文章编号:1671-4598(2026)02-0227-08 DOI:10.16526/j.cnki.11-4762/tp.2026.02.030 中图分类号:TP389.1 文献标识码:A

基于知识蒸馏的量化卷积神经网络模型压缩研究

何龙超，武唯康，李　斌，常迎辉
（中国电子科技集团公司 第五十四研究所，石家庄　050081）

摘要：针对边缘设备部署深度卷积神经网络存在的高资源消耗问题，对知识蒸馏与低比特量化协同优化方法进行了研究；采用了量化感知训练与蒸馏损失联合指导的关键技术，通过教师模型软标签监督和投影梯度下降优化，有效缓解了低比特量化的精度损失；在CIFAR-10和CIFAR-100数据集上的实验分析与验证，该方法实现了ResNet系列网络的4位量化，在CIFAR-10上达到92.1%的准确率，模型大小压缩至0.41 MB；经FPGA端侧部署验证，ResNet-20推理时延从82.3 ms降至5.67 ms，满足了边缘计算对低延迟与高效率的工程需求；证实该方法能在保持精度的同时显著降低资源开销，为资源受限环境下的神经网络部署提供了有效解决方案。

关键词：卷积神经网络；模型压缩；知识蒸馏；量化；FPGA；边缘计算

Research on Model Compression of Quantized Convolutional Neural Networks Based on Knowledge Distillation

HE Longchao，WU Weikang，LI Bin，CHANG Yinghui
（The 54th Research Institute，China Electronics Technology Group Corporation，Shijiazhuang　050081，China）

Abstract：Aiming at the high resource consumption problem in deploying deep convolutional neural networks on edge devices，a collaborative optimization method combining knowledge distillation and low-bit quantization was studied；The key technology of joint guidance from quantization-aware training and distillation loss was adopted，effectively mitigating the accuracy loss of low-bit quantization through teacher model soft label supervision and projected gradient descent optimization；Experimental tests on CIFAR-10 and CIFAR-100 datasets showed that this method achieved 4-bit quantization of ResNet series networks，reaching 92.1% accuracy on CIFAR-10 with model size compressed to 0.41 MB；Verified through FPGA edge deployment，the ResNet-20 inference latency was reduced from 82.3 ms to 5.67 ms，meeting the engineering requirements for low latency and high efficiency in edge computing；The research confirms that this method can significantly reduce resource overhead while maintaining accuracy，providing an effective solution for neural network deployment in resource-constrained environments.

Keywords：convolutional neural networks；model compression；knowledge distillation；quantization；FPGA；edge computing

0　引言

随着万物互联时代的来临，终端设备的连接规模呈指数级增长，传统云计算架构在实时响应、能源效率及服务性能等多方面难以满足日益增长的需求。因此，人工智能（AI，artificial intelligence）技术与边缘设备相结合的新型计算形式不断出现。在从云计算向边缘计算转变的过程中，硬件算力与人工智能算法的共同发展，为边缘及端侧计算能力的崛起奠定了坚实基础。

卷积神经网络[1]（CNN，convolutional neural networks）在图像处理、语义分割和目标检测等不同领域取得了大量成就。然而，深度卷积神经网络[2]（DCNN，deep convolutional neural networks）的发展依赖图形处理器[3]（GPU，graphics processing unit）、张量处理器[4]（TPU，tensor processing unit）、以及神经网

收稿日期:2025-11-02；　**修回日期**:2025-11-16。

作者简介:何龙超(2000-),男,硕士研究生。

引用格式:何龙超,武唯康,李　斌,等.基于知识蒸馏的量化卷积神经网络模型压缩研究[J].计算机测量与控制,2026,34(2):227-234.

络处理器[5]（NPU，neural network processing unit）等高性能计算硬件。这种集中式训练方式有着数据传输能耗高、隐私保护弱等缺点，因此，当前研究的重要方向为边缘计算。除此之外，DCNN 的成功建立在计算复杂度与存储开销增加的基础之上。文献［6］中的 VGG-16 神经网络的模型参数量达到 1.38 亿，存储占用超过 500 MB，识别输入单张 224×224 的图像需要进行约 300 亿次浮点运算。如此高昂的计算和存储代价，使其推理过程严重依赖高性能计算平台，极大地限制了 DCNN 在资源受限的边缘设备上的部署能力。为此，大量研究针对算法层面的神经网络轻量化，一系列高效的模型压缩方法提出，目的是降低模型复杂度来实现在边缘侧的高效部署。

