计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(11):101−106 [doi: 10.15888/j.cnki.csa.007151]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 ZYNQ 和 CNN 模型的服装识别系统①

熊　伟, 黄　鲁

(中国科学技术大学 信息科学技术学院, 合肥 230027)
通讯作者: 黄　鲁, E-mail: luhuang@ustc.edu.cn

摘　要: 商品检索是电商行业智能化发展的一个重要的问题. 本设计实现了基于 ZYNQ 和 CNN 模型的服装识别系统. 利用 TensorFlow 训练自定义网络, 定点化处理权重参数. 利用 ZYNQ 器件的 ARM+FPGA 软硬件协同的特点搭建系统, 使用 ARM 端 OpenCV 进行图像预处理, FPGA 端 CNN IP 进行实时识别. ARM 与 FPGA 之间实现了权重可重加载结构, 无需修改 FPGA 硬件而实现在线升级. 系统采用 fashion-minist 数据集作为网络训练样本, 根据系统资源配置 CNN IP 的加速引擎的数量来提高卷积运算的并行性. 实验表明, 本系统针对电商平台下的图片能够实时准确识别和显示, 准确率达 92.39%. 在 100 MHz 工作频率下, 图像处理速度每帧可达到 1.361 ms, 功耗仅为 0.53 W.

关键词: ZYNQ; CNN; 服装识别; 软硬件协同

引用格式: 熊伟,黄鲁.基于 ZYNQ 和 CNN 模型的服装识别系统.计算机系统应用,2019,28(11):101–106. http://www.c-s-a.org.cn/1003-3254/7151.html

Fashion Recognition System Based on ZYNQ and CNN Model

XIONG Wei, HUANG Lu

(School of Information Science and Technology, University of Science and Technology of China, Hefei 230027, China)

Abstract: Commodity retrieval is fundamental to the intelligent development of the e-commerce industry. This study is concerned with a fashion recognition system which bases on ZYNQ and CNN model, trains the custom network with TensorFlow and processes the weighs by using fixed-point calculations. This system applies ARM + FPGA software and hardware coordination method of the ZYNQ device to construct its framework, ARM to preprocess image by OpenCV and the CNN IP of FPGA to synchronously recognize the image. A weight-reloadable structure is implemented between ARM and FPGA, so online upgrade could be realized without modifying the FPGA hardware. The system uses fashion-minist datasets as the sample in the network training and improves the parallelism of convolution operations by increasing the acceleration engines of CNN IP. According to the experiment, the system realizes accurate and real-time identification and display for actual pictures of the e-commerce platform and its accuracy reaches 92.39%. The image processing speed can reach 1.261 ms per frame and the power consumption is only 0.53 W at 100 MHz.

Key words: ZYNQ; CNN; fashion recognition; hardware-software-collaborative

基于机器视觉的服装识别主要分为两大类, 其一是采用人工提取特征进行识别, 列如基于轮廓傅里叶描述子特征的 svm 分类法[1], 其二是基于深度学习的方法, 列如基于残差的优化卷积神经网络服装分类算法[2]. 虽然深度学习的方法可以大大减少开发难度和周期, 但是由于网络模型太大, FPGA 因有限的片上资源无法放下大型网络. 针对这个问题, 可以通过剪枝[3], 低精度权值[4]等方法压缩模型, 从而适应 FPGA 有限的片上资源.

① 基金项目: 国家自然科学基金 (61874102)
Foundation item: National Natural Science Foundation of China (61874102)
收稿时间: 2019-04-19; 修改时间: 2019-05-16; 采用时间: 2019-05-20; csa 在线出版时间: 2019-11-06

本文采用 CNN 模型对视频流中的服装进行识别. 该系统在 ZYNQ 开发板上进行开发与验证, 利用 ZYNQ 器件的软硬件协同的处理能力, 对计算任务进行软硬件划分, 在 ARM 端完成图像采集、预处理和显示, FPGA 端利用 Verilog 语言设计卷积神经网络的硬件 IP. 采用 AXI 总线实现片内通信, 提出权重数据的重加载结构. 可以根据不同应用场景选择合适网络模型重新训练, 只需要在 ARM 端传送已经训练好的权重数据而无需修改硬件.

