ZYNQ的卷积神经网络硬件加速通用平台设计*

冯光顺，应三丛

（四川大学 计算机学院，成都 610065）

摘要：近年来卷积神经网络（CNN）在人工智能领域备受关注，被越来越多应用到实际生产中。为了较好地实现工程应用，需要将算法固化到嵌入式平台上。由于卷积神经网络的数据计算并行度高、计算量大，现场可编程门阵列成为对其进行硬件加速的重要工具。本文基于Xilinx ZYNQ ZC706设计实现了卷积神经网络硬件加速的通用平台，可以满足不同卷积神经网络算法模块实现硬件加速的需求。

关键词：ZC706；卷积神经网络；硬件加速；FPGA；ZYNQ

中图分类号：TP391　　**文献标识码**：A

Design of Hardware Acceleration General Platform for CNN Based on ZYNQ

Feng Guangshun，Ying Sancong

（School of Computer Science，Sichuan University，Chengdu 610065，China）

Abstract：Convolutional Neural Network is attracted more attention recently in AI field，more and more applications are used in industry field using it. In order to enhance the application value，it is meanful to implement algorithms on embedded systems. Due to the large computing quantity and high degree of parallelism，FPGA has become a conspicuous tool as a hardware accelerator of algorithms. In this paper，a novel hardware platform is designed to accelerate CNN algorithms by Verilog HDL on Xilinx ZYNQ ZC706 board，which meets the needs of different kinds of implementation of CNN design modules.

Key words：ZC706；convolutional neural network；hardware acceleration；FPGA；ZYNQ

引　言

随着计算机技术的不断发展和对人工智能领域的深入研究，卷积神经网络成为近年来的研究热点之一，在图像分类、目标检测、图像语义分割等诸多领域取得一系列突破性研究成果[1]。卷积神经网络（CNN）的智能化程度和处理能力随着深度学习技术的发展不断提高，但同时CNN的结构越来越复杂，数据规模越来越大，导致计算密集度越来越大。基于通用处理器实现的CNN计算时间过长，已不能满足实际应用的需求。采用GPU和多核CPU对CNN进行加速，则会导致系统功耗太大，不适合低功耗的嵌入式系统。FPGA作为一种可编程逻辑器件，不仅可以实现低功耗的高性能计算，还能充分挖掘CNN内部固有的并行性[2]。

Xilinx公司推出的ZYNQ-7000全可编程片上系统采用了ARM＋FPGA的异构架构，在单芯片上集成了处理系统（Processing System，PS）和可编程逻辑（Programmable Logic，PL）两大功能模块[3-4]，两部分以高速片上总线AXI（Advanced eXtensible Interface）互联，保证系统的处理带宽。基于XC7Z045-2FFG900C芯片的ZC706是一款高性能、低功耗的开发套件，它包含了所有必须的接口，是一款理想的设计验证平台。本文基于ZC706设计实现了CNN硬件加速通用平台，可以满足不同CNN算法模块实现硬件加速的需求。

* 基金项目：四川省科技厅科技支持项目（2016GZ0097）。

1　CNN结构介绍

典型的CNN由输入层、卷积层、池化层、全连接层及输出层构成，往往采用交替连接的卷积层和池化层对输入图像进行前向传播，最后经全连接和输出层进行概率分布输出。

图1所示为LeNet-5[5]的网络结构图，其中以字母C开头的为卷积层，以字母S开头的为池化层，以字母F开头的为全连接层。卷积层对输入的特征图谱（或输入层的图像）进行卷积运算，提取特征，输出新的特征图谱；池化层对卷积层输出的特征图谱进行降采样，缩减特征图谱尺寸；全连接层将输入数据经过若干个指定维度的滤波器进行加权求和，得到若干维输出；输出层通常使用softmax函数计算每个类别的概率[6]。在各层中，卷积层是运算量

