

DOI: 10.20098/j.cnki.1001-5132.2025.0711

基于对抗网络的可配置加解密硬件优化设计

涂江涛, 张会红, 张跃军*

(宁波大学 信息科学与工程学院, 浙江 宁波 315211)

摘要: 随着神经密码学的出现, 越来越多研究使用神经网络来训练加解密算法, 其中采用对抗网络可实现端到端的高安全加解密, 但存在开销大、速度慢等问题。通过对运算单元核心、数据存储架构和数据流行为进行协同优化设计, 提出一种面向神经网络的可配置加解密硬件设计方案。该方案首先对加解密模型进行硬件友好型优化, 完成网络训练和量化; 然后, 采用 Winograd+DSP48 的卷积加速方法, 将所需 96 个乘法器降低到 32 个; 最后, 设计 CPU 控制与调度系统架构, 结合动态控制加速器的操作模式, 实现高性能可配置加解密硬件电路。实验结果表明, 所提方案最高工作频率为 133 MHz, 功耗为 32.4 mW, 吞吐量为 17.06 GOPs。加解密网络的正确率达 100%, 破解网络正确率接近 50%, 硬件电路具备可配置和高安全特性。

关键词: 神经密码学; 对抗神经网络; 可配置; 电路优化设计

中图分类号: TP391

文献标志码: A

文章编号: 1001-5132 (2026) 01-0001-07

Configurable encryption and decryption hardware design based on adversarial networks

TU Jiangtao, ZHANG Huihong, ZHANG Yuejun*

(Faculty of Electrical Engineering and Computer Science, Ningbo University, Ningbo 315211, China)

Abstract: With the emergence of neural cryptography, an increasing number of studies employ neural networks to train encryption and decryption algorithms. While the use of adversarial networks enables end-to-end high-security encryption and decryption, challenges such as high overhead and low speed persist. To address this problem, this paper proposes a configurable encryption and decryption hardware design scheme for neural networks through the co-optimization of computational unit cores, data storage architectures, and data flow behavior. The proposed scheme first optimizes the encryption and decryption model for hardware compatibility, completing network training and quantization. Secondly, it employs a Winograd+DSP48-based convolution acceleration method, reducing the required multipliers from 96 to 32. Finally, a CPU control and scheduling system architecture is designed, integrating dynamic control of accelerator operation modes to achieve a high-performance configurable encryption and decryption hardware circuit. Experimental results demonstrate that the proposed solution achieves a maximum operating frequency of 133 MHz, with a power consumption of 32.4 mW, and a throughput of 17.06 GOPs. The accuracy rate of encryption and decryption in the network reaches 100%, the accuracy rate of cracking the network approaches 50%, and the hardware circuit features configurability and high security.

Keywords: neural cryptography; adversarial neural networks; configurable; optimized circuit design

收稿日期: 2025-07-16

宁波大学学报(理工版)网址: <http://journal.lg.nbu.edu.cn/>

基金项目: 国家自然科学基金(62474100, 62134002); 浙江省“尖兵领雁+X”科技计划项目(2025C01063); 宁波市科创甬江 2035 重点研发计划(2024T016); 宁波大学“双一流”合作专项定向委托科技合作项目(HX2024000574, HX2025000106)

第一作者: 涂江涛, 硕士研究生, 主要研究方向为集成电路和信息安全。E-mail: 2311100020@nbu.edu.cn

*通信作者: 张跃军, 教授, 主要研究方向为集成电路和信息安全。E-mail: zhangyuejun@nbu.edu.cn

随着计算机技术逐渐进步, DES、IDEA、RC5、CAST、BLOWFISH、3DES 和 RSA 等传统密码体制正面临着严峻的考验。量子计算机研究的发展和 Shor 算法等的出现,使密码学的安全性受到极大的威胁。随着神经网络的高速发展,有研究开始探索将神经网络与密码学相结合。2012 年,Shukla 等^[1]提出两种人工神经网络用在密码学上,仿真结果显示该加密算法的安全性很高。2016 年,Abadi 等^[2]引进一种方法,与之前的加密方法有着很大的不同,它利用对抗生成网络(Generative Adversarial Networks, GAN)自主学习训练加密算法,即将对抗性神经网络运用到密码学中来保护通信。在 Abadi 的工作提出之后,出现了一系列针对该模型的安全性和在其基础上拓展应用的研究。Coutinho 等^[3]基于 GAN 模型设计一个攻击网络来模拟选择明文攻击,并通过神经网络学习到 OTP(One Time Pad)算法,实现加解密算法安全性提升,但是该方案无法处理大量信息。Li 等^[4]设计了一种能够学习对称加密的端到端神经网络架构,该架构增加了不同的攻击网络,以模拟实际加密通信中的复杂环境。在训练过程中,系统会根据所遇到的攻击,有针对性地学习安全的加密算法。

