目录

[摘要 2](#_Toc514254962)

[Abstract 2](#_Toc514254963)

[1. 绪论 3](#_Toc514254964)

[1.1 选题目的及意义 3](#_Toc514254965)

[1.2 国内外论文综述 3](#_Toc514254966)

[1.3 本论文所做的主要工作 3](#_Toc514254967)

[1.4 MNIST数据集 4](#_Toc514254968)

[2. 总体设计 7](#_Toc514254969)

[3. 神经元模块设计与实现 9](#_Toc514254970)

[4. 处理核心 10](#_Toc514254971)

[5.神经网络模块设计与实现 11](#_Toc514254972)

[6.实验结果 11](#_Toc514254973)

[结论 11](#_Toc514254974)

摘要

深度学习作为一类主流的机器学习算法，在图像识别、语音识别以及自然语言处理等多个领域有着广泛应用。然而深度学习应用程序具有计算量大、访存密集且不规则等特点，传统的CPU和GPU等多核体系结构受自身限制，无法有效满足深度学习应用的硬件需求。而FPGA作为一种灵活高效的半定制化硬件技术，被学术界广泛选作面向特定应用的加速器平台技术。发展基于FPGA的深度学习应用加速器能有效改善深度学习应用的性能与功耗，从而提高深度学习技术的可用性和可扩展性。

通过实验分析深度学习应用的基本计算和数据访问特征；掌握FPGA HLS（高层次综合）技术，了解基本的FPGA加速器设计方法与技巧；在选定的FPGA开发板中设计实现面向深度学习应用的FPGA加速器原型，并对其性能与功耗进行分析比较。最终实现面向深度学习应用的FPGA加速器原型设计。

关键词： 神经网络；深度学习；FPGA；图像识别

Abstract

As a mainstream machine learning algorithm, deep learning has been widely used in many fields such as image recognition, speech recognition and natural language processing. However, deep learning applications have the characteristics of large amount of calculation, intensive and irregular access, and the traditional multi-core architectures such as CPU and GPU are limited by themselves and cannot effectively meet the hardware requirements of deep learning applications. As a flexible and efficient semi-customized hardware technology, FPGA has been widely selected as an accelerator platform tech-

nology for specific applications by the academic community. The development of FPGA-based deep learning application accelerators can effectively improve the performance and power consumption of deep learning applications, there-

by improving the usability and scalability of deep learning technologies.

Exploring basic calculations and data access features of deep learning app-

lications through experiments; mastering FPGA HLS (high-level synthesis) techniques, understanding basic FPGA accelerator design methods and tech-

niques; designing FPGAs for deep learning applications in selected FPGA de-

velopment boards Accelerator prototype, and analyze and compare its perfor-

mance and power consumption. Finally, FPGA accelerator prototyping for de-

ep learning applications is achieved.

Keywords： neural net ; Deep learning ; FPGA ; Image recognition

1. 绪论

1.1 选题目的及意义

神经网络的深度学习的概念由Hinton等人于2006年提出，概念源于人工神经网络的研究。建立、模拟人脑进行分析学习的神经网络，它模仿人脑的机制来解释数据，例如图像，声音和文本。基于深信度网(DBN)提出非监督贪心逐层训练算法，为解决深层结构相关的优化难题带来希望，随后提出多层自动编码器深层结构。此外Lecun等人提出的卷积神经网络是第一个真正多层结构学习算法，它利用空间相对关系减少参数数目以提高训练性能。

FPGA作为一种灵活高效的半定制化硬件技术，被学术界广泛选作面向特定应用的加速器平台技术。发展基于FPGA的深度学习应用加速器能有效改善深度学习应用的性能与功耗，从而提高深度学习技术的可用性和可扩展性。

1.2 国内外论文综述

1.2.1

[1]中提出了一种新的用于NN加速器的领域专用指令集架构（ISA），称为Cambricon，它是一种集标量、向量、矩阵、逻辑、数据传输和控制指令于一体的加载-存储架构。

1.3 本论文所做的主要工作

通过实验分析深度学习应用的基本计算和数据访问特征；掌握FPGA HLS（高层次综合）技术，了解基本的FPGA加速器设计方法与技巧；在选定的FPGA开发板中设计实现面向深度学习应用的FPGA加速器原型，并对其性能与功耗进行分析比较。最终实现面向深度学习应用的FPGA加速器原型设计。

