Software Engineering and Applications 软件工程与应用, 2018, 7(3), 160-167
Published Online June 2018 in Hans. http://www.hanspub.org/journal/sea
https://doi.org/10.12677/sea.2018.73019

Hans汉斯

Asynchronous Convolution Acceleration Algorithm Based on 4 × 4 Convolution Kernels

Haibo Cheng, Lvying Yu, Pengfei Li, Haitao Zhang, Anping He, Yi Yang

School of Information Science and Engineering, Lanzhou University, Lanzhou Gansu
Email: chenghb16@lanu.edu.cn

Received: Jun. 7th, 2018; accepted: Jun. 21st, 2018; published: Jun. 27th, 2018

Abstract

Due to convolutional operations based on software-side convolutional neural networks, it is difficult to meet the computational performance and power consumption requirements of current convolutional neural networks. In order to overcome the difficulties, this paper designs an asynchronous convolution accelerating algorithm based on 4 × 4 convolutional kernel to accelerate the convolutional neural network. Using Add Tree to implement multiplication and addition of kernel matrix and pic matrix, a single Add Tree calculation unit is a 4 × 4 convolution kernel and the same size image matrix data is multiplied and added to obtain an eigenvalue. Parallel computing using multiple Add Tree can greatly increase the convolution calculation rate. The experimental results show that the acceleration algorithm has the advantage of not being limited by the clock frequency and can work at any clock frequency, and the calculation speed of a single computing unit is also very fast. The time to calculate an eigenvalue is about 500 ns. Compared with the calculation speed on the software side, it has increased by about 10 times.

Keywords

Convolution Neural Network, Parallel, 4 × 4 Convolution Kernels

基于4 × 4卷积核的异步卷积加速算法研究

程海波，余旅莹，李鹏飞，张海涛，何安平，杨　裔

兰州大学，信息科学与工程学院，甘肃 兰州
Email: chenghb16@lanu.edu.cn

收稿日期：2018年6月7日；录用日期：2018年6月21日；发布日期：2018年6月27日

文章引用: 程海波, 余旅莹, 李鹏飞, 张海涛, 何安平, 杨裔. 基于4 × 4卷积核的异步卷积加速算法研究[J]. 软件工程与应用, 2018, 7(3): 160-167. DOI: 10.12677/sea.2018.73019

摘 要

由于基于软件端卷积神经网络的卷积运算难以满足现在的卷积神经网络对运算性能与功耗的要求，为了克服困难，本文设计了一种基于4 × 4卷积核的异步卷积加速算法来对卷积神经网络进行加速。采用Add Tree的形式来实现kernel矩阵和pic矩阵的乘加运算，1个Add Tree计算单元是1个4 × 4的卷积核与相同大小的图片矩阵的数据做乘加运算得到一个特征值，采用多个Add Tree的并行计算方式可以大幅度提升卷积计算速率。实验结果表明，该加速算法还有不受时钟频率限制的优点，可以工作在任何时钟频率下，且单个计算单元的计算速度也十分的快，计算一个特征值的时间大约在500 ns左右，相对于软件端的计算速率提升了10倍左右。

关键词

卷积神经网络，并行，4 × 4卷积核

1. 引言

卷积神经网络(Convolutional neural network, CNN)是一种源自人工神经网络的深度机器学习算法[1]，它对各种形式的图片的识别具有高度的适应性，是提取图形特征灵敏的传感器，其权值共享网络结构使之更类似于生物神经网络结构，大大降低了网络模型的复杂度，减少了权值的数量。另外，该优点在网络输入多维图像时变现得更明显，可以使图片的输入直接作为网络的输入，这样就避免了传统识别中复杂的特征提取和数据重建过程[2]。因此，近年来 CNN 神经网络结构在图像识别[3]与检索、人脸识别、性别/年龄/情绪识别、物体检测、场景判定与危险监控等领域已经得到越来越广泛的应用了[4]。

CNN 的常用框架有 Caffe, Torch, Tensor Flow 几个框架。Caffe 框架源于 Berkeley 的主流 CV 工具包，支持 C++, python, matlab, Model Zoo 中有大量预训练好的模型供使用[5]。Prototxt 是用命令行做训练时要用的，定义了 CNN 的层级结构；而 Torch 框架则是 Facebook 用的卷积神经网络工具包，通过时域卷积的本地接口，使用非常直观，torch 定义新网络层比较简单，可以做图像或者 RNN；最后 Tensor Flow 框架 Google 的深度学习框架，可视化很方便，数据和模型并行化好，速度非常快，也是目前用的很多的学习框架。

