arXiv:2301.07050v1 [cs.LG] 17 Jan 2023

FPGA上的节能可重构自编码器实现

Murat Isik†, Matthew Oldland†, Lifeng Zhou

摘要——**自编码器是一种无监督神经网络，用于处理和压缩输入数据，然后将数据重构为原始数据大小。因此，自编码器可用于多种处理应用，例如数据压缩、图像分类、图像降噪和图像着色。在硬件方面，现场可编程门阵列（FPGA）等可重构架构凭借其灵活性、性能和能效的独特组合，已被用于加速多个领域的计算。本文考察了现有的各种自编码器，并采用卷积自编码器，在基于FPGA和图形处理器的实现中处理含噪静态MNIST图像。我们比较了基于FPGA和图形处理器的不同实现所取得的结果，并讨论了各自的优缺点。所提出设计的评估准确率达到80%；实验结果表明，所提出的加速器在100 MHz下可实现21.12 GOP/s的吞吐量，片上功耗为5.93 W。与现成器件及近期最先进实现的比较结果表明，所提出的加速器在能效和设计灵活性方面具有明显优势。我们还讨论了利用所提出加速器开展的未来工作。**

I. 引言

自编码器是一种无监督神经网络，旨在学习如何压缩输入数据，然后重建数据，使其尽可能准确地匹配输入数据，损失尽可能小 [1]。这通过编码器和解码器方法实现：编码器压缩数据，解码器重建数据 [1],[2],[3],[7]。机器学习中有许多类型的自编码器可供使用；例如简单自编码器、卷积自编码器、去噪自编码器和稀疏自编码器。这些不同的自编码器可用于满足不同的处理需求，例如存储前的数据压缩、图像分类以及去除图像中的噪声。

本文展示了卷积自编码器的应用，以及它们在去除图像噪声方面的有效性。卷积自编码器在编码和解码层中使用卷积神经网络，分别帮助压缩和重建图像。卷积神经网络使用不同的滤波器从数据中提取特征。这些滤波器旨在提取特定特征，具体取决于所需的特征。滤波器会扫描数据，并通过生成映射来表示数据与滤波器的匹配程度，得分越高，匹配度越高 [4]。滤波器按照用户设定的层结构，以步幅在图像上移动。随后，这些映射会用于下一层，该层可以是池化层或上采样层。池化层会查看一个数据窗口，并将窗口中的数据合并为一个样本。池化操作可以取数据的最大值、总和或平均值来生成新样本 [4]。编码过程中会使用池化层来压缩数据并存储重要特征，而解码过程中则使用上采样来辅助重建。使用不同硬件实现自编码器的方法有很多。本文比较了两种方法：一种是使用 VHDL 在现场可编程门阵列上实现，另一种是使用 Python 在图形处理器上实现。

在开发其余框架的过程中，机器视觉框架的开发人员可能会陷入对这些步骤该如何选择的斟酌之中。首次构建系统原型时，通常会选择组织选项。必须对原型应用进行测试，以确定每秒需要处理多少个片段，或每秒需要处理多少帧实时视频。嵌入式系统的功耗是大多数应用的一项重要考量，研究人员也致力于开发节能设计方法 [14-15]。

处理海量数据集的应用，以及高分辨率图像处理应用，都需要更快速、可配置、吞吐能力更强且能效更高的框架 [16-17]。现场可编程门阵列（现场可编程门阵列）能够发挥重要作用，因为它们具备可配置性、适应性和并行性，可满足相关应用所需的吞吐率 [13]。得益于其执行能力，现实世界中的应用可以在FPGA器件上实现。现场可编程门阵列的发展进一步提升了硬件的整体灵活性。最后，这些器件的应用开发工具链也取得了显著进步，使更多开发者能够使用这些器件 [18]。现场可编程门阵列通常用于要求并发性、高传输速度和可重新编程性的应用。

过去十年间，现场可编程门阵列迅速普及并不断发展，因为它们可用于范围极广的应用。提取特征的算法耗时较长，这在开发实时应用时是一个重大缺点。解决这一问题的方法之一，是为这些算法使用专用硬件，例如

†所有作者对本研究的贡献均相同。所有作者均来自德雷塞尔大学电气与计算机工程系。GitHub 链接：
https://github.com/Muratcanisik4/CNN-Autoencoder.git

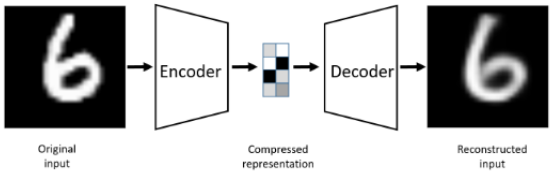

图1. 自编码器示例：输入图像压缩与重建 [1]