研究发现[7]大多数卷积神经网络参数是冗余的，5%的核心参数可以推断出 95%的参数。主流深度模型普遍采用浮点数（float）进行存储和计算，因此，将权重与激活值从高比特精度转换为低比特精度，能够有效地压缩模型规模并降低计算复杂度。将 32 位浮点数量化为 8 位整型，模型大小可压缩至原来的四分之一，能够在基本保持精度的前提下，减少存储和访存开销。虽然研究者[8]提出量化卷积神经网络（QCNNs，quantized convolutional neural networks）方法来量化模型参数，但是在大规模数据集上的性能损失比较严重。文献［9］指出当权重和激活值都量化为 8 bit 时 ResNet-50 模型 ImageNet 数据集上的 Top-1 准确率相较于 float 精度下降了 14.04%。当量化精度继续降低到 4-bit 时，模型性能损失更加严重，ResNet-50 的测试准确率仅为 62.11%，与浮点模型存在 14.04%的差距。

将深度模型部署在边缘设备仍面临诸多挑战。首先，现有模型普遍具有参数量庞大、计算复杂度高的特点，导致其难以在存储、算力受限的边缘设备中上直接进行部署。尽管量化等方法可以将浮点量化成整型，从而降低参数规模程度，但是低比特量化可以保持硬件友好性的同时，往往会引发模型性能严重下降。因此，如何在保持模型精度的同时，并实现更高的压缩率，在模型部署过程中首先需要考虑的问题。其次，在边缘设备上进行推理的过程中存在大量的冗余计算。诸如剪枝等方法需要从大量的参数架构中识别到可精简结构，面对动态输入时其稳定性也面临严峻的挑战。因此，如何系统的降低模型复杂度、实现推理加速，是优化模型时延的需要解决的关键问题。

针对上述问题，本文采用量化与知识蒸馏协同优化的技术路径：知识蒸馏提供的软监督信号能够有效地缓解量化过程中的信息损失，并且量化引入的结构化约束可以进一步加强知识蒸馏后模型的硬件友好性。该方法实现了卷积神经网络模型的 4-bit 量化，将优化后的模型部署在 FPGA 端侧设备上，并能够在保持模型精度的同时，ResNet-20 网络的推理时延从 82.3 ms 降低至 5.67 ms，证明了蒸馏与量化协同的可行性，为卷积神经网络在嵌入式场景进行部署提供了可行的技术方案。

1 卷积神经网络结构及原理

1.1 卷积神经网络基本结构

卷积神经网络是具有局部连接和权值共享特点的前馈神经网络，其基本结构主要由卷积层、激活函数、池化层和全连接层构成。这些被分成模块化单元，通过多层堆叠形成完整的卷积神经网络模型。如图 1 所示，在一个典型的卷积神经网络架构中，卷积层对输入的图像执行卷积操作，提取出它的空间特征，池化层对特征图进行下采样，减少特征维度，实现降低计算复杂度和增强特征的鲁棒性。最终，全连接层将高级语义特征映射至目标类别空间，并输出分类结果。

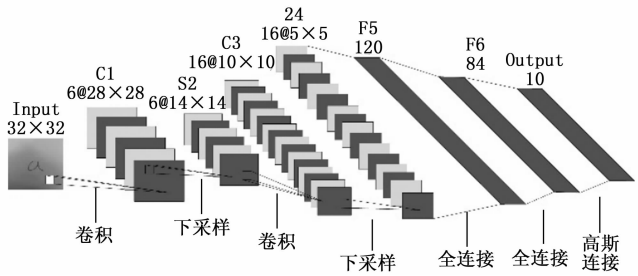

图 1 卷积神经网络基本结构（LeNet-5 为例）

1.2 卷积神经网络的训练与推理

在卷积神经网络中，每个神经元都有输入维度相对应的权重向量，这些参数在训练过程中通过优化算法不断调整。卷积神经网络的训练过程包含前向传播与反向传播两个关键阶段。在前向传播过程中，输入数据从网络底层向顶层逐层传递，依次经过卷积层、池化层和全连接层的处理，最终生成网络输出。反向传播阶段则沿相反路径传递误差梯度，通过计算损失函数对各层权重的偏导数，并利用优化算法更新网络参数，从而持续提升模型性能。如图 2 所示，CNN 的训练流程可分为以下步骤：