由于 CNN 的特殊计算模式，通用处理器实现 CNN 并不高效，所以很难满足性能的需求。最近基于 FPGA，GPU 甚至 ASIC 的不同的加速器被相继提出以提升 CNN 设计性能[6]。在这些方案中，基于 FPGA 的加速由于其更好的性能，高效，快速开发周期以及可配置的能力吸引了很多研究者们的注意[7]。其中在控制逻辑较为复杂的情况的时候，选用异步电路作为设计的优势在于不用考虑时钟的工作频率，在任何工作频率下，异步电路都能有很快的计算速度，各个模块之间工作互不干扰。在实验中，研究人员发现在 FPGA 相同的逻辑资源利用率情况下，两种不同的解决方案可能会有高达 90%的性能差异。所以找出最佳解决方案是十分重要的，特别是当考虑到 FPGA 平台的计算资源和存储器带宽的限制时。实际上，如果加速器结构没有精心设计，其计算吞吐量与提供 FPGA 平台的内存带宽不匹配[8]。这意味着由于逻辑资源或存储器带宽的利用不足将造成性能的降级。

目前基于FPGA的CNN实现有很多方案，文献[9]中提到了FPGA的可配置性，大幅度的提高了FPGA的资源利用率，将复杂的CNN卷积计算转化成简单的密集矩阵乘法运算；文献[10]中提到寒武纪芯片在计算CNN神经网络算法的时候，可以用CPU(Xeon E5-4620)和GPU(K20M)十分之一的面积，分别达到CPU的117倍，GPU的1.1倍的性能。可以得出一个结论：用FGPA来实现CNN卷积神经网络算法不是不可实现的，而且是更有效的更快速的方法。

相比较之下，基于FPGA的异步CNN的实现就更少了，本文中我们提出一种基于4 × 4卷积核的异步卷积加速算法，首先将需要的数据通过bram保存在内存当中，当开始计算的握手信号到来时，又通过bram将数据写给计算单元PE，PE开始进行乘加运算，最后将得到的特征值返回给bram中保存起来；通过多个4 × 4卷积核与图片矩阵相乘之间的并行计算，从而达到对卷积计算的加速过程。

2. 卷积神经网络

一个卷积神经网络主要由以下四个部分组成：卷积层(Convolution Layer)，池化层(Pooling Layer)，全连接层(Fully Connected Layer)，损失函数(Loss Function)。通过各个层之间的级联，实现对图片的特征提取和组合[11]。

1) 卷积层：卷积层的主要工作是特征提取工作。在卷积神经网络中低层次的卷积层,主要完成对整个图片的特征提取，而高层次的卷积层则是对提取到的一些特征图更进一步的特征提取，得到更利于分类工作的特征图。每一个卷基层中都含有若干个卷积核，本文中选用的固定的4 × 4的卷积核，通过不同的卷积核完成对不同特征的提取，其公式表达式(1)所示：

$$X_f^l = f\left(\frac{1}{n}\sum_{i\in M_j} X_i^{l-1} + B^l\right) \tag{1}$$

2) 池化层：池化层实现对特征图的再采样，通过将图像和分成若干个区域，在每个区域内采样，不仅减少了数据量，同时也使得到的特征更加明显。常见的池化方法有均值池化(Mean-Pooling)，最大池化(Max-Pooling)。其池化公式如式(2)所示：

$$O_{i,j,k} = \max_{0\le x\le m, 0\le y\le m}\left(I_{i\cdot m+x, j\cdot m+y, k}\right) \tag{2}$$

3) 全连接层：全连接层是一种特殊的卷积层，通常位于网络的最顶层。全连接层和卷积层一样，含有多个卷积层，跟卷积层不同的是它的卷积核的大小和输入特征图的大小一样，所以全连接层是对最后的抽象特征进行组合拼接的过程。

4) 损失函数：损失函数是网络在训练过程中必不可少的部分。在训练的过程中，通过比较网络的输出与选定的目标之间的特征距离，再利用反向传导算法，不断调整整个网络中的参数，从而不断优化网络结构，使网络慢慢的变的更加稳定，能够更好的完成任务目标。常用的损失函数有SoftMax损失函数以及欧氏距离损失函数等等，针对不同目标任务会用到不同的损失函数[12]。