在加密领域,对加密操作的性能、功耗和安全具有严苛要求。Abadi 等^[2]提出的对抗加密模型具有 4 层卷积操作,涉及大量乘积累加操作。卷积运算具有高度的并行性,通用处理器或图形处理器无法高效利用这种并行性。硬件加速器可以针对这种并行性进行专门设计,例如采用多个并行处理单元,每个处理单元负责一部分卷积计算任务,从而显著提高计算速度。这对于将对抗神经网络加密应用于嵌入式与边缘场景具有现实意义,是满足实时安全需求、开启新兴应用场景的战略必需。

1 对抗神经网络加解密模型分析

1.1 对抗神经网络加解密模型

2016 年, Abadi 所在的 Google Brain 团队提出由名为 Alice 和 Bob 的神经网络组成的多智能体系统,目标是限制名为 Eve 的神经网络窃听 Alice 和 Bob 之间通信的内容。该系统没有为这些神经网络规定特定的加密算法,而是通过端到端的对抗训练实现高安全性加解密,以及选择性地应用各种加密

和解密操作以满足机密性目标。

3 个神经网络具有相同的神经网络结构,这样有利于它们训练收敛^[5]。Alice 和 Bob 具有共同的密钥, Eve 没有权限获取密钥。在训练期间 Bob 要尽可能解密出 Alice 训练生成的密文,同时要防止 Eve 破解密文。如果 Eve 成功解密,它的高准确性将促使 Alice 生成更复杂的密文,以防止 Eve 解密密文。因此, Alice 与 Eve 对抗并试图生成在没有秘密密钥的情况下难以解密的密文,而 Eve 试图解密这些密文与 Alice 对抗。这个“游戏”一直持续,直到 3 个神经网络达到平衡,最终使得 Bob 能够利用密钥从密文重构明文^[6],获得接近 100%的解密准确率,而 Eve 在没有密钥的情况下只能得到 50%左右的解密正确率。但该系统在软件端 CPU 上单次加密的延迟超 16 ms,这在实际运用中是难以满足实时通信要求的。

1.2 对抗网络加解密模型优化

本文提出基于对抗网络的加解密模型(图 1),在不影响加解密正确率的前提下提高加解密速度,降低网络硬件电路开销,网络参数配置见表 1。网络卷积核尺寸均为 1×3 。使用 1×3 卷积核能够有效平衡计算效率、模型表达能力与特征提取能力,可以在保证模型性能的同时显著降低计算开销。单个神经网络包含 1 个全连接层、4 个卷积层和 1 个池化层。网络将 N 位明文和 N 位密钥通过全连接层转换为 $2N$ 位输出,再进行卷积、激活和池化操作,输出 N 位密文。同时网络在训练时,卷积层舍去卷积层偏置参数,因为在量化硬件实现中,偏置的累加操作会引入额外量化误差,且这些误差在逐层计算中可能被放大,导致模型精度显著下降^[7]。