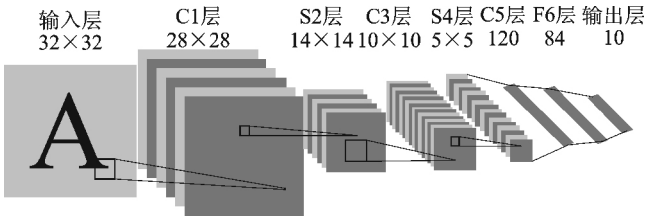


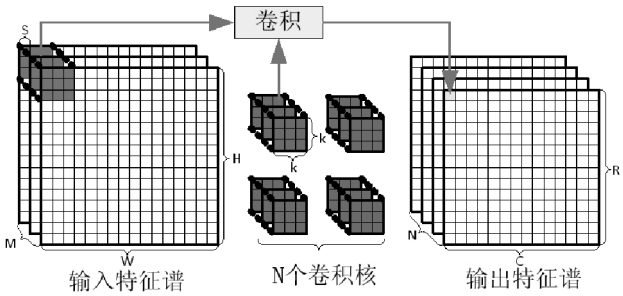
图 1　LeNet-5 网络结构图

最大和并行度最高的层，也是 FPGA 硬件加速的主要部分。卷积层的形式一般如式(1)所示：

$$x_j^l = f\left(\sum_{i \in M_j} x_j^{l-1} * k_{ij}^l + b_j^l\right) \tag{1}$$

式中，l 表示层数，k 为卷积核，M_j 表示输入特征谱的一个选择，$*$ 表示卷积运算，b 表示输出特征谱的偏置。

图 2 为卷积运算的软件实现方法，假设步长 S 为 1，M




```
for(row = 0; row < H; row += S)      //row_loop
  for(col = 0; col < W; col += S)    //col_loop
    for(pi = 0; pi < M; ++pi)        //pi_loop
      for(po = 0; po < N; ++po)      //po_loop
        for(i = 0; i < k; ++i)       //kernel_loop
          for(j = 0; j < k; ++j)
          {
            output[row][col][po] +=
            kernel[pi][po][i][j] *
            input[row+i][col+j][pi];
          }
```

图 2　卷积运算的软件实现方法

个 W×H 的输入特征谱，与 N 个 k×k 卷积核进行卷积运算，最终得到 N 个 C×R 输出特征谱。代码中使用了 6 层循环嵌套，可见其运算量非常大，同时像素点之间的计算是互相独立的，体现了卷积运算内部的并行性，这与 FPGA 的运算特性非常契合。

2　CNN 硬件加速通用平台设计

FPGA 很适合对 CNN 作加速运算，但运算所需的图像和权值参数以及运算完成后的结果数据，都需要特定的通道进行传输，针对该问题本文基于 ZYNQ-7000 设计了适用于使用 FPGA 对 CNN 算法进行硬件加速的通用平台，可以使 CNN 算法硬件加速模块快速地进行板级验证。

CNN 硬件加速通用平台结构原理图如图 3 所示，左上方为 ZYNQ 的 PS 部分，阴影区为 PL 部分。在 PS 端以 ARM 处理器为中心对数据流和运算进行控制；在 PL 端(即 FPGA 部分)搭建了直接内存存取模块(即 DMA)作为 PS 与 PL 之间数据传输的高速通道，使用片上存储器 Block RAM(简称 BRAM)存储数据。本文设计了 AXI_CCIF(AXI CNN Chip Interface Module)模块，该模块给 CNN 硬件加速器(CNN Hardware Accelerator，CNNHA)提供了通用的数据通道。

ARM 通过 IP 网络接口从上位机获取图片和权值参数数据暂存在 DDR3 Memory 中，启动 DMA Engine，将图像和权值参数通过 DMA 通道传输到 PL 端的 BRAM 中。根据图片和权值参数的数量，ARM 对 AXI_CCIF 的内部相应寄存器进行配置，各项参数配置完成后，ARM 通过 AXI_CCIF 的控制寄存器启动 CNN 运算。接收到启动运算的命令后，CNNHA 模块从 AXI_CCIF 的图像通道(Image Channel)和权值参数通道(Weight Channel)读取所需的图像和权值参数数据进行运算，同时将运算结果写入结果通道(Result Channel)，AXI_CCIF 再把接收到的运算结果存储在 BRAM 中。整个运算结束后，AXI_CCIF 向 ARM 发起中断，ARM 再次启动 DMA Engine，将运算结果传输到 PS 端的 DDR3 Memory 中，ARM 通过 IP 网络接口将运算结果传输给上位机，作进一步处理。

3　AXI_CCIF 的实现

3.1　AXI_CCIF 结构原理图

AXI_CCIF 是平台的核心设计模块，可根据不同 CNN 的需求进行配置，是平台通用性的重要体现，给 CNNHA 提供了 3 个统一的数据通道。AXI_CCIF 结构原理图如图 4 所示，包括 AXI 接口模块、寄存器模块、中断模块、数据通道模块及 BRAM 接口模块 5 个部分。AXI 接口模块即 AXI4 总线协议模块，是 ARM 访问其内