3. 异步4 × 4卷积计算框架

本文提出一种基于4 × 4卷积核的异步卷积加速算法，将卷积神经网络中的卷积层剥离出来，拿到FPGA上实现，考虑到在应用中基本都是将训练好的CNN模型部署到现有计算平台上进行预测操作，所以，很多的FPGA加速方案中仅考虑优化前向操作。同时又有研究表明，卷积操作占据了CNN模型将近90%的计算时间，FPGA上的计算单元可以达到纳秒级运算，这是软件端无法做到的，所以本文也是仅考虑CNN神经网络框架中前向传播[13]计算中的卷积运算[14]。

3.1. 卷积计算算法

基于 bram 的存储设计

本次设计是以 4 × 4 的卷积核为例进行设计的，在顶层模块同时调用 4 个 click_control 模块，如图 1 所示，将 data1-data16 这 16 个数据分成 4 组，每组有 4 个数据，每组中的数据以串行的方式传入到 BRAM 中(即每次传送一个 data (32 位))，然后将 16 个 data 数据并行输出。

令A和B是 32 位浮点数，read address 为输出数据的地址，kernel data 权值矩阵的数据，pic_data 为图片矩阵的数据，m，n 是全权值矩阵的行数和列数，row 表示当前数据的行数，cow 表示当前数据的列数。若当前数据所在的行数 row 和列数 cow 都不等于 m，n 的时候，数据所在列数将会执行一个加 1 的操作，同时输出 kernel 矩阵和 pic 矩阵的数据，read address 是矩阵每个数据对应的地址；但是,如果当前数据的列数 cow 等于 n，数据所在的行数将会执行一个加 1 操作。本文提出的基于 4 × 4 卷积核的异步卷积加速算法中，能够将数据准确的输入存储模块 bram 中，与传统的数据传输方法比较，虽然牺牲大量的 bram 资源来进行存储，但增加了资源的利用率，大幅度的提高了卷积计算的速度。算法 1 是本文提出的基于 bram 的读写算法的伪码。

算法 1：基于 bram 的读写算法

```
procedure bram(A，B，read address，kernel data，pic data)
  Input：A，B；
  Output：read address，kernel data，pic data；
parameter：m← 0~1024；n← 0~1024；
01：for(n= 0；n< 1024；n++）
02：for(m= 0；m< 3；m++）begin
03：if(row!= m）
04：    if(cow != n)begin
05：cow ++；
06：Output：  read address((row-1）*n + cow)；
07：                    Output：  kernel data (A)；
08：                    Output：  pic data (B)；
09：  elserow ++；
10：return
11：end procedure
```

3.2. 卷积计算算法

3.2.1. 卷积计算算法

为了提高数据计算的速度，卷积计算算法 PE 单元是以 Add Tree 的形式设计的，流程图如下图 2 所示，Add Tree 总共分为 6 层：1) 由 16 个浮点数乘法器构成；2) 由 8 个浮点数加法器构成；3) 由 4 个浮点数加法器构成；4) 由 2 个浮点数加法器构成；5) 由 1 个浮点数加法器构成；6) 由 1 个选择器构成；