现场可编程门阵列能够提供专用功能模块，并行执行复杂的图像处理操作。过去十年间，现场可编程门阵列迅速普及并不断发展，因为它们可用于范围极广的应用 [19]。

我们设计了卷积自编码器的现场可编程门阵列和基于图形处理器两种实现方案，用于处理含噪静态MNIST图像。结果表明，所提出的设计在面积和功耗方面都具有高效性。

II. 自编码器类型

自编码器是一种无监督神经网络，可用于各种处理应用。它们经过训练，能够对输入数据进行编码与压缩，然后重建并解码数据，使其尽可能匹配输入数据。自编码器主要由三个部分组成 [3]:

- 编码与压缩，
- 瓶颈，
- 解码与重构。

自编码器的编码与压缩部分由多层神经网络组成，负责将数据压缩至指定大小。这些经过编码和压缩的数据被称为瓶颈部分，因为在重建之前，它是系统中最小的部分。如有需要，可以存储这些数据，从而减少所需的存储容量。压缩后的数据随后会用于系统的下一部分，即解码与重建部分。这部分负责解码已编码的数据，并将其重建为与输入数据相匹配的内容。该过程使用的神经网络与编码与压缩部分中的网络结构相同，但不是压缩数据，而是对数据进行上采样。压缩和重建部分所使用的层数由用户根据所需的实现方式设定。下方的图 1 展示了一个示例：输入图像经过压缩和重建后，得到一幅与其十分相似的输出图像 [2]。

机器学习中使用了许多不同类型的自编码器。这些自编码器具有相同的流程和组成部分，但编码和解码所使用的层有所不同。以下各节介绍了几种不同类型的自编码器。

A. 简单自编码器

简单自编码器是自编码器的基本实现，是一种简单的前馈网络，可以在编码和解码部分包含多个层。一种实现包含 3 层，各层彼此全连接 [1]。该实现还使自编码器的输入和输出大小相同，从而迫使自编码器学习压缩后的数据，且不产生信息损失 [1]。

B. 卷积自编码器

卷积自编码器与简单自编码器类似，但会使用卷积神经网络中的卷积层来替换原有层。卷积神经网络层由以下几个部分构成 [4]:

- 卷积，
- 修正线性单元，
- 池化（编码）或上采样（解码）。

卷积通过从数据中提取特征，并利用卷积窗口滤波器生成特征图来实现。这些滤波器用于保持像素与输入图像 [4]之间的关系。生成的特征会产生高分和低分，用于表示像素与滤波器的匹配程度；匹配得越好，得分越高。卷积有两个参数：填充和步长 [4]。填充用于确保无论输入数据的大小如何，特征都具有相同的处理范围。步长参数用于确定滤波器在生成特征时跨输入数据移动的像素数。

修正线性单元（ReLU）用于确保各层之间传递的数值均为非负数。具体做法是将发现的任何负数替换为零，以便后续计算。池化用于压缩图像。选定窗口大小后，将滤波后的特征中的部分内容合并为新图像中的一个像素，再传递到下一层。在池化窗口中，窗口内的最大值、平均值或求和值会被传递到新图像的下一个像素。解码与重建层会使用上采样代替池化，以增大图像尺寸，使其在重建过程中恢复到与输入尺寸相匹配的大小。

C. 去噪自编码器

去噪自编码器 [2] 在输入数据上加入高斯噪声，以扰乱数据，然后训练自编码器重建去除噪声后的数据。去噪自编码器旨在用于纠正受损数据中的错误。自编码器的效果通过输出数据与输入数据之间的重建损失来衡量。

D. 稀疏自编码器

稀疏自编码器在各层中利用稀疏性，以帮助降低重建误差，并在中间形成有意义的瓶颈图像。在 [2],中，通过在层中应用 L_1 正则化来强制实现稀疏性。这会将自编码器的优化目标改为:

$$\text{argmin}_{A,B} E[\Delta(x, B \circ A(x)] + \lambda \sum_i |a_i|, \qquad (1)$$

层	Type	窗口大小	步幅
1	卷积	4	1
2	最大池化	4	1
3	卷积	4	1
4	最大池化	4	1
5	卷积	4	1
6	最大池化	4	1
7	卷积	4	1
8	上采样	4	1
9	卷积	4	1
10	上采样	4	1
11	卷积	4	1
12	上采样	4	1
13	卷积	4	1