1 卷积神经网络软件架构设计

1.1 CNN 模型搭建

本设计采用一种类似 VGG[5]的一种自定义神经网络为基础, 命名为 Mini-VGGNet. 如图 1 所示, 基本结构与 VGG 网络十分类似, 该网络结构非常简洁, 整个网络都使用了同样大小的卷积核尺寸 (3×3) 和最大池化尺寸 (2×2). 通过不断加深网络结构来提升性能. 每一层卷积或全连接层后面都使用易于在硬件上实现的 Relu 激活函数.

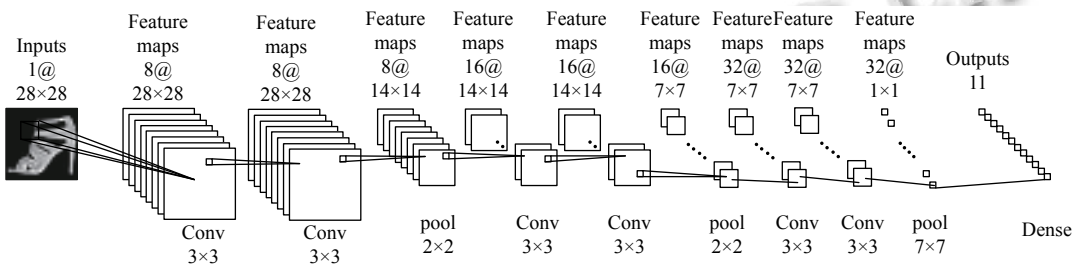

图 1 网络模型结构

由于传统卷积神经网络的 90% 的参数都集中在全连接层, 这里在第 6 层卷积后面加入 global_max_pooling[6]而不是普通的池化, 使得 7×7 大小的特征图直接变为 1×1, 大大减少了全连接层的参数. 使得可以利用 FPGA 的片上内存即可储存所有的权重参数. 最后在全连接层后面加上 Softmax 函数, 实现 11 个分类, 第 11 分类为未知类.

模型的时间复杂度即运算次数, 可用 FLOPS (FLOating-Point operationS) 衡量, 也就是浮点运算次数. 公式如下:

$$Time \sim (M^{2*}K^{2*}C_{in}{}^{*}C_{out}) \tag{1}$$

其中, M 是每个卷积核输出特征图的尺寸. K 是每个卷积核的尺寸. C_{in} 是每个卷积核的输入通道数. C_{out} 是本卷积层具有的输出通道数. 由表 1, 可以看出 90% 以上的运算都集中在卷积层.

综上该网络架构的特点是:

(1) 只使用 3×3 的卷积层堆叠在一起来增加深度.

(2) 使用最大池化来减小数组大小.

(3) 网络末端全连接层在 Softmax 分类器之前.

(4) 卷积核都是 8 的倍数, 方便卷积核的并行化.

1.2 CNN 模型训练结果和参数定点化处理

根据本设计提出的网络结构, 通过使用 TensorFlow 深度学习库来进行整体网络结构搭建, 最终可以对 fashion-minist[7]数据集实现 92.39% 的准确率. 对比多份对该数据集提交的测试模型[8]. 由表 2 可以看出, Mini-VGGNet 仅使用了大约 20 K 的参数, 没有复杂的预处理操作就能达到相对较高的识别率.