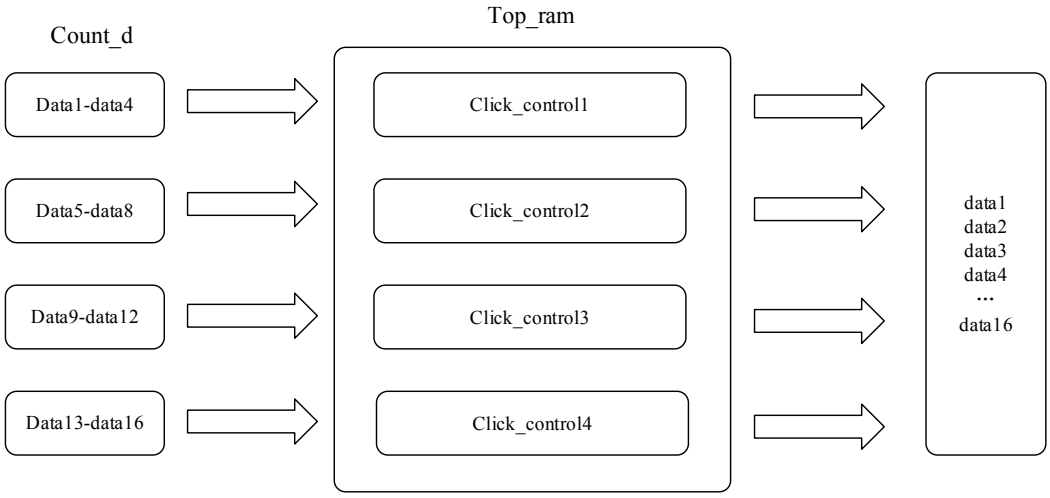

Figure 1. The overall block diagram
图 1. 整体框图

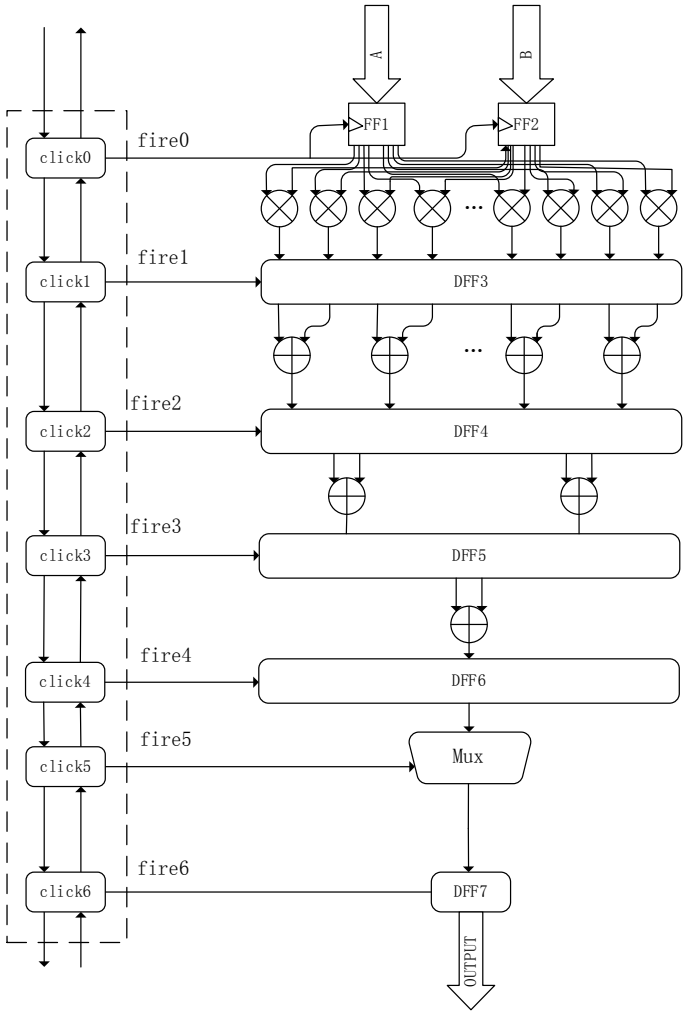

Figure 2. Algorithm for convolution computation (PE)
图 2. 卷积计算算法(PE)

当数据从 3.1 中 bram 模块中输入到卷积计算模块 PE 中，通过异步握手信号通讯控制分析：

1) 首先存储到寄存器 DFF1 和 DFF2 中，当异步握手信号 click0 到来的时候，会产生一个触发信号 fire0,而 fire0 在这里作为 DFF1 和 DFF2 的使能信号控制 kernel 矩阵和 pic 矩阵的数据进入到下一级浮点数乘法运算单元中，此时浮点数乘法器计算得到的结果是 kernel 矩阵和 pic 矩阵中的点乘运算，所有的数据会保存在寄存器 DFF3 中。

2) 当握手信号 click1 到来时，fire1 控制寄存器 DFF3 把乘法器得到的结果送到浮点数加法运算单元中，此时只是进行一次加法运算，将得到的结果存储在寄存器 DFF4 中。

3) 当握手信号 click2 到来时，fire2 控制寄存器 DFF4 将上一级加法器得到的结果送到第二级的加法运算单元中进行加法运算。

4) 以此类推，当握手信号 click3 到来时，fire3 控制寄存器 DFF5 将上一级加法器得到的结果送到最后一级的加法运算单元中进行加法运算。此时得到一个唯一值，是一个 4 × 4 的 kernel 矩阵和一个同样大小的 pic 矩阵相乘得到结果。

