. 电子设计工程, 2023, 31(21): 76-80.
- [4] ANDREE H M A, BARKEMA G T, LOURENS W, et al. A comparison study of binary feedforward neural networks and digital circuits[J]. Neural networks, 1993, 6(6): 785-790.
- [5] MCCULLOCH W S, PITTS W. A logical calculus of the ideas immanent in nervous activity[J]. The bulletin of mathematical biophysics, 1943, 5: 115-133.
- [6] ROSENBLATT F. The perceptron: a probabilistic model for information storage and organization in the brain[J]. Psychological review, 1958, 65(6): 386-408.
- [7] 张驰, 郭媛, 黎明. 人工神经网络模型发展及应用综述[J]. 计算机工程与应用, 2021, 57(11): 57-69.
- [8] MINSKY M, PAPERT S. Perceptron[J]. American journal of psychology, 1969, 84(3): 449-452.
- [9] RUMELHART D E, HINTON G E, WILLIAMS R J. Learning representations by back-propagating errors[J]. Nature, 1986, 323: 533-536.
- [10] 吴志豪, 张德军, 吴亦奇, 等. 一种基于 MLP 的高效高精度三维视线估计方法[J]. 计算机工程与科学, 2023, 45(11): 1982-1990.
- [11] 刘昌平, 刘海, 夏梦, 等. 俄罗斯方块的 Hamming 神经网络模型[J]. 计算机与现代化, 2020(2): 51-54.
- [12] 郭佳兴, 王金梅, 韩国英. 高性能全加器电路版图优化设计研究[J]. 宁夏电力, 2023(2): 51-58.
- [13] 周冉冉, 周文宸, 王永. 一种基于 55nm 工艺的超前进位加法器设计[J]. 中国集成电路, 2023, 32(8): 49-53.
- [14] DURAIVEL N, PAULCHAMY B. Simulation and performance analysis of 15 Nm FinFET based carry skip adder[J]. Computational intelligence, 2020, 37(2): 799-818.
- [15] HE Q B, CHEN F Y, JIN W. Topological equivalence classification and enumeration of n-input linearly separable Boolean functions[J]. Information sciences, 2024, 658: 120020.
- [16] 贺勤斌, 刘曾荣. 非线性可分 Boolean 函数分解及神经网络实现[J]. 科学技术与工程, 2010, 10(11): 2646-2649.
- [17] CHEN F Y, WANG G Y, CHEN G R, et al. A novel neural network parallel adder[J]. Advances in computational intelligence, 2013, 7902: 538-546.

【作者简介】

徐陈靖(2000—), 男, 江苏南通人, 硕士研究生, 研究方向: 布尔函数、神经网络, email: xuchenjing320@163.com.

贺勤斌(1972—), 通信作者(email: heqinbin@126.com), 男, 浙江台州人, 博士, 教授, 研究方向: 布尔函数、神经网络、系统生物学。

(收稿日期: 2024-10-31)