表 1 各层参数量

网络层	输出尺寸	参数数量 权重/bias	各层计算量 (FLOPS)
input	28×8×1	—	—
conv1	28×28×8	72/8	56 448
conv2	28×28×8	576/16	451 584
pool1	14×14×8	—	—
conv3	14×14×16	1152/16	225 792
conv4	14×14×16	2304/16	451 584
pool2	7×7×16	—	—
conv5	7×7×32	4608/32	225 792
conv6	7×7×32	9216/32	451 584
global_max_pooling	1×1×32	—	—
dense	11	352/11	352
Softmax	11	—	—
总计	—	18 403	1 863 136

表 2 不同测试模型

网络模型	预处理	参数数量	测试准确性 (%)
Mini-VGGNet	无	~18 K	92.39
2 Conv+Dropout	正则化	~113 K	92.2
2Conv+3FC	正则化	~1.8 M	93.2
WRN40	标准预处理	~48.9 M	96.7
VGG16	无	~26 M	93.5

由于 CPU 平台训练使用的是浮点数, 所以需要对网络权重定点化处理. 通过在软件端对比不同权重位宽对应的准确率, 在 FPGA 上使用 11 bit 来进行定点化处理而不损失精度. 在 PC 端将权重定点化, 按照网络模型的顺序将权重参数保存为头文件, 通过 ARM 将权重加载到 PL 端的 CNN IP 中去.

2 卷积神经网络硬件设计及实现

2.1 软硬件协同开发系统

本设计提出的服装系统发挥了 ARM+FPGA[9–12]的软硬件协处理能力, 将运算量巨大的 CNN 放在 FPGA 端, 充分发挥 FPGA 并行运算能力. 如图 2 所示, 该系统在 ARM 端实现摄像头的视频采集和前景分割, FPGA 端实现 CNN 算法的硬件加速以及 HDMI 的高清显示.

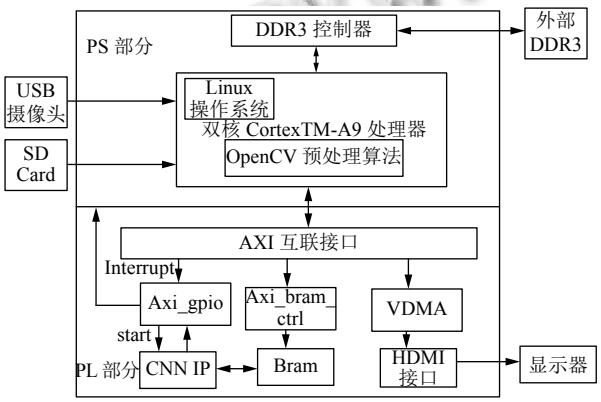

图 2 系统框架图

PS 部分主要包括基于 Petalinux 工具的嵌入式 Linux 系统的构建以及交叉编译器的搭建, OpenCV 视觉库和 QT 库的移植. PL 部分主要包括 CNN IP 的编写和封装, HDMI 视频显示驱动.

2.2 Linux 下 BRAM 驱动设计

在 FPGA 上使用 CNN 识别算法之前, 需要将 ARM 端预处理的图片加载进来, 众所周知, AXI DMA 是 PS 与 PL 之间高速传输的方法, 由于数据量并不是很多, 而且 PL 与 PS 之间传输地址不连续且长度不规则的数据, 此时 AXI DMA 便不再适用了. 权衡之下, 本系统基于 BRAM 的方式, 来进行 PS 和 PL 之间的数据交互.

当需要根据不同实际情况需要重新训练网络, 必须经常改变权值, 因此提出权重加载结构. 在将它们发送到 FPGA 层之前, ZYNQ 系统的 DDR 内存将会保存权重. PS 通过 AXI 总线把权重发给 PL 端的 BRAM, CNN IP 从 BRAM 里面加载需要的权重数据. 由于需要在 Linux 下完成, 因此需要移植 AXI_BRAM 驱动, 根据 Vivado 工程的地址信息, 更改设备树以将 BRAM 内存范围添加到内存节点并添加保留的内存节点, 以便内核不使用内存, 但会将内存映射到内核内存中. 设备树添加如下:

```
memory {
device_type="memory";
reg=<0x0 0x40008000>;
};
reserved-memory {
ranges;
reserved {
reg=<0x40000000 0x8000>;
};
```