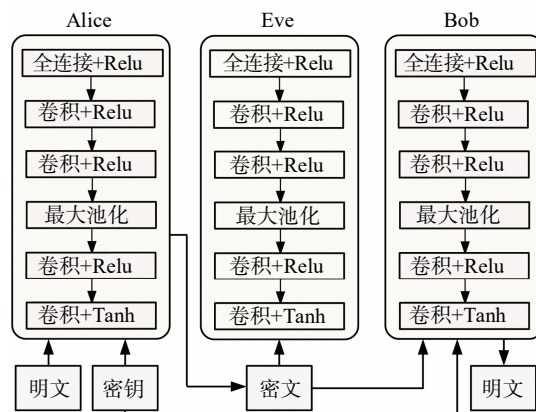


图 1 基于对抗网络的加解密模型

表1 网络参数配置

序号	模块	激活函数	输入尺寸	卷积核数量	卷积核尺寸	步长	保持尺寸	输出尺寸
1	全连接	ReLU	$1 \times 2N \times 1$	—	—	—	—	$1 \times 2N \times 1$
2	一维卷积	ReLU	$1 \times 2N \times 1$	2	1×3	1	是	$1 \times 2N \times 2$
3	一维卷积	ReLU	$1 \times 2N \times 2$	4	1×3	1	是	$1 \times 2N \times 4$
4	最大池化	—	$1 \times 2N \times 4$	—	—	2	—	$1 \times N \times 4$
5	一维卷积	ReLU	$1 \times N \times 4$	4	1×3	1	是	$1 \times N \times 4$
6	一维卷积	Tanh	$1 \times N \times 4$	1	1×3	1	是	$1 \times N \times 1$

网络训练完成后, 进行 Int8 训练推理量化, 将权重和激活值的数据形式由 Float32 转换为 Int8, 从而显著减少推理时的计算开销和内存占用。Pytorch 框架的 Post Training Static Quantization 提供一种线性量化方法^[8], 如图 2 所示, 其中 r 为浮点数, q 为量化后的整数。浮点张量的动态范围 $r_{\max}$ 和 $r_{\min}$ 预先计算出, 固定的缩放因子 S 和零点 Z 按下式计算, 其中 round 函数为四舍五入操作。

$$S = (r_{\max} - r_{\min}) / (q_{\max} - q_{\min}) \quad (1)$$

$$Z = \text{round}(q_{\max} - r_{\max} / S) \quad (2)$$

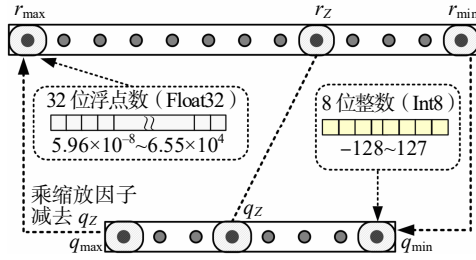


图2 Int8 线性量化示意图

卷积运算由式 (3) 定义, 通过量化转换将其浮点数替换为 Int8 整数, 得到量化后的式 (4)。

$$r_3^{i,k} = \sum_{j=1}^N r_1^{i,j} r_2^{j,k} \quad (3)$$

$$S_3(q_3^{i,k} - Z_3) = \sum_{j=1}^N S_1(q_1^{i,j} - Z_1) S_2(q_2^{j,k} - Z_2) \quad (4)$$

其中 Z_1 、 Z_2 、 Z_3 、 S_1 、 S_2 、 S_3 分别表示浮点数 r_1 、 r_2 、 r_3 的零点和缩放因子。

对式 (4) 变换处理得到轻量型卷积运算公式:

$$q_3^{i,k} - Z_3 = \frac{S_1 S_2}{S_3} \sum_{j=1}^N (q_1^{i,j} - Z_1)(q_2^{j,k} - Z_2) \quad (5)$$

此时, 只有 S_1 、 S_2 和 S_3 为浮点数。使用 M 来表示 $S_1 S_2 / S_3$, M 通过 frexp 函数可分解为 2^{-n} 和指数 M_0 的乘积, 如公式 (6) 所示。 2^{-n} 在硬件中可以用右移 n 位操作实现。这样整个卷积操作就转化为定点数运算和右移操作。

$$M = S_1 S_2 / S_3 = 2^{-n} M_0 \quad (6)$$

权重量化通常会导致精度下降 3% 左右, 但网络最后一层的 Tanh 函数使得在硬件设计时, 即使缺失一些精度仍能保持正确输出。

2 对抗加解密网络的可配置硬件设计

所提出的可配置加解密硬件设计主要包含以下 4 个方面: 不同位数的整型数据转换电路、卷积运算核心电路、控制与调度模块和模式选择模块。

2.1 整型数据转换电路

在卷积计算过程中, 鉴于数据量化结果为 Int8, 以 Int16 类型表示权重和数据乘积结果。为进行累加操作, 采用 Int18 类型来存储累加过程的中间结果。当累加操作完成后, 须将 Int18 类型的结果转换回 Int8 类型, 以确保数据格式的一致性和后续处理的兼容性。具体变换步骤如图 3 所示。