1.4 MNIST数据集

（参数，结构，文字描述，训练测试，样本大小，数据格式，原始数据怎么输入）、MATLAB实现过程

MNIST是一个手写数字数据库，是NIST数据库的一个子集。

本论文中所用的两个文件不是标准的图像格式，图像数据都保存在解压后得到的二进制文件中。每个样本图像的宽高为28\*28，也就是说每一张图片包含 28 \*28 个像素点，可以用一个数组来表示这张图片。每一张图片都有其对应的标签，就是图片对应的数字。

**1.5感知机网络**定义，引用文献说明可行性，结构3层怎么组成、参数）

神经网络是神经元互相连接构成的一个非循环的图，也就是说一些神经元的输出会作为其他神经元的输入，另外环路是不允许的因为这会使得神经网络的前向传播陷入无止尽的循环中。当然，神经元之间的排列是有规律的，通常情况下被构建成层层连接的形式，每一层中又有多个神经元。通俗的来讲，神经网络就是一个学习器，给它一组输入，它会得到一组输出，神经网络里的节点相互连结决定了输入的数据在里面经过怎么样的计算。神经网络具有这个功能：通过大量的输入，神经网络调整自身的连接情况，给出我们的预期输出。

任何多层的网络结构必须有一个输入层、一个输出层，隐藏层可多可少。输入层与输出层的节点数往往是固定的，隐藏层则可以自由指定。一个三层的神经网络结构如下图1.5.1：

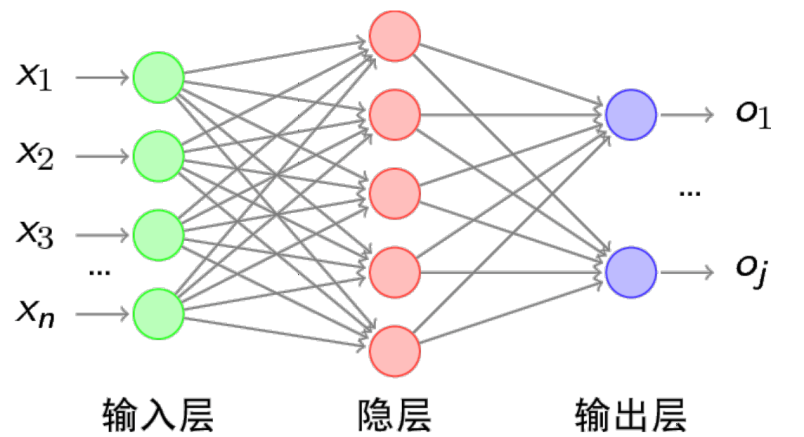


图1.1神经网络结构图

节点也就是神经元。多层网络，顾名思义就是有多个节点层结构组成的网络系统，它的每一层都是由若干神经元节点构成，该层的任意一个节点和上一层的每一个节点相连，由它们来提供输入，经过计算产生该节点的输出并作为下一层节点的输入。

**1.5 用FPGA加速神经网络**为什么要使用，怎么实现，用FPGA怎么加速神经网络

FPGA(Field-Programmable Gate Array)，即现场可编程门阵列，是在PAL、GAL、CPLD等可编程器件的基础上进一步发展的产物。它是作为专用集成电路(ASIC)领域中的一种半定制电路而出现的，既解决了定制电路的不足，又克服了原有可编程器件门电路数有限的缺点。以硬件描述语言（Verilog或VHDL）所完成的电路设计，可以经过简单的综合与布局，快速的烧录至 FPGA 上进行测试。FPGA主要有以下几个特点：

（1）采用FPGA设计ASIC电路(专用集成电路)，用户不需要投片生产，就能得到合用的芯片。

（2）FPGA可做其它全定制或半定制ASIC电路的中试样片。

（3）FPGA内部有丰富的触发器和I/O引脚。 · ··

（4）FPGA是ASIC电路中设计周期最短、开发费用最低、风险最小的器件之一。

（5) FPGA采用高速CMOS工艺，功耗低，可以与CMOS、TTL电平兼容。

可以说，FPGA芯片是小批量系统提高系统集成度、可靠性的最佳选择之一。

FPGA是由存放在片内RAM中的程序来设置其工作状态的，因此，工作时需要对片内的RAM进行编程。用户可以根据不同的配置模式，采用不同的编程方式。

加电时，FPGA芯片将EPROM中数据读入片内编程RAM中，配置完成后，FPGA进入工作状态。掉电后，FPGA恢复成白片，内部逻辑关系消失，因此，FPGA能够反复使用。FPGA的编程无须专用的FPGA编程器，只须用通用的EPROM、PROM编程器即可。当需要修改FPGA功能时，只需换一片EPROM即可。这样，同一片FPGA，不同的编程数据，可以产生不同的电路功能。因此，FPGA的使用非常灵活。