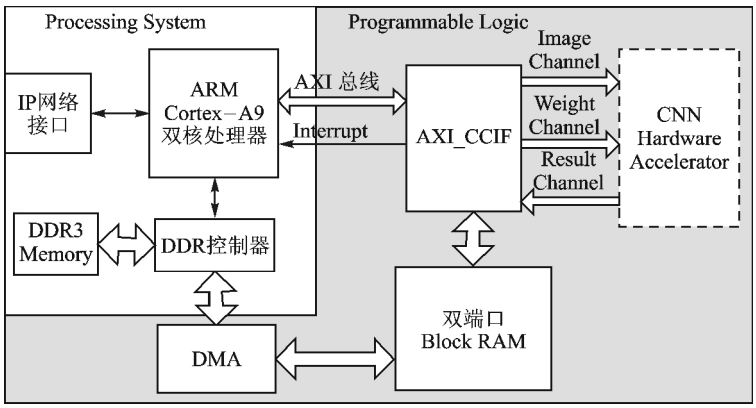


图 3　CNN 硬件加速通用平台结构原理图

部功能寄存器的通信接口；寄存器模块由 7 个 32 位的寄存器组成，经 AXI 互联模块映射到 ARM 的地址空间，可由 ARM 直接访问；中断模块是 AXI_CCIF 向 ARM 反馈信息的通信接口；数据通道模块由图片、权值参数及运算结果三个数据通道组成，为 CNNHA 模块提供了通用的数据交互接口；BRAM 接口模块与双端口 BRAM 相连，实现数据存取功能。

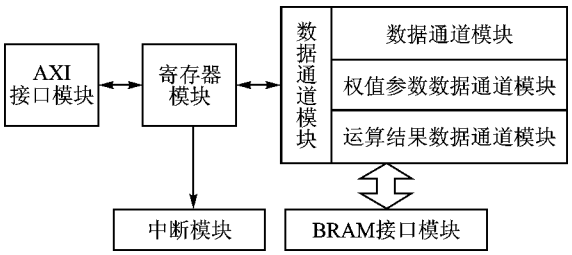


图 4　AXI_CCIF 结构原理图

3.1.1　寄存器模块

寄存器模块包括 7 个 32 位的功能寄存器：reg0_Ctrl（控制寄存器），通过写寄存器特定功能位可以复位功能模块和启停 CNN 硬件加速运算；reg1_Status（状态寄存器），标记平台的工作状态；reg2_ParamLen（参数长度寄存器），其数值表示在本次运算中权值参数的总个数；reg3_ImgReso（图像分辨率寄存器），低 16 位和高 16 位分别存储图像的横向和纵向分辨率；reg4_ImgLen（图片长度寄存器），其数值表示在本次运算中图片数据的总个数；reg5_ResultLen（结果长度寄存器），CNN 运算完成后会有若干个运算结果，该寄存器的值表示运算结果的总个数；reg6_ImgReTxTimes（图片重复发送次数寄存器），CNN 的第一个卷积层往往要得到若干个特征图谱，每张特征图谱都需要重新读取一遍图像数据，该寄存器的值则表示重复的次数。

3.1.2　数据通道模块

数据通道模块是 AXI_CCIF 与 CNNHA 进行数据交互的关键模块，包括图像、权值参数及结果三个数据通道模块。图像与权值参数数据通道模块功能类似，数据从 AXI_CCIF 流向 CNNHA，相当于读数据，其接口时序图如图 5 所示，rd_en 为读使能信号，dout 为数据输出总线。当 rd_en 信号有效时，输出图像或权值参数数据。last 信号指示图像或权值参数已传输完，须保持 8 个时钟周期有效，确保被 CNNHA 接收到。

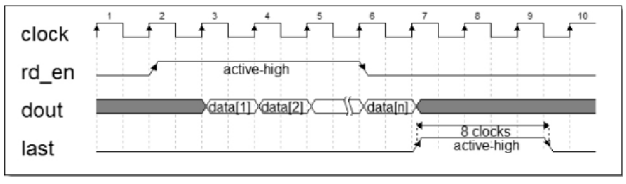


图 5　图像与权值参数数据通道接口时序图

结果数据则是从 CNNHA 流向 AXI_CCIF，相当于写数据，其接口时序图如图 6 所示，wr_en 为写使能信号，din 为数据输入总线。当 wr_en 有效时，AXI_CCIF 锁存总线上的结果数据并存储到 BRAM 中。last 信号指示结果数据传输完毕，须保持 8 个时钟周期有效，确保被 AXI_CCIF 成功接收。