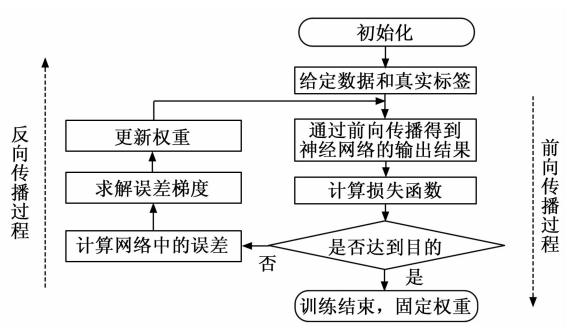

图 2 卷积神经网络训练流程示意图

1）网络初始化；

2）前向传播：在前向传播过程中，训练样本输入至网络中，数据依次通过卷积层、池化层和全连接层等进行逐层计算，最终得到网络输出结果：

$$O^0 = x \tag{1}$$

$$O^l = \sigma(W^l O^{l-1} + b^l) \tag{2}$$

$$y_{\text{output}} = O^L \tag{3}$$

其中：x 为输入数据，O^l 表示网络第 l 层的输出，σ 表示激活函数，W^l 和 b^l 分别代表第 l 层的权重和偏置，L 为神经网络的层数，y_{output} 代表网路的输出结果；

3）计算损失函数：通过损失函数求出网络输出结果与真实标签之间的误差：

$$\mathcal{L} = Loss(y_{\text{output}}, y_{\text{label}}) \tag{4}$$

其中：y_{label} 为数据的真实标签。

4）误差传播：将输出层的误差沿网络反向传播，基于链式法则逐层计算各神经元的梯度；

5）权重更新：基于反向传播求得的梯度，采用优化算法对网络权重参数进行迭代更新。优化算法包括随机梯度下降[10]（SGD，stochastic gradient descent）、动量[11]（Momentum）以及 Adam 优化器等。通过对参数不断地调整，使网络损失函数不断降低。

6）重复训练：重复执行 2）～4）步，直至模型在验证集上表现稳定，达到需要的标准。

卷积神经网络的推理过程是在模型训练完成后展开的。在图像分类任务中，经过大规模数据训练的 CNN 能够从输入图像到类别标签的映射；在部署阶段过程中，系统接收真实场景的图像输入，通过前向传播输出分类结果。该过程只涉及数据从输入层到输出层的单向流动，因而具有更高的计算效率和更低的资源开销。

2　蒸馏量化的方案

2.1　量化原理分析

根据实现阶段的不同，卷积神经网络模型的量化方法包括两种：训练后量化[12]（PTQ，post-training quantization）与量化感知训练[13]（QAT，quantization aware training）。研究发现，量化感知训练能够让模型在训练过程中适应量化带来的精度损失，从而更好地保持模型准确率。在具体量化策略选择方面，二值权重网络虽可将参数量化为－1 和＋1，高达 32 倍的压缩比，但会引发显著的精度下降。为此，本文选择将模型参数量化为 INT4 精度，在实现模型压缩的同时，有效维持了模型的推理精度。

1）激活量化器：当前大多数卷积神经网络采用 ReLU 作为激活函数，然而当使用传统量化方法处理激活值时，会出现明显的量化误差。因此，研究者提出了截断式 ReLU 激活函数，通过引入可训练阈值参数 α，将量化后的激活值限制在特定动态范围内。激活的 k-bit 量化过程可表示为以下形式：

$$\hat{x} = 0.5(|x| - |x-\alpha| + \alpha) = \begin{cases} 0, & x \in (-\infty, 0) \\ x, & x \in [0, \alpha) \\ \alpha, & x \in [\alpha, +\infty) \end{cases} \tag{5}$$

$$Q_A(x) = \left\lfloor \hat{x} \cdot \frac{2^k - 1}{\alpha} \right\rfloor \frac{\alpha}{2^k - 1} \tag{6}$$

在量化过程中，参数 α 作为可学习的缩放因子，通过训练动态调整以将激活值范围约束在 $[0, \alpha]$ 区间。符号 $\lfloor \cdot \rfloor$ 表示向下取整运算。以 4 比特量化为例，激活值将被量化为 16 个均匀分布的离散值，其量化结果可表示为：

$$Q_A(x) \in \left\{0, \frac{a}{15}, \frac{2a}{15}, \frac{3a}{15}, \cdots, \frac{14a}{15}, a\right\} \tag{7}$$

参数 α 的梯度更新通过直通估计器[14]（STE，straight through estimator）方法实现，该方法在反向传播过程中绕过量化操作的导数不连续性，实现有效的梯度计算与参数优化：

$$\frac{\partial L}{\partial \alpha} = \frac{\partial L}{\partial Q_A(x)} \frac{\partial Q_A(x)}{\partial \alpha} = \begin{cases} \dfrac{\partial L}{\partial Q_A(x)}, & x > \alpha \\ 0, & \text{else} \end{cases} \tag{8}$$