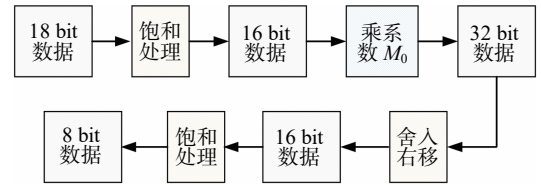


图3 Int18 转 Int8 变换步骤

首先, 进行饱和处理, 将数据截断至 Int16 范围。接着与一个 UInt16 类型的指数 M_0 相乘, 得到 Int32 类型的结果。然后, 进行舍入操作, 右移 ($15 + \text{Shift_Factor}$) 位, 以得到 Int16 类型的数据。最后, 再次进行饱和处理, 将 Int16 类型的数据截断为 Int8 类型, 以完成整个数据变换过程。这一系列操作步骤旨在确保经过量化处理后的数据结果与原始的 Pytorch 量化推理结果保持一致^[9]。

2.2 轻量型卷积运算核心电路

对于资源受限的 FPGA 芯片, 其 LUT 资源足够, 但 DSP 资源比较匮乏, 比如本文使用的

ZYNQ7010, 仅有 80 个 DSP 资源, 在保留 20% 裕量前提下, 仅有 64 个可用。本文通过将 FPGA DSP48 资源和 Winograd 算法结合, 提出一种仅需 40 个 DSP48 资源的卷积运算核心电路设计。

Winograd 算法以少量加法为代价换取乘法数量的减少, 特别对于 1×3 卷积核运算, 能够减少 2 个乘法器。对于卷积核尺寸为 1×3 、输入为 1×4 的 Winograd 卷积操作如式 (7) 所示^[10], 通过线性变换将输入和卷积核映射到新空间, 在新空间中执行逐元素乘积累加。

$$F(2,3) = \begin{bmatrix} d_0 & d_1 & d_2 \\ d_3 & d_4 & d_5 \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} m_0 + m_1 + m_2 \\ m_1 - m_2 - m_3 \end{bmatrix} \quad (7)$$

$$m_0 = D_0 G_0, m_1 = D_1 G_1, m_2 = D_2 G_2, m_3 = D_3 G_3 \quad (8)$$

D 和 G 分别为转换后的输入和权重, 转换公式为

$$D_0 = d_0 - d_2, D_1 = d_1 + d_2, D_2 = d_2 - d_1, \\ D_3 = d_1 - d_3 \quad (9)$$

$$G_0 = g_0, G_1 = (g_0 + g_1 + g_2) / 2, \\ G_2 = (g_0 - g_1 + g_2) / 2, G_3 = g_2 \quad (10)$$

G_0 、 G_1 、 G_2 和 G_3 可以提前在软件端算出, D_0 、 D_1 、 D_2 和 D_3 使用硬件电路中的加法进行计算。原先卷积运算所需要的乘法由 6 次减少到 4 次, 加法由 4 次增加到 8 次, 但加法的开销远小于乘法的开销。

ZYNQ7010 的 FPGA 拥有 DSP48E2 资源, 能够完成 25 位乘 18 位有符号数的乘法操作。为有效利用 DSP 资源, 将两个权重参数进行拼接, 同时与一个输入数据相乘。具体为将权重 A 和权重 B

各自填充为 25 位数据后相加。权重 A 放置在 25 位数据的高 9 位, 最高位为符号位填充, 其余位为 0。权重 B 放置在 25 位数据的低 8 位, 其余位由其符号位填充。最终经过相乘后输出 43 位数据, 其中低 16 位结果为 $B \times C$, 中间 16 位结果为 $A \times C$ 。

本文提出乘加 (Multiply-Accumulate, MAC) 阵列硬件结构的设计, 如图 4 所示。将转换的 $1 \times 4 \times 2$ 权重数据进行拼接后, 再和转换后的 $1 \times 4 \times 1$ 输入数据一起输入 Winograd 卷积运算模块中, 使用 DSP48 进行乘法操作。一个 MAC 阵列同时进行 4 个乘法操作和 4 个 2 级流水加法树操作, 1 个时钟延迟后完成 $1 \times 2 \times 2$ 输出, 表示完成 2 个通道的 2 个数据输出。基于这样的数据输出格式, 进行最大池化操作时无需额外缓存, 可以直接对同一通道输出进行比较。MAC 阵列需要 4 个乘法, 而常规卷积操作完成相同输出需要 12 个乘法, 所以单个 MAC 阵列乘法数减少 $2/3$ 。