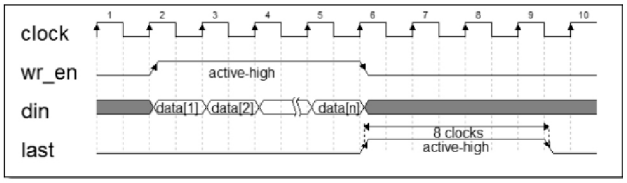


图 6　结果数据通道接口时序图

3.2　AXI_CCIF 的工作流程

ARM 经 AXI 接口模块访问寄存器模块，配置 reg2_ParamLen、reg3_ImgReso、reg4_ImgLen 以及 reg6_ImgReTxTimes 寄存器，配置完成后，ARM 写 reg0_Ctrl 寄存器的启动位，启动 CNN 运算；CNNHA 从图像及权值参数数据通道读取运算数据，进行卷积运算，在卷积运算过程中，CNNHA 将运算结果传回结果数据通道，AXI_CCIF 再经 BRAM 接口模块存储到 BRAM 中。卷积运算完成后，中断模块向 ARM 发起中断，ARM 读取 reg5_ResultLen 寄存器获取运算结果长度，再将运算结果传回。

4　软件设计与实现

该平台使用 Xilinx 公司的 Vivado 2015.4 进行搭建，并采用配套的 SDK 2015.4 进行软件开发。在 Vivado 中搭建好硬件平台，经综合、实现后生成 bit 流文件，把整个硬件设计导入 SDK。在 SDK 中基于生成的板级支持包（BSP，Board Support Package）开发设计平台的软件程序。

基于硬件平台的工作流程开发了与之配套的裸机验证软件程序，软件流程图如图 7 所示。初始化硬件平台，启动 DMA Engine，将 DDR3 Memory 中图像数据搬移到 BRAM 存储块中。当图像数据传输完成后，再次启动 DMA Engine，将 DDR3 Memory 中的权值参数传输到 BRAM 存储块中。调用 AXI_CCIF_WriteReg()函数配置 AXI_CCIF 的 reg2_ParamLen、reg3_ImgReso、reg4_ImgLen 及 reg6_ImgReTxTimes 四个寄存器，调用 AXI_CCIF_ReadReg()函数访问 AXI_CCIF 的状态寄存器（reg1_Status）判断是否配置成功。若配置失败，则返回上一步重新配置 AXI_CCIF，若配置成功，则执行下一步，启动 CNN 运算。调用 AXI_CCIF_WriteReg()函数写 reg0_Ctrl 寄存器的 31-bit 为 1，启动 CNN 运算，进入空闲状态，

等待中断。当接收到中断信号后，调用 AXI_CCIF_ReadReg()函数写 reg0_Ctrl 寄存器的中断复位为 1，再调用 AXI_CCIF_ReadReg()函数读取 reg5_ResultLen 寄存器获取运算结果个数，启动 DMA Engine，把运算结果传回 PS 端。

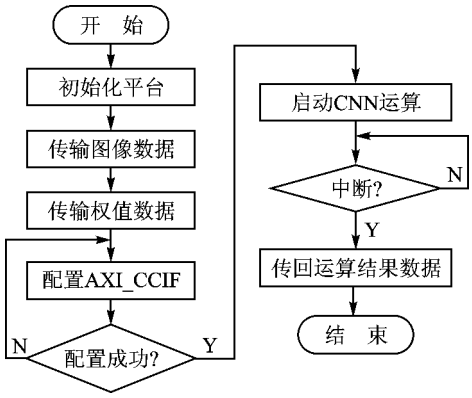


图 7 软件流程图

5 平台验证与分析

CNN 运算的第一层为卷积层，本文使用 Verilog HDL 实现了经典的 LeNet-5 的卷积层 1 作为 CNNHA 模块，验证平台功能完整性，图 8 为 LeNet-5 卷积层 1 的 CNNHA 原理图。输入 32×32 的图像，经过归一化处理后，依次与 6 个 5×5 的卷积核进行卷积运算，最终得到 6 个 28×28 的特征图谱。在 FPGA 中通常使用定点数进行计算，参考文献[7]、[8]研究表明，采用定点数代替浮点数进行 CNN 运算，对计算结果影响很小。因此 CNNHA 的基本运算单元均采用 Q28 表示的 32 位定点数进行运算，降低计算复杂度，提高计算性能。