根据上述公式推导可知，当所有激活输出值均小于参数 α 时，其梯度计算结果为零。当检测到所有激活值均低于当前 α 阈值时，需通过引入 L2 正则化约束来持续缩小 α 的数值范围，从而确保训练过程的有效推进与参数的持续优化。

2）权重量化器：权重量化器与激活量化器相似，采用 k 比特量化方案，其具体定义如下：

$$\hat{w} = 0.5(|w + \alpha_w| - |w - \alpha_w|) \tag{9}$$

$$\hat{w}' = \frac{\hat{w}}{2\alpha_w} + 0.5 \tag{10}$$

$$Q_W(w)' = \frac{\lfloor \hat{w}' \cdot (2^k - 1) \rfloor}{2^k - 1} \tag{11}$$

$$Q_W(w) = 2\alpha_w(Q_W(w)' - 0.5) \tag{12}$$

其中：ω 表示网络权重参数，α_ω 为权重的截断变量，α_ω 通过以下方式进行梯度更新：

$$\frac{\partial L}{\partial \alpha_w} = \begin{cases} \dfrac{\partial L}{\partial Q_W(w)}, & x > \alpha \\ -\dfrac{\partial L}{\partial Q_W(w)}, & x < -\alpha \\ 0, & \text{else} \end{cases} \tag{13}$$

2.2　知识蒸馏

通常情况下，具有更深层结构和全精度参数的教师网络相较于学生网络展现出更优异的性能，能够实现更为复杂的任务目标。然而，这类网络通常具有的参数量大与计算复杂度高，这些特点导致其难以在 FPGA 等

硬件平台部署至小型化、资源受限的边缘设备。相比之下，结构精简的轻量化网络虽具备参数量少、计算效率高的优势，更符合硬件部署要求，但其固有性能往往难以直接满足实际应用需求，通常需要经过进一步的优化调整才能达到预期效果。

模型训练过程中以优化训练集上的性能指标为直接目标，然而模型的根本价值在于对未知数据表现出良好的泛化能力。尽管直接优化泛化性能在理论上更为合理，但由于泛化机制的内在复杂性以及缺乏对最优泛化形式的先验知识，这一目标往往难以直接实现。

在知识蒸馏框架中，通过将大型教师模型所蕴含的知识迁移至小型学生模型，可有效引导学生模型模仿教师模型的泛化特性。当教师模型本身具有卓越的泛化能力时，经过蒸馏训练的学生模型在测试集上的表现通常显著优于仅在原始训练集上常规训练的同类模型。将教师模型泛化能力转移至学生模型的关键技术在于利用其输出的类别概率分布作为" 软标签" 来监督训练过程。这一迁移阶段可使用原始训练集或专设的迁移数据集完成。当教师模型由多个基础模型集成构成时，可将各成员预测分布的算术平均或几何平均作为软标签源。相较于传统的 one-hot 标签，高熵值的软标签不仅包含了更丰富的类别间关联信息，还能在不同样本间提供更为平缓的梯度变化，这使得学生模型通常能够在显著减少训练数据量的情况下，以更高的学习率实现高效收敛。

本文采用的知识蒸馏方法如下：通过调整 softmax 函数的温度参数 T，生成蕴含丰富类别关联信息的教师软标签；在训练学生模型时保持相同温度设置，以对齐其输出分布与软标签，其基本框架如图 3 所示。实验表明，结合原始训练数据与复合目标函数可获得最优性能。该函数同时包含两项监督：一项指导学生模型拟合真实标签，另一项促使其逼近教师模型产生的软标签分布。值得注意的是，学生模型通常难以完全复现软标签分布，而其在正确类别方向上存在的预测偏差，有助于提升模型的泛化能力。