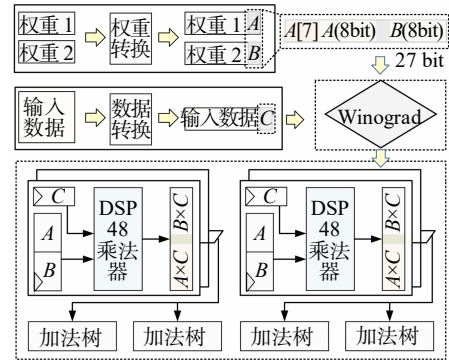


图 4 MAC 阵列硬件结构

2.3 整体硬件电路结构

整体硬件电路结构如图 5 所示。

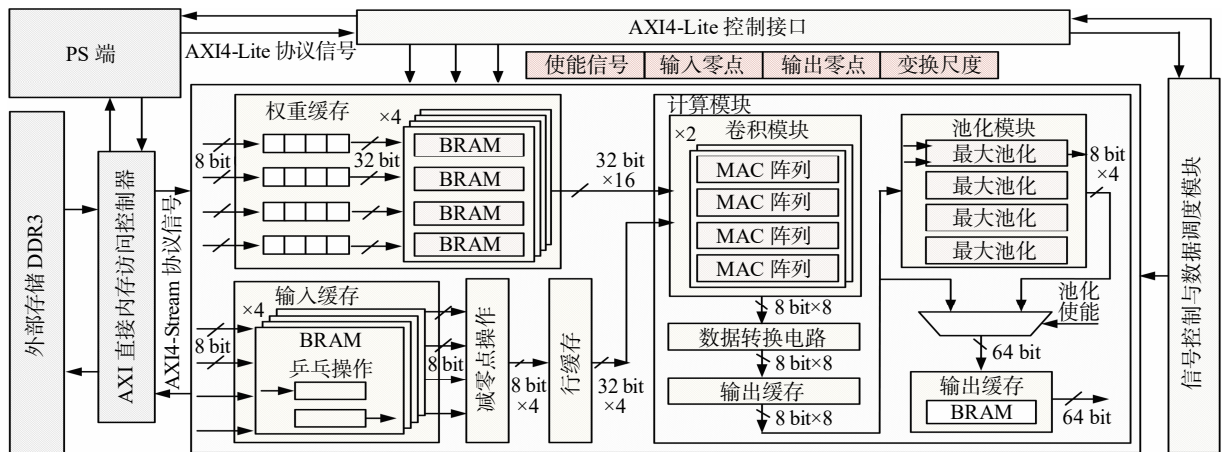


图 5 整体硬件电路结构

权重量化和 Winograd 算法权重均是网络训练成功后在软件端计算得出,其余操作均在硬件实现。ZYNQ PS 端实现信号的整体调度,初始化时通过直接内存访问控制器将权重参数和输入数据输入到加速器中,分别存储到权重缓存和输入缓存中。然后 PS 端通过 AXI4-Lite 接口向加速器传入使能信号、输入零点、输出零点和变化尺度等信息。数据调度与控制模块收到开始信号后向计算模块提供 Block RAM 起始写地址、起始读地址、有效信号和完成信号等,来控制正确的数据流在计算模块中流动,并通过 AXI4-Lite 控制接口向 PS 端提供模块状态信号。

经过量化后,数据以 Int8 的形式进行卷积运算,单个通道的数据位宽为 8。设计并行度选择 4,正好对应前文设置网络的最大输出通道数。采用 32 位的输入输出数据带宽,AXI 直接内存访问控制器,提供 32 位数据流传输,即同时实现 4 个通道的数据传输。权重数据通过 AXI 直接内存访问控制器以 32 位流水形式传入。将这 32 位的数据分成 4 组 8 位数据,经过 4 个时钟周期后,每组数据串转并输出 32 位数据,分别分配到 4 个 Block RAM 中。输入数据同样分成 4 组 8 位数据存入 Block RAM 中,输入缓存模块使用乒乓操作进行输入输出来提高电路吞吐率,再经过减零点和行缓存操作后输入计算模块。

计算模块主要包含卷积模块和池化模块。卷积模块包含 8 个 MAC 阵列,每个 MAC 阵列可以完成 8 组 Int8 的卷积运算,总共可以同时进行 64 组卷积运算。卷积运算输出依次进入数据转换电路和激活函数电路,之后作为池化模块的输入。输出缓存的输入数据由池化使能信号进行选择。当输出缓存读地址有效时,开始数据的读出。