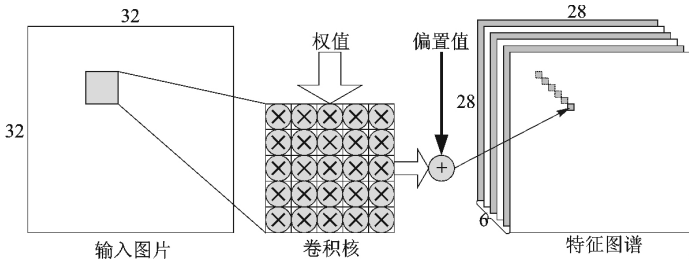


图 8 LeNet-5 卷积层 1 的 CNNHA 原理图

在板级验证过程中，使用 Ila 在线逻辑分析仪捕捉芯片上的信号。图 9 是捕捉到的 LeNet-5 的运算结果和运算结果个数等信号，可以看到其结果与仿真结果相同，没有出现误码。

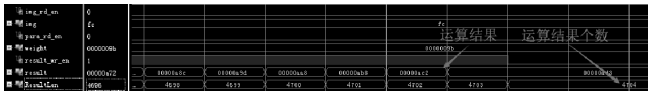


图 9 Ila 捕捉 LeNet-5 的信号

在满足功能需求的同时，还要尽量提高数据的传输速度。该平台采用 DMA 方式，有效提高了 PS 与 PL 之间的数据传输效率。实验用了 10 张图片对 LeNet-5 进行了验证，并计算出了数据的平均传输时间和传输速度。如表 1 所列，LeNet-5 的输入为 8 位灰度图像数据，数据量为 1024(32×32)字节；权值参数为 32 位定点数，数据量为 624(权值，5×5×6×4；偏置，6×4)字节；运算结果为 32 位定点数，数据量为 18 816(28×28×6×4)字节。通过读取 ARM 的 Global Timer 寄存器，可以估算数据传输的时间，进而计算出传输速度，该寄存器的计数频率为 ARM 工作频率的一半。由表 1 可知，数据传输速度大于 100 Mbps。传输时间和传输速度的计算公式如下：

$$T = \frac{GTIMER_e - GTIMER_s}{f_{GT}} \quad (2)$$

$$V = \frac{amount_d}{T} \quad (3)$$

式(2)计算传输时间，其中$GTIMER_e$表示数据传输结束时 Global Timer 寄存器的计数值，$GTIMER_s$表示数据传输开始时的计数值，f_{GT}表示 Global Timer 寄存器的计数频率。式(3)计算传输速度，$amount_d$表示传输的数据量。

表 1 LeNet-5 的数据传输速度

数据名称	数据量/B	传输时间/ms	传输速度/Mbps
图像	1024	0.00801	121.92
权值参数	624	0.00497	119.74
运算结果	18816	0.08324	215.57

结 语

该平台对 LeNet-5 的 CNNHA 模块进行了板级验证，能够快速、稳定地完成运算。针对不同 CNN 算法，仅需要修改 CNN 算法模块中与平台数据通道相连的部分，即可使用该平台进行板级验证，平台数据通道的时序简单，便于实现，可有效提高开发效率。但是，PS 与 PL 之间的数据传输带宽仍比较低，导致整个 CNN 运算过程耗费的时间较长。优化 PS 端与 PL 端的数据传输通道，提升平台的效率，是进一步需要解决的问题。

参考文献

[1] 李彦冬，郝宗波，雷航. 卷积神经网络研究综述[J]. 计算机应用，2016，36(9)：2508-2515.

[2] 余子健，马德，严晓浪，等. 基于 FPGA 的卷积神经网络加速器[J]. 计算机工程，2017，43(1)：109-114.

[3] Xilinx Inc. Zynq-7000 all programmableSoC technical reference manual，2013.

[4] 何宾. Xilinx All Programmable Zynq-7000 SoC 设计指南[M]. 北京：清华大学出版社，2013.