FPGA以并行运算为主，以硬件描述语言来实现。

目前,卷积神经网络主要基于通用处理器实现,但基于软件方式无法充分挖掘卷积神经网络的并行性，在实时性和功耗方面都不能满足应用的需求。由于FPGA计算资源丰富、灵活可配、开发周期短，越来越多研究者喜欢采用FPGA开发基于卷积神经网络的应用。

**3.2 FPGA芯片Zynq-7000** why使用，特点，结构（PS,PL，AXI协议通信），

Zynq-7000系列FPGA继承了两个1 GHz 的ARM CortexA9内核，打破了传统的FPGA+ARM/DSP的架构，使用单片FPGA就能很好地完成工作。FPGA和ARM通过高达100 Gbps的内部高速总线通信，比FPGA+ARM/DSP外部通信的架构更加迅速而且更加可靠。带有DDR控制器硬核，支持1 GB多种数据位的地址宽度，在ARM（PS）端有64位的数据通道，最高支持1 066 MT/s的速度,其在速度、稳定性和泛用性上都非常优秀，特别适用于做系统的高速存储。

与传统的FPGA相比，Zynq-7000系列最大的特点是将处理系统PS和可编程资源PL分离开来，固化了PS系统的存在，实现了真正意义上的SOC（System On Chip）。同时，这种软硬件均可编程的全可编程 SoC集成了ARM®处理器的软件可编程性与FPGA的硬件可编程性，不仅可实现算法的硬件加速，还在单个器件上高度集成 CPU、DSP slice, 高速收发器以及模拟信号处理等功能。是单位功耗性价比最高的全面可扩展的 SoC 平台。

Zynq-7000系列是全可编程片上系统，主要包含PS（processing system）和PL（Programmable Logic）两部分。

PL和PS的接口类型总共有两种：（1）功能接口：AXI、EMIO、中断、DMA流控制、时钟调试接口。（2）配置接口：PCAP、SEU、配置状态信号和Program/Done/Init信号。这些信号连接到PL内配置模块的固定逻辑上，给PS提供对PL的控制能力。

PS和PL部分之间有多个接口，具体包括：

1、AXI类数据接口：2个32bit的AXI 主接口、2个32bit的AXI从接口、专用于PL访问DDR控制器的32/64bit的AXI从接口、1个64bit的访问CPU存储器的从接口

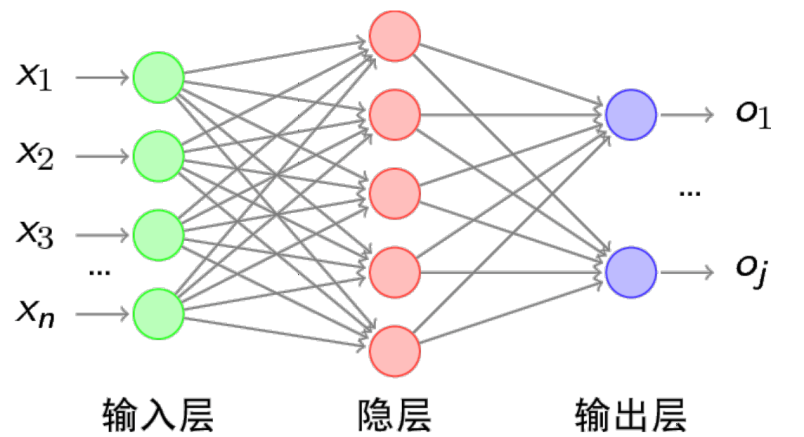
2、其他接口：DMA通道信号、PS的中断输入信号、事件信号、触发信号、EMIO、PS提供给PL的时钟信号及复位信号、XADC接口、JTAG接口