2.4 控制与调度寄存器

硬件加速器作为专用计算单元,其核心功能聚焦于高效执行卷积运算。为实现系统级协同工作,须引入外部控制机制进行任务调度与资源管理^[11]。为构建更灵活高效的解决方案,本文通过 SoC 架构集成可编程处理器核,并利用轻量级总线协议实现关键控制。

控制与调度寄存器由 2 个寄存器构成。CPU 通过 AXI4-Lite 总线对这 2 个寄存器进行读写,从

而实现对卷积加速器的控制。第一个寄存器是控制寄存器,完成任务的初始准备、信号使能和任务有效等控制信号,如图 6 所示。第二个寄存器用于存储量化参数中的输入零点、输出零点和激活零点,如图 7 所示。为充分利用寄存器,把乘积因子 M_0 和移位值 n 分别放入 REG0 和 REG1 剩余的位置。

加速器控制寄存器地址: 0x00

保留	[26:11]	[10:9]	8	7	6	5	4	3	2	1	0
REG0[26:11]	乘积因子 M_0	REG0[4]	池化使能	REG0[10:9]	缓存数据选择	REG0[3]	缓存数据选择	REG0[8]	输入长度选择	REG0[2]	卷积开始
REG0[7]	权重输入	REG0[1]	权重输入	REG0[6]	任务有效信号	REG0[0]	数据输入	REG0[5]	乒乓操作		

图 6 加速器控制寄存器

量化参数寄存器地址: 0x04

保留	[27:24]	[23:16]	[15:8]	[7:0]
REG0[27:24]	移位值 n	REG0[15:8]	输出零点	REG0[23:16]
REG0[23:16]	激活零点	REG0[7:0]	输入零点	

图 7 量化参数寄存器

3 实验结果与分析

本文所提对抗神经网络软件端在 python 3.7、Anaconda 3 和 pytorch (GPU 版本) 框架环境下完成训练和数据量化;采用 Modelsim 进行电路功能仿真验证;使用 Vivado 软件进行硬件电路综合和布局布线,量化后的加密数据和权重数据通过 DMA IP 核传入加解密加速电路中,Xilinx SDK 软件进行 PS 端代码编写来控制数据输入、输出和加速器参数配置,最后在正点原子领航者 ZYNQ 开发板上进行板级调试和功能验证,加解密速度相对于软件实现 2.5 倍的提升。

3.1 模型训练结果

本文所提可配置电路设计可以实现不同长度的明文和密钥加解密,但如果明文长度过大,网络训练难以收敛,全连接层也因权重参数增加使得加解密过程变慢。这里选择 16、32 和 64 位的明文和密钥进行实验。

图 8 展示密钥和明文长度分别为 16、32 和 64 位时,网络训练 200 个 epoch 的准确率和损失率进化曲线。可以看到当加密长度较短时,优化后模型 Bob 的准确率可以达到 100%,Eve 的准确率接近 50%。表明 Bob 可以完全恢复出明文,而 Eve 无法破解密文,只能输出随机猜测值。但随着加密明文