这样就可以在用户空间可以使用/dev/mem 去访问物理地址 (0x40000000) 大小为 8 K 的 BRAM, 不修改内核. 建立内存映射函数如下:

```
BRAM64_vptr=(u64*)mmap(NULL, BRAM_size, PROT_READ|PROT_WRITE, MAP_SHARED, fd, BRAM_pbase);
```

2.3 PS 端图像分割研究操作

由于摄像头采集的图像容易受光线干扰, 而训练集的图像背景全是 0, 这样带来的后果就是网络仅对数据集比较友好, 而面对实际情况就会出现相对较大的误差. 这里采取两种办法来解决这个问题,

一种是将 USB 采集的真实图像也加入到 Mini-VGG Net 网络进行训练, 这里一共拍摄了 50×10=500 张, 640×480 的真实图片, 利用 Python 进行预处理, 这里仅仅就是利用双线性插值法 resize 到 28×28 大小.

另一种因为采集的图片是 640×480, 然后在 ARM 上对图片进行缩放加背景分离 (为了还原和训练集的图片差不多), grabCut 算法在 ARM 端比较耗时, 为了满足实时性要求, 即处理时间不能超过 33 ms, 因此系统采用先将图像缩放到 56×56 大小, 然后进行背景分离算法, 再缩放到 28×28, 最后进行形态学操作, 滤波输出 28×28 的图片.

分割流程如图 3 所示, 即 640×480–>56×56–>grabCut –>28×28. 最后再将 CNN_28×28 的图片送给 CNN IP 来计算.

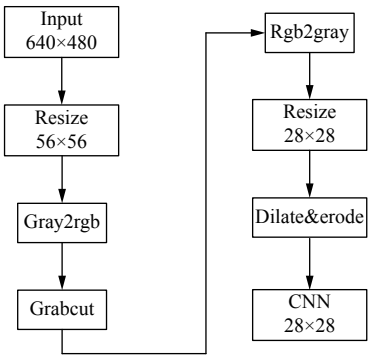

图 3 背景分割流程图

2.4 PL-PS 数据加载结构设计

由于本系统的权重和图像数据都通过 ARM 传给 FPGA, 图 4 为 ARM 和 FPGA 之间的数据加载结构图. ARM 在 DDR 中按顺序保存指定层的权重. ARM 向 FPGA 发送指定层的索引号、权重数量和需要在 DDR 中读取权重的地址. PS 通过 M_AXI_GP 口向 BRAM 地址依次存入 32 bit (仅使用低 11 bit) 的权重数据. PS 每向 BRAM 写完数据后通过 AXI GPIO 给出一个上升沿信号. PL 捕获上升沿后立即将 PS 写入的 32 位数据读出, 并写入 CNN IP 的 database 模块. 同理, 图像数据加载也是如此工作.

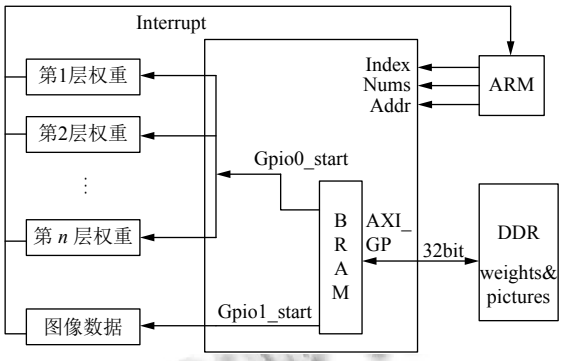

图 4 PL-PS 数据加载结构图

通过 Vivado 的 ila(逻辑分析仪) 抓取的波形如图 5 所示. 可以看出, 该模块能够正常的把 784 个图像数据写入 database 模块, 并且拉低 CNN IP 的启动信号 (低电平有效).