AXI总线：AXI （Advanced eXtensible Interface）本是由ARM公司提出的一种总线协议。用于PL和PS之间的通信。

2. 总体设计

**2.1 神经网络设计**实现神经网络的流程，画模块框图（主要参数，接口，端口之间的连线，端口的数据类型（位））

画图：lucidchart（主要参数，接口，端口之间的连线，端口的数据类型（位））



实现神经网络首先要实现单个神经元，其次在单个神经元的基础上实现层，最后整合多个层实现神经网络。

一个神经网络的训练算法就是让权重的值调整到最佳，以使得整个网络的预测效果最好。

**2.2 激活函数**（定义，为什么需要激活函数，类型（常用的如sigmoid，特点（曲线））

激活函数是神经网络的一个重要组成部分，代表了轴突接收到冲激信号的频率。它对于人工神经网络模型去学习、理解非常复杂和非线性的函数来说具有十分重要的作用。它们将非线性特性引入到我们的网络中。其主要目的是将NN模型中一个节点的输入信号转换成一个输出信号。该输出信号现在被用作堆叠中下一个层的输入。在NN中的具体操作是这样的，我们做输入（X）和它们对应的权重（W）的乘积之和，并将激活函数f（x）应用于其获取该层的输出并将其作为输入送到下一层。

激活函数通常有如下一些性质：

• 非线性： 当激活函数是线性的时候，一个两层的神经网络就可以逼近基本上所有的函数了。但是，如果激活函数是恒等激活函数的时候（即），就不满足这个性质了，而且如果MLP使用的是恒等激活函数，那么其实整个网络跟单层神经网络是等价的。

•可微性：当优化方法是基于梯度的时候，这个性质是必须的。

•单调性：当激活函数是单调的时候，单层网络能够保证是凸函数。

•： 当激活函数满足这个性质的时候，如果参数的初始化是random的很小的值，那么神经网络的训练将会很高效；如果不满足这个性质，那么就需要很用心的去设置初始值。

• 输出值的范围： 当激活函数输出值是有限的时候，基于梯度的优化方法会更加稳定，因为特征的表示受有限权值的影响更显著；当激活函数的输出是无限的时候，模型的训练会更加高效，不过在这种情况小，一般需要更小的learning rate.

在神经网络中，激活函数的作用是能够给神经网络加入一些非线性因素，使得神经网络可以更好地解决较为复杂的问题。如果不运用激活函数的话，则输出信号将仅仅是一个简单的线性函数。线性函数一个一级多项式。现如今，线性方程是很容易解决的，但是它们的复杂性有限，并且从数据中学习复杂函数映射的能力更小。一个没有激活函数的神经网络将只不过是一个线性回归模型（Linear regression Model）罢了，它功率有限，并且大多数情况下执行得并不好。同样是因为没有激活函数，神经网络将无法学习和模拟其他复杂类型的数据，例如图像、视频、音频、语音等。这就是为什么要使用人工神经网络技术，诸如深度学习（Deep learning），来理解一些复杂的事情，一些相互之间具有很多隐藏层的非线性问题，而这也可以帮助了解复杂的数据。

本论文选用的是S型激活函数sigmoid函数，是常用的非线性的激活函数：

. ()

其对应的图像如下图：

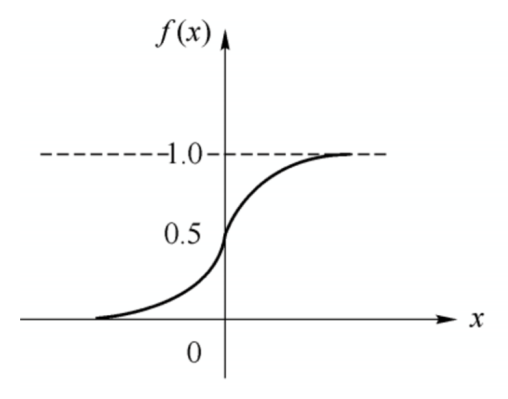


图2. sigmoid 函数图

输入一个实值的数，然后将其压缩到0~1的范围内，是一个S形曲线。特别地，大的负数被映射成0，大的正数被映射成1。它能够很好的表达“激活”的意思，未激活就是0，完全饱和的激活则是1。

Sigmoid函数的输出映射在(0,1)之间，单调连续，输出范围有限，优化稳定，可以用作输出层；求导容易。但由于其软饱和性，容易产生梯度消失，导致训练出现问题；且其输出并不是以0为中心的。

在FPGA中实现激活函数，激活函数怎么在FPGA中实现(在FPGA中怎么表达定点数（位宽和小数部分位宽），

查找表（为什么用，怎么用）)