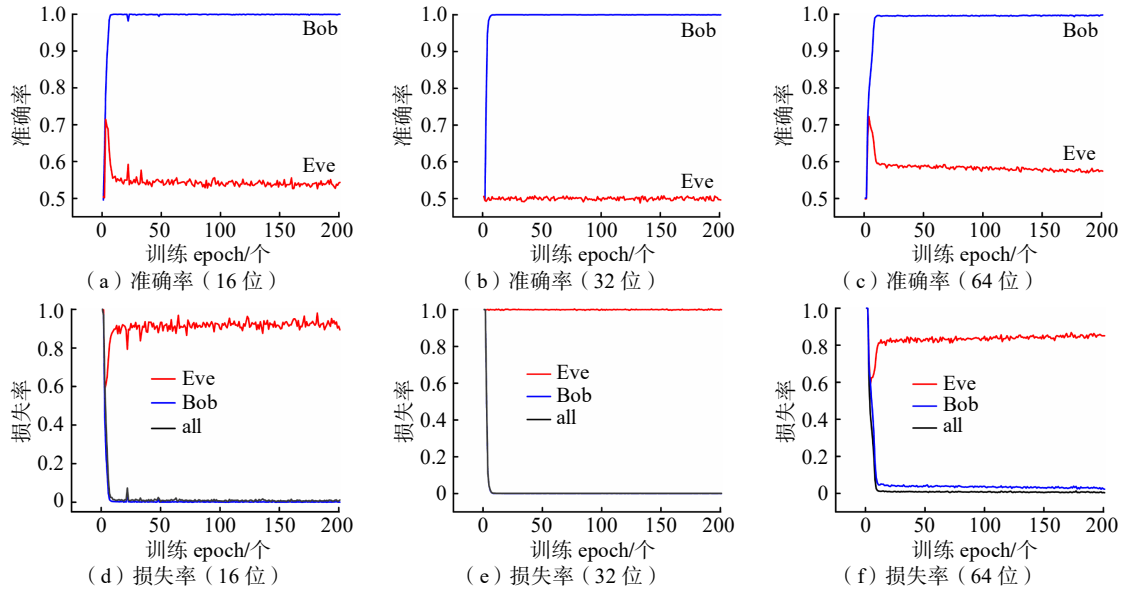


图8 不同长度明文和密钥的准确率和损失率进化曲线

长度增加,训练逐渐困难,网络收敛速度变慢。当明文长度为64位时,虽然Bob解密正确率可以达到100%,但Eve的破解率也远高于50%,表明加密安全性降低。这可能是加密算法对长序列信息建模能力不足的体现,长序列可能引发梯度消失/爆炸,导致Bob和Eve均难以收敛,所以整个网络的训练可能侧重于Bob完全恢复出密文,而失去限制Eve正确率的能力。图9是加密网络Alice和解密网络Bob训练好后各层权重值域和Winograd转换后的权重值域。可以看到每层权重数据都处于-3~5范围,使用Int8形式足以表示出权重参数。

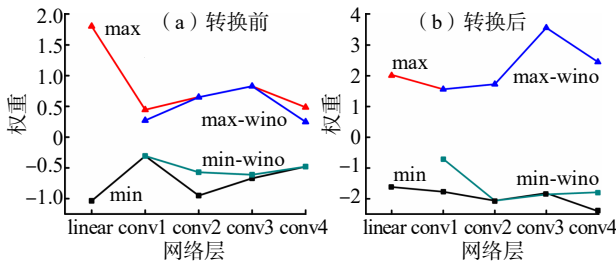


图9 各网络层权重范围

3.2 加解密硬件加速电路时序分析

所提加解密加速电路部分仿真时序如图10所示,图中包含卷积层的输入和权重数据加载、卷积计算、量化、激活和池化等操作的时序信息。行缓存、卷积、量化、激活和池化分别需要4、8、6、1和2个时钟延迟。以上几种操作均采用非阻塞的流水运算,所以从卷积运算开始19个时钟后输出

有效的数据。池化使能后需要额外2个时钟延迟。

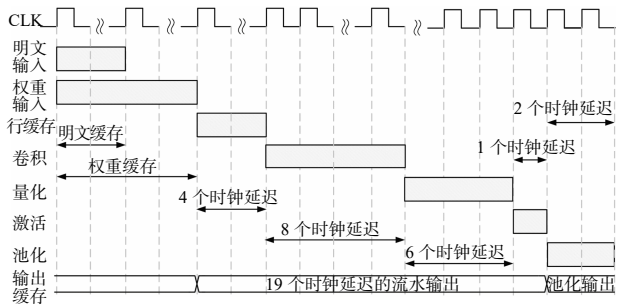


图10 加解密加速电路时序图

3.3 电路开销和功耗分析

加解密加速电路的功耗分布情况如图11所示,在所提电路允许的最高时钟频率133 MHz下,总功耗为32.4 mW,设计实现17.06 GOPs的峰值吞吐量,能效比为 5.265×10^{11} 次/(s·W)。缓存和卷积模块功耗之和占总功耗的76.58%。量化、激活和池化模块功耗占比分别为6.76%、2.25%、2.25%。控制信号、AXI接口和DMA控制模块的功耗分别占4.95%、4.50%和2.70%。