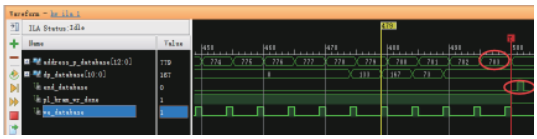

图 5 图像数据加载

2.5 CNN IP 设计

在系统框架中核心就是 CNN IP 的设计. 如图 6 所示, 这个 CNN 加速 IP 在开始信号 (start) 的上升沿到来时开始工作. ARM 通过 AXI 总线, 将数据写入到 CNN IP 的 database 中, databae 存放 CNN 需要的图像数据还有权重参数. 通信是加速器的主要约束条件, 在 CNN 的整个前向计算过程中, 数据吞吐量不断上升, 对于大的特征图基本上都是传到片外内存中去, 而 DDR 数据交互时间对帧率的影响特别大. 而本设计由于大大减少了网络结构和参数, 优化 feature map 数据的存储方式, 使得所有数据可以存放在 BRAM 中, 避免了和 DDR 之间数据交互, 只需要在片上操作. 这里需要两块片上 RAM (0~28×28×8 和 28×28×8+1~28×28×16) 来储存卷积计算的中间特征图. 卷积模块从 mem0 读取数据而计算结果写到 mem1, 接着下次就从 mem1 读取数据而计算结果写到 mem0, 利用乒乓操作实现流水线, 提高了数据吞吐率. 流水化的设计可以提高加速器中的计算单元利用率.

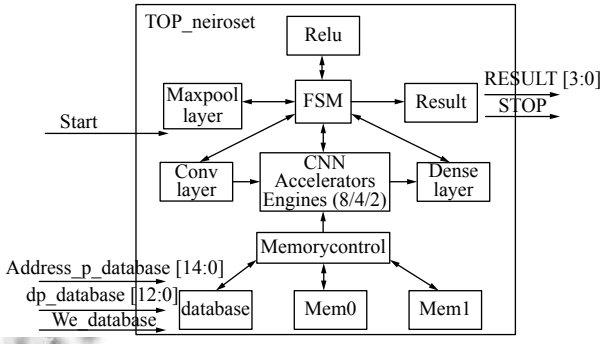

图 6 CNN IP 框架图

卷积层计算时整个网络的核心计算单元, 图 7 是单个处理单元的结构图, 该结构是由 9 个乘法器和 8 个加法器构成, 本文利用 im2col 操作和移位寄存器将图像和权重数据拼接成两个 99 bit 的数据矩阵, 单时钟周期可以对 3×3 的特征图计算输出一个特征图. 而权重数据复用可以降低带宽提高计算效率. 特征数据由一个卷积窗口不断在特征图上滑动生成.

由于 CNN 网络模型的卷积核都是 8 的倍数, 如图 8 所示, 这里最高可以配置为 8 个卷积单元同时工作, 以更多的资源来换取运算时间. 所以本设计能够实现单时钟周期 72 次 MAC(乘累加), 工作在 100 M 的频率下, 峰值运算速率最高可达 7.2 GMAC/s. 与普通

的嵌入式设备相比, 计算效率具有显著的提高, 极大地提高了数据带宽和传输速率.

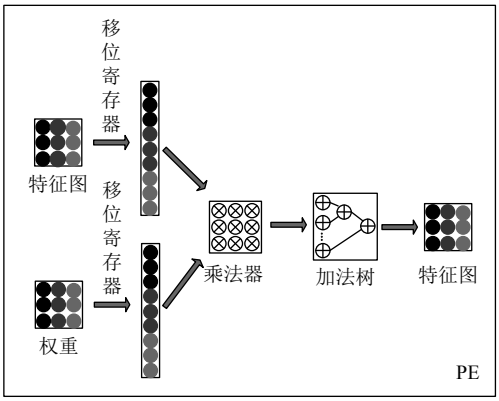

图 7 卷积处理单元 (PE)