表 I 卷积自编码器的层构造

其中，a_i 表示第 i个隐藏层的激活值， i 遍历所有隐藏层激活值 [2]。[2], 中介绍的另一种方法是 KL 散度方法。该方法通过将激活值 a_i 转换为概率为 p [2]的伯努利变量，来衡量两个概率之间的距离。该方法通过获取一个批次的测量概率、计算其差值，然后将差值作为正则化因子来调整概率 p 。

III. 方法

我们假设图像已受到噪声污染，并且需要去除噪声，以便后续处理。我们设计了一种卷积自编码器，用于对受污染的输入图像进行图像降噪。该卷积自编码器采用三个卷积层和三个池化层交替排列，以进行编码和压缩。解码与重建部分采用四个卷积层和三个交替排列的上采样层。池化层采用最大池化操作，在图像窗口上进行处理。

表 I 展示了用于编码和解码的卷积自编码器层结构。

卷积层采用卷积神经网络进行设计，以训练权重，并在池化层或上采样层处理之前对图像进行特征提取。使用二维卷积作为滤波器，从像素中提取特征。所使用的一般二维函数公式如下：

$$f = \sum_{a=-y}^{y} \sum_{b=-z}^{z} x_{i-a,j-b} * k_{a,b}. \tag{2}$$

卷积滤波器对像素进行处理后，会执行 ReLU函数，以确保没有负数传递到下一层。这样可以保证数学运算始终得到正确结果，不会在计算时出错。所使用的 ReLU函数定义如下：

$$y = \max(0, x), \tag{3}$$

其中，x 是经过滤波的像素特征。ReLU函数的输出随后会传递到下一层，进行池化或上采样。池化函数是对经过滤波的像素特征在指定窗口大小内取最大值，其定义如下：

$$y = \max(x_1, 1, ..., x_{i,j}), \tag{4}$$

其中， i 和 j 是为该层选定的窗口大小。

IV. 结果

A. 数据集

该数据集是 MNIST 数字集的含噪变体，噪声源自一种内在的风格感知。训练集占全部数据的 80%，验证集占 20%。实际训练数据总计包含 60,000 张图像。由于当前任务所需的数据集必须包含含噪图像，我们定义了向数据集元素添加随机高斯噪声的函数。新数据集现在由含噪图像组成，并且可以进行可视化。



图 2. 项目含噪数据集

我们的含噪数据集如图 2 所示。训练数据总计包含 60k 张图像。我们使用随机高斯噪声函数，将 MNIST 数据集转换为含噪数据集。

高斯噪声是一种呈高斯分布形式的噪声，例如常见的白噪声。它具有随机值和脉冲特性。

B. 训练详情

最终我们选定了 20 轮次的训练时间，并在每个训练轮次结束时显示训练损失和验证损失。由于输入图像尺寸较小，所需的处理功耗不高，因此每个模型的训练时间约为 100 秒。虽然它有时会降低原本清晰可见的数字的可见度，但在清除数字并使其更加清晰方面效果更好，尤其是在第一幅图像等更具挑战性的情况下。训练参数数量不多，所得结果尚可，最高准确率为 80.83%。图 3 展示了自编码过程的输出。

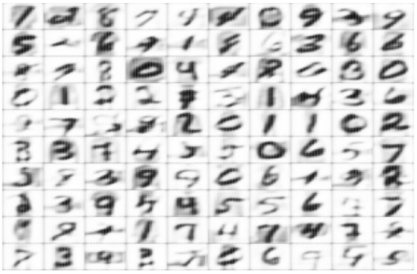

图 3. 去噪图像

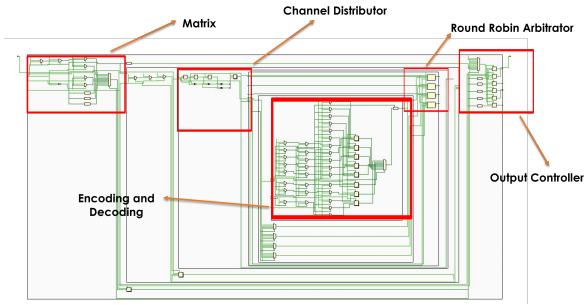

图 4. 实现方案的寄存器传输级设计

C. 实验设置

我们重点研究了卷积自编码器的两种硬件实现方案：一种采用图形处理器，另一种采用现场可编程门阵列。图形处理器实现使用 Python 和 Google Colaboratory 完成。Google Colaboratory 可在 NVIDIA Tesla K80 上运行机器学习 Python 可执行程序。我们在 XCZU7EV现场可编程门阵列芯片上实现了相同的设计，该芯片拥有 504K 个查找表（LUT），并使用了 Vivado 2021.1 版本的软件。