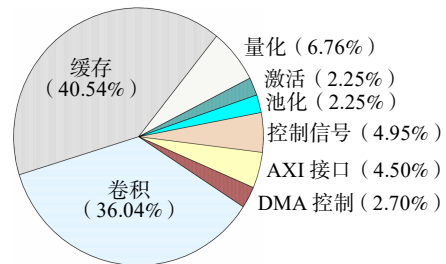


图11 加解密加速电路功耗分布图

所提电路总共使用 2 546 个 LUT 单元、40 个 DSP48 单元、3 072 个 FF 和 4 个 BRAM16K, 用于实现输入、权重和行缓存以及卷积、量化、池化和数据控制单元等功能模块, 具体资源使用情况见表 2。卷积模块除了占用数量相对最多的 LUT 和 FF 外, 还使用 32 个 DSP48, 是计算量最大、使用资源最多的单元。量化模块则消耗另外 8 个 DSP48。

表 2 资源使用情况

功能模块	LUT	FF	DSP48	BRAM16K
输入缓存	32	—	—	4
权重缓存	394	656	—	—
行缓存	128	128	—	—
卷积模块	1 206	1 548	32	—
激活模块	48	64	—	—
池化模块	48	64	—	—
量化模块	380	296	8	—
控制信号模块	119	84	—	—
AXI 接口模块	165	174	—	—
DMA 控制模块	26	25	—	—
总计	2 546	3 072	40	4

4 结论

本文在分析对抗神经网络加解密模型的整体性能基础上提出优化的网络模型, 采用硬件方式重构卷积操作的运算核心, 完成数据格式转换、运算单元、存储架构和数据流控制等功能电路的硬件设计, 进而提出一种对抗神经网络加解密加速电路的可配置硬件设计。本方案使用 Int8 数据直接运算, 而无需常规量化方案的反量化操作。通过对卷积运算进行优化, 相对于常规卷积所需乘法个数减少 2/3。同时采用软硬协同的方式, 实现加解密数据规模和操作模式的可配置。实验结果表明, 所提加解密加速电路具有解密正确率高、硬件开销少和功耗低等特点, 提出的方案对提高对抗神经网络加解密电路的效率和灵活性具有重要借鉴意义。

参考文献:

[1] SHUKLA N, TIWARI A. An empirical investigation of using ANN based N-state sequential machine and chaotic neural network in the field of cryptography[J]. Global journal of computer science and technology D: neural &

artificial intelligence, 2012, 12(10):17-26.

- [2] ABADI M, ANDERSEN D G. Learning to protect communications with adversarial neural cryptography [EB/OL]. [2025-03-21]. <https://arxiv.org/abs/1610.06918>.
- [3] COUTINHO M, DE OLIVEIRA ALBUQUERQUE R, BORGES F, et al. Learning perfectly secure cryptography to protect communications with adversarial neural cryptography[J]. Sensors, 2018, 18(5):1306.
- [4] LI Z Z, YANG X Y, SHEN K Q, et al. Information encryption communication system based on the adversarial networks foundation[J]. Neurocomputing, 2020, 415: 347-357.
- [5] KLEIN E, MISLOVATY R, KANTER I, et al. Synchronization of neural networks by mutual learning and its application to cryptography[EB/OL]. [2025-03-11]. https://papers.nips.cc/paper_files/paper/2004/file/5a45828dead8c065099cb653a2185df1-Paper.pdf.
- [6] SUN Y N, LO F P, LO B. Lightweight Internet of Things device authentication, encryption, and key distribution using end-to-end neural cryptosystems[J]. IEEE internet of things journal, 2022, 9(16):14978-14987.
- [7] SOLOVYEV R, KUSTOV A, TELPUKHOV D, et al. Fixed-point convolutional neural network for real-time video processing in FPGA[C]//2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). Saint Petersburg and Moscow, Russia. IEEE, 2019:1605-1611.
- [8] JACOB B, KLIGYS S, CHEN B, et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference[C]//2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA. IEEE, 2018:2704-2713.
- [9] CHEN X, LI J D, ZHAO Y. Hardware resource and computational density efficient CNN accelerator design based on FPGA[C]//2021 IEEE International Conference on Integrated Circuits, Technologies and Applications (ICTA). Zhuhai, China. IEEE, 2021:204-205.
- [10] LI S Z, WANG Q, JIANG J F, et al. An efficient CNN accelerator using inter-frame data reuse of videos on FPGAs[J]. IEEE transactions on very large scale integration (VLSI) systems, 2022, 30(11):1587-1600.
- [11] KIM M, OH K, CHO Y, et al. A low-latency FPGA accelerator for YOLOv3-tiny with flexible layerwise mapping and dataflow[J]. IEEE transactions on circuits and systems I: regular papers, 2024, 71(3):1158-1171.

(责任编辑 韩超)