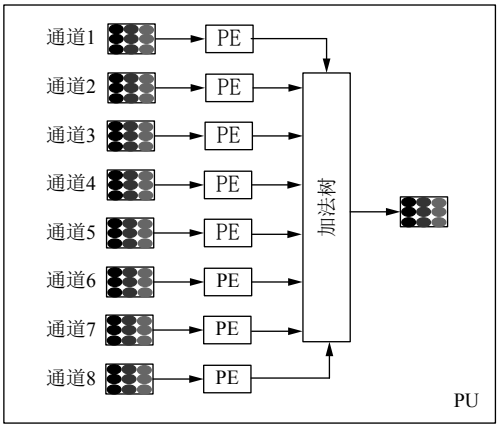

图 8 卷积块处理单元 (并行度为 8)

卷积处理单元 (PE) 同样可以用来全连接层的计算, 在全连接层, 每一个神经元都与上一层相连接, 网络的最后一层是 32 个神经元, 每 8 个为一组, 所以卷积处理单元在这里仅仅使用了 8 个乘法器. 迭代 4 次, 得到 11 个输出.

本文采用 Softmax 函数作为分类器输出, Softmax 的公式如式 (2) 所示.

$$P(j)=\frac{e^{k_i}}{\sum_{i=0}^{n} e^{k_i}} \tag{2}$$

在布置前项传播时, 我们不需要进行指数操作, 指数在硬件上实施代价太大, 而且没有必要, 我们只需要找到 11 个值中最大的那一个就可以, 所以只需要一个比较器即可.

整个流程通过状态机来调度 CNN 各个模块的运算.

3 实验分析

本文实验使用 Xilinx 公司的 Vivado 进行硬件开发和 modelsim 来进行软件仿真. FPGA 使用 ZYNQ 7000 系列 AX7020 开发板. 芯片为 xc7z020clg400-2, 该芯片拥有 53 200 个 LUT, 106 400 个 FF, 140 个 BRAM 36 KB (4.9 M), 220 个 DSP 资源, 完全满足系统实验要求, 将开发板和 USB 摄像头还有 HDMI 显示器连接, 实验平台如图 9 所示.

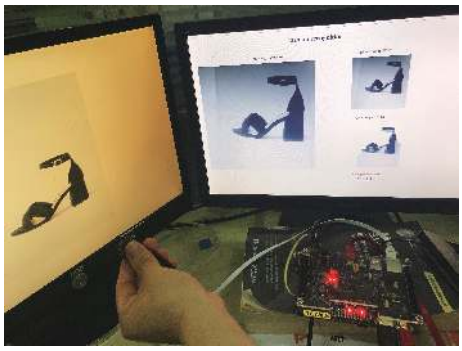

图 9 硬件开发平台图

3.1 数据集测试

将 fashion-minist 数据集存放到 DDR (8 GB) 中去, 该数据集一共有 60 000 张训练图片和 10 000 张测试图片, 大小为 28×28×1, 在实验前在 PC 上利用 Python 将 10 000 张测试集数据转成二进制. ARM 通过 BRAM 把数据加载到 CNN 中, 计算完成触发中断, 在中断里面对错误率进行统计, 对比 PC 机上利用 Python 处理结果, 如表 3, 可以发现在 FPGA 上, 准确率几乎没有下降.

表 3 测试集结果

Lable	FPGA 实现		PC 实现	
	Acc	*Error*	*Acc*	*Error*
T 恤	92.25%	126	92.39%	123
裤子		13		13
套衫		135		135
连衣裙		80		77
外套		81		79
凉鞋		16		13
衬衫		218		216
运动鞋		59		56
包		11		11
裸靴		36		35

3.2 实际图片测试

对电商网站上真实的图片进行测试, 部分测试结果如图 10 所示, 成功完成图像采集、预处理和识别结果的显示.

(a) 裸靴　(b) 连衣裙　(c) 凉鞋　(d) 未知类

图 10　服装识别实际运行结果

每种服装选取 50 个样本, 通过 USB 摄像头对不同服装进行实时采集, 利用 OpenCV 进行预处理缩放到 28×28 大小, 再送到 CNN IP 计算, 并通过 QT 显示. 识别结果如表 4, 除了套衫是 88%, 其余种类识别率基本都达到 90% 以上.