D. 现场可编程门阵列实现

使用由矩阵运算、通道分配器、编码模块、循环仲裁和输出控制器模块组成的卷积自编码器架构，对编码后的图像进行进一步分析。

矩阵运算：卷积运算期间的边界信息（28*28*1）。

通道分配器：矩阵被发送至通道分配器模块，该模块包含通道控制器和先进先出存储器。通道控制器对通道数据点进行交错处理，然后依次将其发送到先进先出队列。先进先出队列缓存交错后的数据，再将其分成八个通道。

编码与解码模块：首先，卷积编码模块执行卷积运算和激活，然后进行最大池化。解码通过上采样而非最大池化来实现。

轮询仲裁：卷积编码模块的输出数据会按顺序打包，然后发送至输出控制器。轮询仲裁器会优先处理请求方。

输出控制器：此模块用于判断数据流是否已处理完毕。

E. 比较

神经网络（NN）的实现一直采用中央处理器、图形处理器和专用集成电路。基于中央处理器/图形处理器的神经网络功耗很高，且受限于内存带宽，吞吐量有限。许多研究人员开发了定制专用集成电路，以加速网络推理工作负载，从而实现最佳性能和能效。尽管专用集成电路颇具吸引力，但它们无法提供足够的灵活性来适应神经网络的快速发展，而现场可编程门阵列作为可编程器件，能够构建独特的逻辑，从而缓解神经网络实现所面临的限制。因此，当前的研究热点之一是开发基于现场可编程门阵列、支持神经网络推理的硬件系统，以实现高吞吐量和高能效。

如表 II 所示，我们将所实现的卷积神经网络自编码器与文献中提出的其他卷积神经网络自编码器模型进行了比较。所提出设计的评估准确率达到 80%。实验结果表明，该加速器在 100 MHz 下可实现 21.12 GOP/s 的吞吐量，片上功耗为 5.93 W。与现成器件及近期最先进的实现方案进行比较后可见，所提出的加速器在能效和设计灵活性方面具有明显优势。

V. 结论与未来工作

在混淆矩阵中，模型确定了哪些类别的预测准确，哪些类别的预测不准确。增加训练参数并加深自编码器的层数，或许能提升模型性能。不过，就本研究而言，该模型在重建去噪照片方面表现出色。研究表明，若要基于编码器和解码器架构的区别，使用深度学习进行图像去噪，卷积层的使用至关重要。该模型已在多种异构设备上实现硬件加速，并在这些设备上构建出节能的可重构系统。因此，可以采用更深的模型和更合适的潜空间维度，以处理涉及 RGB 图像的更复杂去噪任务。作为本研究的下一步，未来的工作可以是在图像处理之前，去除自主机器人所搭载摄像头拍摄的图像中的噪声。我们正寻求将模型集成到机器人中，以便在通过无线方式接收来自不同机器人的图像后，若在检测位置之前图像过渡不清晰，便能通过清理图像作出响应。