时，发送者一直发送连续帧，直到帧结束。

STmin 和 BS 的设置决定了多帧通信的速度，进而决定了软件升级时间的长短。这两个参数的设置受到接收缓冲区容量和接收端数据帧处理速度的限制。

4 软件升级测试

笔者为某车厂开发了一款带有软件升级功能的 PEPS，MCU 选用恩智浦中端 16 位单片机 MC9S12G128，升级文件采用 S19 格式，文件大小为 124 KB，采用 canoe 设计了上位机升级软件，选择升级文件后进行一键式下载，上位机升级软件界面如图 3 所示。

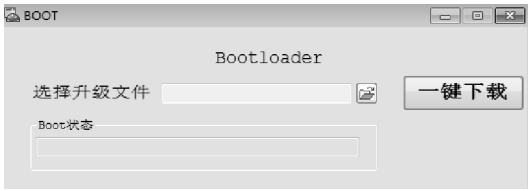


图 3 升级软件界面

软件升级时间取决于程序指令的下载时间，程序指令下载采用多帧通信方式，STmin 和 BS 的设置决定了多帧通信的速度。此外，多帧报文数据设定长度不同，升级时间也不同。STmin、BS、多帧报文长度的设置取决于报文接收缓冲区容量和接收端数据帧处理速度，将 BS 设置为固定值 0，选择不同的 STmin 和多帧报文长度，统计下载 Flash 驱动和下载应用程序的执行时间，实测下载指令数据时间如表 1 所列。

表 1 下载指令数据时间(单位：s)

长度 / STmin	960	768	512	256
1	12.88	16.24	21.32	25.64
2	19.66	22.96	29.08	32.45
4	35.18	40.42	48.26	46.33
8	64.90	60.16	56.74	59.58

可见，多帧报文长度相同时，STmin 越小，下载指令数据时间越短。在 STmin 等于 1 的情况下，多帧报文长度越长，下载指令数据时间越短，在 STmin 较大的情况下，下载指令数据时间和多帧报文长度不存在明确的关系。

为了提高接收端数据帧处理速度，将 CAN 报文接收中断服务程序放入 RAM 中执行，考虑到功能安全、MCU RAM 容量和运行主频，最终将多帧报文长度设定为 512 字节，STmin 设置为 1 ms，在这种设置下，软件升级时间可以控制在 30 s 以内，改善了开发人员和 4S 店服务人员现场升级的用户体验。

结 语

为了避免产品生命周期内因程序跑飞导致 Flash 数据误擦除的问题，在软件升级期间，将 Flash 驱动程序复制到指定 RAM 空间中，升级完成后清空指定 RAM 空间，保证了系统中不存在 Flash 驱动程序的任何拷贝，从而在根本上消除了程序 Flash 数据被错误擦除的隐患。最后，根据对 UDS 数据传输服务的分析，设置合理的多帧报文长度和 STmin 参数，在保证功能安全的前提下，尽可能降低了软件升级时间。

参考文献

[1] 郭帅，李军伟，高松. 多节点软件触发式 Bootloader 设计与实现[J]. 现代电子技术，2017，40(18)：35-39.

[2] 山东省科学院自动化研究所. 一种基于 UDS 的 CAN 节点 bootloader 设计方法及系统：中国，201810796758. 9[P]. 2018-7-19.

[3] Road vehicles-Diagnostics on Controller Area Net works (CAN)-Part 3：Implementation of unified diagnostic services (UDS on CAN) ISO15765-3[S]，2004(E).

[4] 李娟娟，刘孔祥，李济林. 智能前照灯的 CAN 刷新软件的设计[J]. 汽车电器，2012(9)：1-4.

[5] 常欣红，于金泳，刘志远. 汽车故障诊断标准 ISO15765 的网络层分析与实现[J]. 汽车技术，2006(9)：40-44.

马建辉(工程师)，主要研究方向为汽车电子、嵌入式系统应用。

(责任编辑：薛士然 收稿日期：2018-11-15)

[5] LECUN Y，BOTTOU L，BENGIO Y，et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE，1998，86(11)：2278-2324.

[6] 卢宏涛，张秦川. 深度卷积神经网络在计算机视觉中的应用研究综述[J]. 数据采集与处理，2016，31(1)：1-17.

[7] COURBARIAUX M，BENGIO Y，DAVID J P. Training deep neural networks with low precision multiplications[J]. Computer Science，2014.

[8] MOINI S，ALIZADEH B，EMADM，et al. A Resource-Limited Hardware Accelerator for Convolutional Neural Networks in Embedded Vision Applications[J]. IEEE Transactions on Circuits&Systems II Express Briefs，2017，64(10)：1217-1221.

[9] SUN Y，WANG X，TANG X. Deep Learning Face Representation by Joint Identification-Verification[J]. 2014(27)：1988-1996.

冯光顺(硕士研究生)，主要研究方向为嵌入式系统设计、深度学习；应三丛(副教授)，主要研究方向为计算机应用。

(责任编辑：薛士然 收稿日期：2018-12-04)