表 4　真实图片测试准确率

Lable	样本	正确	识别率 (%)
T 恤	50	45	90
裤子	50	47	94
套衫	50	44	88
连衣裙	50	47	94
外套	50	45	90
凉鞋	50	47	94
衬衫	50	45	90
运动鞋	50	48	96
包	50	48	96
裸靴	50	47	94

最终 FPGA 工作于 100 Mhz 时钟频率下. 表 5 列出了 CNN IP 在配置为不同卷积块时的资源使用情况和计算速度. 功耗由 Vivado Power Report 获取.

表 5　CNN IP 资源消耗以及功耗

卷积块	LUT (53200)	FF (106400)	BRAM (140)	DSP (220)	时间 (us)	功耗 (W)
1	3680(7%)	2909(3%)	29.5(21%)	17(8%)	9094	0.247
2	2748(5%)	1592(1%)	33(24%)	30(14%)	4811	0.253
4	3912(7%)	2243(2%)	43.5(31%)	54(25%)	2605	0.375
8	6039(11%)	3527(3%)	70.5(50%)	106(50%)	1361	0.53

由表 5 可以看出当卷积块为 1 时, 一副图片的识别需要 9.094 ms, 完全满足实时性的要求, DSP 资源消耗 8%, BRAM 使用率达 21%, 在一些低成本的 FPGA 上可以实现, 应用在加速物联网边缘设备机器学习推理的开发. 在资源允许的情况下, 最高可以实现卷积块为 8 的并行度, 识别一副图片仅需 1.361 ms, DSP 资源消耗 50%, BRAM 使用率达 50%, 速度提高了近 5.68 倍. 在一些对时间要求非常高的场合, 可以利用资源去换取时间.

4　结论与展望

本文提出一种基于 ZYNQ 平台的服装识别系统. 采用 Verilog 设计服装识别 CNN 的 IP 核, 留出信号接口, 权重可加载, 通用性好, 便于移植, 大大缩减了 SOC 的开发周期, 对算法芯片化的施行也具有重要意义.

该系统采用软硬件协同的方式, 成功移植 Linux 操作系统, 利用 OpenCV 和 QT 将分割和识别结果通过 HDMI 传输接口进行实时显示, 达到有实时性好、界面友好, 并且具有较高的识别率和较强的扩展性.

参考文献

1 李东. 基于数字图像处理的服装款式识别方法研究[硕士学位论文]. 上海: 东华大学, 2017.

2 张振焕, 周彩兰, 梁媛. 基于残差的优化卷积神经网络服装分类算法. 计算机工程与科学, 2018, 40(2): 354–360. [doi: 10.3969/j.issn.1007-130X.2018.02.023]

3 Theis L, Korshunova I, Tejani A, *et al*. Faster gaze prediction with dense networks and Fisher pruning. arXiv: 1801.05787, 2018.

4 Gudovskiy DA, Rigazio L. ShiftCNN: Generalized low-precision architecture for inference of convolutional neural networks. arXiv: 1706.02393, 2017.

5 Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv: 1409.1556, 2014.

6 Tolias G, Sicre R, Jégou H. Particular object retrieval with integral max-pooling of CNN activations. arXiv: 1511.05879, 2015.

7 Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. arXiv: 1708.07747, 2017.

8 https://github.com/zalandoresearch/fashion-mnist.

9 李申煜. 基于 Zynq 的卷积神经网络加速器设计[硕士学位论文]. 北京: 北京理工大学, 2016.

10 余子健, 马德, 严晓浪, 等. 基于 FPGA 的卷积神经网络加速器. 计算机工程, 2017, 43(1): 109–114, 119. [doi: 10.3969/j.issn.1000-3428.2017.01.019]

11 Crockett LH, Elliot RA, Enderwitz MA, *et al*. The zynq book: Embedded processing with the arm cortex-A9 on the xilinx zynq-7000 all programmable soc. Strathclyde Academic Media, 2014.

12 周鑫. 全连接神经网络在 FPGA 上的实现与优化[硕士学位论文]. 合肥: 中国科学技术大学, 2018.