5) 当握手信号 click4 到来时，fire4 控制寄存器 DFF6 将得到的最终结果送到选择器 Mux 中进行帅选。

6) 当握手信号 click5 到来时，fire5 控制选择器 Mux 选择合适的结正确的值输出。

7) 当握手信号 click6 到来时，fire6 控制寄存器 DFF7 输出正确的值。

3.2.2. 顶层卷积算法设计

图 3 中 kernel 矩阵数据预先存放在 bram 中(kernel 矩阵只存储一次，可以多次调用)，而 pic 矩阵的数据由于数据量过大只能分多次传输。所以，顶层的卷积算法可以分为以下几个部分：

kernel 矩阵和 pic 矩阵数据的读取算法；卷积计算单元 PE 的算法；计算结果返回给 bram 中的写入算法；

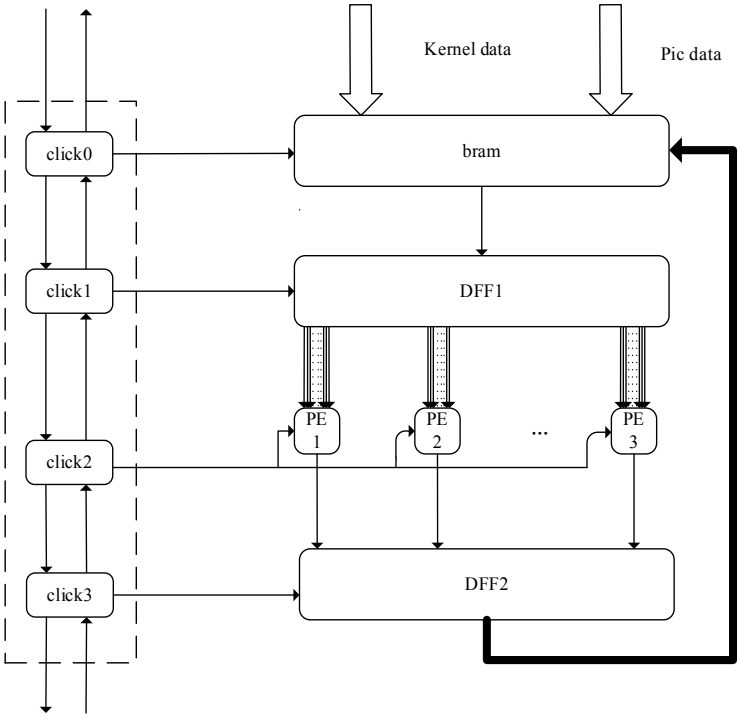

Figure 3. The top convolution algorithm design
图 3. 顶层卷积算法设计

4. FPGA 验证与仿真

4.1. 实验环境

本文实验使用 Xilinx 公司的 Vivado 进行硬件开发和仿真。FPGA 使用 zynq7000 系列 AX7020 开发板。

加速算法的整体架构如图 4 所示。系统中的 CPU 负责通信与运算过程的控制以及数据的传输。USB 借口是 FPGA 与 PC 端 host 的通信接口。计算单元是 CNN 运算单元，完成卷积神经网络前向传播过程的计算。Block RAM 用于存放计算所用的神经元以及权值，以及运算的中间结果。

运算的整体控制还是由 PS 端的 CPU 来控制完成，主机会发送输入图像数据和运算命令。PL 端在接受到命令后，将数据搬运到 RAM 中，然后控制 CNN 进行计算。计算完成后,输出结果返回给 PS 端。

在考虑到开发板的资源的有限的情况下，尽可能的将资源的利用率达到最大化，单个 4 × 4 卷积核所需要的资源如表 1 所示，可以得出在 AX7020 开发板上可以同时用到 4 个 4 × 4 的卷积核，形成 4 个数据流往下流，从而达到高速的并行卷积计算。

4.2. 实验结果分析

表 2 中列出了各个模块在 FPGA 上计算一次得到结果的时间，可以得到 4 个顶层模块同时开始计算不同的权值矩阵消耗的时间一样，这样会大幅的提升卷积计算的速度。

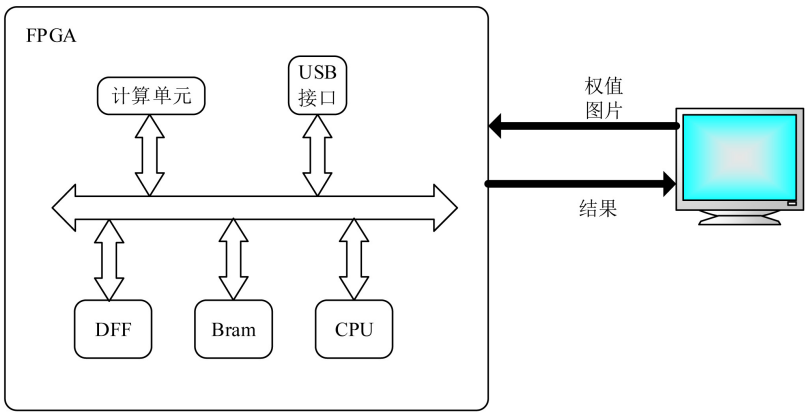

Figure 4. The experimental system diagram
图 4. 实验系统图

Table 1. Resource utilization of a single 4 × 4 convolution kernel on the FPGA
表 1. 单个 4 × 4 卷积核在 FPGA 上的资源使用情况