查找表是用简单的查询操作替换运行时计算的数组或者 associative array 这样的数据结构。由于从内存中提取数值经常要比复杂的计算速度快很多，所以这样得到的速度提升是很显著的。尽管查找表经常效率很高，但是如果所替换的计算相当简单的话就会得不偿失，这不仅仅因为从内存中提取结果需要更多的时间，而且因为它增大了所需的内存并且破坏了高速缓存。如果查找表太大，那么几乎每次访问查找表都会导致高速缓存缺失，这在处理器速度超过内存速度的时候愈发成为一个问题。

**2.3 Zynq-7000芯片**具体用的芯片型号（特点

**5.3单元测试模块的设计**

**5.4本章小结**

3. 神经元模块设计与实现

**4.2神经元的端口**神经元的端口（用列表：方向，数据类型，位宽，描述）

神经元是神经网络中最基本的结构，也可以说是神经网络的基本单元，它的设计灵感完全来源于生物学上神经元的信息传播机制。神经元模型是一个包含输入、输出与计算功能的模型。输入可以类比为神经元的树突，而输出可以类比为神经元的轴突，计算则可以类比为细胞核。连接是神经元中最重要的东西。每一个连接上都有一个权重。一个典型的神经元模型如下图（包含3个输入，1个输出， 2个计算功能。）：

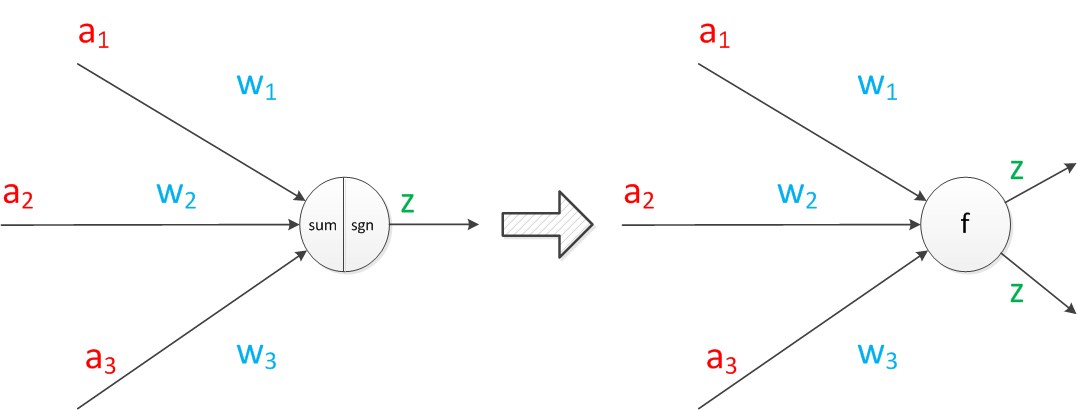


图3.1 神经元模型

a1、a2、a3表示三个输入，w1、w2、w3表示三个连接（中间的箭头线）上的权重，用f代表神经元的内部计算（sum函数与sgn函数），z表示输出。一个神经元可以引出多个代表输出的有向箭头，但值都是一样的。

4.3及神经元中的状态机（几种状态（表或描述），状态迁移图（主要触发条件）

神经元有一般两种状态：兴奋和抑制。一般情况下，大多数的神经元是处于抑制状态，但是一旦某个神经元受到刺激，导致它的电位超过一个阈值，那么这个神经元就会被激活，处于“兴奋”状态，进而向其他的神经元传递信息。

本论文将神经元的状态细分为空闲、忙、已完成。

4.4计算过程（数学公式））

神经元可以看作一个计算与存储单元。计算是神经元对其的输入进行计算功能。存储是神经元会暂存计算结果，并传递到下一层。

如上图所示的神经元内部计算为：

. ()

4. 处理核心

**.1**

**.2**

**.3**

**本章小结**

5.神经网络模块设计与实现

7.1 神经网络结构（输入输出，结构）

7.2单元测试模块的设计

7.3本章小结

6.实验结果

结论

**参考文献**

[1] Shaoli Liu , Zidong Du , Jinhua Tao, Dong Han, Tao Luo, Yuan Xie†, Yunji Chen and Tianshi Chen，《Cambricon: An Instruction Set Architecture for Neural Networks》， ACM/IEEE第43届国际计算机体系结构研讨会，2016

[2]

[3]

[4]

[5]

[6]

[7]

[8]

[9]

[10]

[11]

[12]

[13]

[14]

[15]

[16]

[17]

[18]

[19]

[20]

[21]