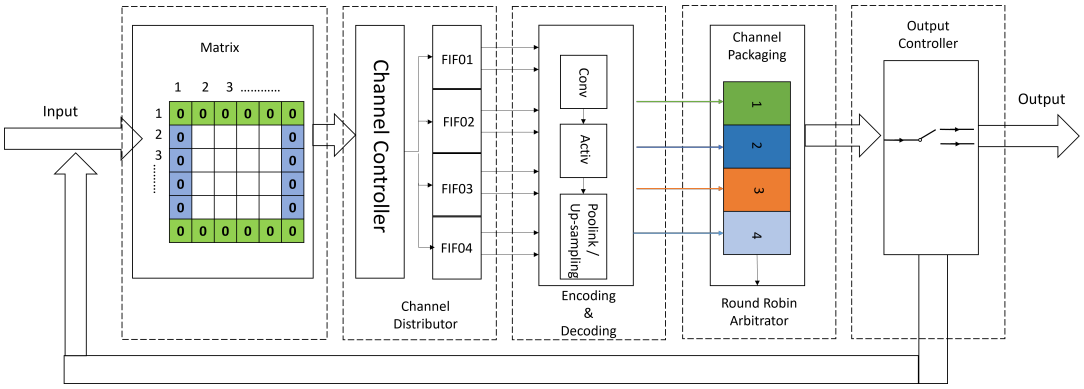

图 5. 实现框图

表 II 不同异构设备上的自编码器

平台	技术	工作频率	功耗	延迟	吞吐量	能效
NVIDIA K80	专用集成电路 (28nm)	$1.48GHz$	$30W$	$1137.62ms$	22000(GOP/s)	733。33($GOP/s/W$)
NVIDIA GTX 1080TI 图形处理器(16nm)		$1.48GHz$	$250W$	$6.15ms$	235.77(GOP/s)	0.94($GOP/s/W$)
Chakradhar等人 [12]	现场可编程门阵列(28nm)	$200MHz$	$15W$	–	16(GOP/s)	1。06($GOP/s/W$)
Gokhale等人 [11]	现场可编程门阵列(28nm)	$150MHz$	$8W$	$4.50ms$	23.18(GOP/s)	2.90($GOP/s/W$)
Zhang等人 [5]	现场可编程门阵列(28nm)	$100MHz$	$18.61W$	$21.61ms$	61.62(GOP/s)	3。31($GOP/s/W$)
Ours	现场可编程门阵列(16nm)	$100MHz$	$5.93W$	$2.91ms$	21。12(GOP/s)	3.56($GOP/s/W$)

参考文献

[1] Zhang, Yifei. “用于图像的更优自编码器：卷积自编码器。” ICONIP17-DCEC。在线获取：http://users. cecs. anu. edu. au/Tom. Gedeon/conf/ABCs2018/paper/ABCs2018_paper_58. pdf（访问日期：2017年3月23日）。2018年。[2] Bank, Dor, Noam Koenigstein, 和 Raja Giryes。“自编码器。” arXiv 预印本 arXiv:2003.05991（2020）。[3] Badr, Will。“自编码器：它是什么？又有什么用途？（第 1 部分）。” Medium，2021年12月9日，towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726。[4] Dataman, Chris Kuo/Dr. “用于图像降噪的卷积自编码器。” Medium，2022年9月8日，towardsdatascience.com/convolutional-autoencoders-for-image-noise-reduction-32fce9fc1763.[5] Zhang, Chen 等。“面向深度卷积神经网络的现场可编程门阵列加速器设计优化。”《2015年 ACM/SIGDA 现场可编程门阵列国际研讨会论文集》。2015年。[6] Xu, Jiayang 和 Karthik Duraisamy。“用于时空动力学参数化预测的多层卷积自编码器网络。”《计算机在应用力学与工程中的方法》372（2020）：113379。[7] Turchenko, Volodymyr, Eric Chalmers 和 Artur Luczak。“Caffe 中带有池化-反池化层的深度卷积自编码器。”arXiv 预印本 ar Xiv:1701.04949（2017）。[8] Chen, Min 等。“利用卷积自编码器神经网络进行医学图像分析的深度特征学习。”IEEE《大数据汇刊》7.4（2017）：750-758。[9] Koushik, Jayanth。“理解卷积神经网络。”arXiv 预印本 arXiv:1605.09081（2016）。[10] Wang, Wei 等。“广义自编码器：一种用于降维的神经网络框架。”《IEEE计算机视觉与模式识别会议研讨会论文集》。2014年。[11] Gokhale, Vinayak 等。“一种用于深度神经网络的 240 g-ops/s 移动协处理器。”《IEEE计算机视觉与模式识别会议研讨会论文集》。2014年。[12] Chakradhar, Srimat 等。“一种用于卷积神经网络的动态可配置协处理器。”《第37届计算机体系结构年度国际研讨会论文集》。2010年。[13] Woods, Roger 等。《信号处理系统的现场可编程门阵列实现》。约翰·威立父子出版公司，2008年。

[14] Camposano, Raul 和 Jörg Wilberg。《嵌入式系统设计》。嵌入式系统设计自动化 1.1（1996）：5-50。[15] Polzer, Andreas 等。《管理基于模型的嵌入式软件产品线的复杂性与可变性》。系统与软件工程创新 8.1（2012）：35-49。[16] Kattenborn, Teja 等。《卷积神经网络（卷积神经网络）在植被遥感中的综述》。ISPRS 摄影测量与遥感杂志 173（2021）：24-49。[17] Benzigar, Mercy R. 等。《柔性超级电容器新兴材料的进展：当前趋势与未来展望》。《先进功能材料》30.40（2020）：2002993。[18] Sass, Ronald 和 Andrew G. Schmidt。《基于平台型现场可编程门阵列的嵌入式系统设计：原理与实践》。Morgan Kaufmann，2010。[19] Trimberger, Stephen M. Steve。《现场可编程门阵列的三个时代：回顾现场可编程门阵列技术的前三十年：本文探讨摩尔定律如何推动现场可编程门阵列的设计经历三个训练轮次：发明时代、扩张时代和积累时代》。《IEEE固态电路杂志》10.2（2018）：16-29。