模块	占用资源数	占用率/%
Registers	4520	5
LUTs	9456	18
DSPs	36	16
Bram	8	6

Table 2. Calculated time
表 2. 计算时间

模块	计算一次需要的时间/ns
存储模块	263
计算模块	415

5. 结束语

本文对卷积神经网络中的卷积运算速度不够快的问题提出了一种基于 4 × 4 卷积核的异步卷积加速算法，实现过程是 CPU 端提供第一个握手请求信号，让该卷积计算模块启动，此时导入外部的图片数据和权值矩阵的数据并加以存储，之后，存储部分将存数单元里的数据递给 PE 计算单元，当计算模块计算完毕会递给存储模块一个反馈信号，此时存储模块将会递给计算模块第二组数据，以此类推，直至所有数据全部计算完毕，此时存储模块将导入第二张图片数据。计算一个 4 × 4 卷积核加上 BRAM 通信的时间总共为 555 ns，其中数据的写入和读取花费的总时间是 235 ns，数据的计算花费的时间是 320 ns。实验结果表明，该加速算法可以有效的提高运算速度。

基金项目

受国家自然科学基金(61602224, 61402121)；中央高校基础研究基金(lzujbky-2017-194, lzujbky-2018-130, Grant No.lzujbky-2016-br03)；广西科技计划项目(桂科 AB17129012)佛山市科技创新项目(Grant No. 2015IT100095)；中国教育科研网创新项目(Grant No. NGIL20150606)；广东省科技创新项目(Grant No. 2016B010108002)；广西混杂计算与集成电路设计分析重点实验室开放基金课题资助(HCIC201714)。

参考文献

[1] 杨薇. 卷积神经网络的 FPGA 并行结构研究[J]. 数字技术与应用, 2015(12): 51.

[2] 王羽. 基于 FPGA 的卷积神经网络应用研究[D]: [硕士学位论文]. 广州: 华南理工大学, 2016.

[3] 蒋帅. 基于卷积神经网络的图像识别[D]: [硕士学位论文]. 长春: 吉林大学, 2017.

[4] 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.

[5] 刘进锋. 一种简洁高效的加速卷积神经网络的方法[J]. 科学技术与工程, 2014, 14(33): 240-244.

[6] 余奇. 基于 FPGA 的深度学习加速器设计与实现[D]: [硕士学位论文]. 合肥: 中国科学技术大学, 2016.

[7] 方睿, 刘加贺, 薛志辉, 等. 卷积神经网络的 FPGA 并行加速方案设计[J]. 计算机工程与应用, 2015, 51(8): 32-36.

[8] 陆志坚. 基于 FPGA 的卷积神经网络并行结构研究[D]: [博士学位论文]. 哈尔滨: 哈尔滨工程大学, 2013.

[9] 魏少军, 刘雷波, 尹首一. 可重构计算处理器技术[J]. 中国科学: 信息科学, 2012(12): 1559-1576.

[10] 廖红艳. 寒武纪: 打造人工智能芯片[J]. 环球科学, 2016(9): 20.

[11] Yu, Z.J., Ma, D., Yan, X.L., *et al.* (2017) FPGA-Based Accelerator for Convolutional Network. *Computer Engineering*, **43**, 109-114, 119.

[12] 叶浪. 基于卷积神经网络的人脸识别研究[D]: [硕士学位论文]. 南京: 东南大学, 2015.

[13] 钟楠, 刘明, 李圣辰. 基于 FPGA 的卷积神经网络加速方案设计[J]. 2017.

[14] 凡保磊. 卷积神经网络的并行化研究[D]: [硕士学位论文]. 郑州: 郑州大学, 2013.

知网检索的两种方式：

1. 打开知网页面 http://kns.cnki.net/kns/brief/result.aspx?dbPrefix=WWJD
   下拉列表框选择：[ISSN]，输入期刊 ISSN：2325-2286，即可查询
2. 打开知网首页 http://cnki.net/
   左侧“国际文献总库”进入，输入文章标题，即可查询

投稿请点击：http://www.hanspub.org/Submission.aspx
期刊邮箱：sea@hanspub.org