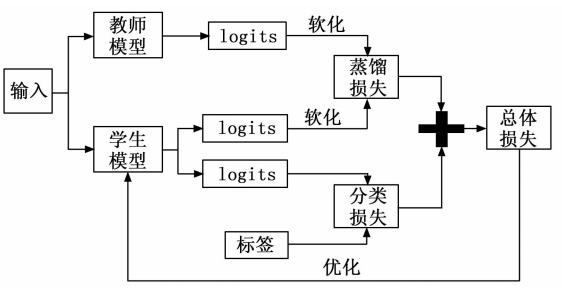

图 3 知识蒸馏流程图

在神经网络中，类别概率通常由 Softmax 输出层生成，该层将各类别的逻辑值 z_i 转换为对应的概率值 q_i，转换过程通过将 z_i 与其他类别的逻辑值进行比较而实现：

$$q_i = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}} \tag{14}$$

其中：T 为知识蒸馏中的温度系数。当温度 $T=1$ 时，模型输出的分类概率分布称为“硬目标”（hard targets）。随着温度 T 升高，softmax 函数输出的概率分布趋于平滑，模型在训练过程中更加关注非正确类别间的关系信息，此时得到的输出称为“软目标”（soft targets）。

在知识蒸馏的基本形式中，通过将教师模型在较高温度下生成的软目标分布作为训练目标，将各类别概率分布平滑，从而可以使那些非正确类别所携带的相对信息凸显出来，让学生模型进行学习。将知识传递至学生模型。训练阶段使用相同温度 $T=3$，训练完成后温度恢复为 1。由于软目标所产生的梯度幅度按 $1/T^2$ 缩放，在同时使用硬目标与软目标时，需将软目标对应的梯度乘以 T^2，从而确保在不同温度设置下硬目标与软目标的相对贡献保持稳定。

对于迁移集中的每个样本，蒸馏模型在每个逻辑值 z_i 上的交叉熵梯度为 $\partial C/\partial z_i$。设教师模型的逻辑值为 v_i，生成的软目标概率为 p_i，训练温度为 T，则该梯度可表示为：

$$\frac{\partial C}{\partial z_i} = \frac{1}{T}(q_i - p_i) = \frac{1}{T}\left(\frac{e^{z_i/T}}{\sum_j e^{z_j/T}} - \frac{e^{v_i/T}}{\sum_j e^{v_j/T}}\right) \tag{15}$$

当温度 T 远大于逻辑值的数量级时，可近似为：

$$\frac{\partial C}{\partial z_i} \approx \frac{1}{T}\left(\frac{1+z_i/T}{N+\sum_j z_j/T} - \frac{1+v_i/T}{N+\sum_j v_j/T}\right) \tag{16}$$

进一步假设每个迁移样本的逻辑值已进行零均值化处理，即 $\sum_j z_j = \sum_j v_j = 0$，则可简化为：

$$\frac{\partial C}{\partial z_i} \approx \frac{1}{NT^2}(z_i - v_i) \tag{17}$$

因此，在高温条件下，若对每个迁移样本的逻辑值分别进行零均值化处理，蒸馏过程近似于最小化 $1/2(z_i - v_i)^2$。在较低温度下，蒸馏会减弱对远低于平均值的负逻辑值的匹配程度，这些逻辑值在复杂模型训练中受代价函数约束较小，可能包含较多噪声信息。

2.3 蒸馏量化方案

在完成卷积神经网络的知识蒸馏训练后，为在进一步压缩模型规模的同时保持甚至提升其精度表现，本文采用蒸馏损失指导后续的量化训练过程。在量化训练阶段，优化目标同时结合真实标签对应的任务损失与教师模型输出的软目标损失，通过双重监督机制引导学生模型在低比特量化约束下保持更强的泛化能力，从而有效缓解因数值精度降低而导致的性能下降问题。

首先定义一个归一化函数 sc：$R^n \longrightarrow [0, 1]$，该

函数将任意取值范围的输入向量映射为各元素值均位于[0，1]内的归一化向量。基于该归一化函数和量化函数的一般结构可定义如下：

$$Q(\boldsymbol{v}) = sc^{-1}\{\hat{Q}[sc(\boldsymbol{v})]\} \tag{18}$$

其中：sc^{-1}是缩放函数的逆函数，$\hat{Q}$是实际的量化函数，其输入输出值域均限定在[0，1]区间的值，$\boldsymbol{v}$为待量化的原始向量。在实际应用中，网络权重通常为多维张量形式，为便于处理可将其重塑为一维向量进行量化操作，待量化完成后再恢复至原始维度结构。缩放函数具有多种实现形式，本文采用线性缩放策略，即：

$$sc(\boldsymbol{v}) = \frac{\boldsymbol{v}-\beta}{\alpha} \tag{19}$$

其中：$\alpha = max_i v_i - min_i v_i$，计算数据集的极差，$\beta = min_i v_i$，选取数据集中的所有观测值的最小值 $v-\beta$。先将数据进行平移，使其最小值移动到 0，线性缩放函数可将平移后的数据按照极差进行缩放，使得原数据中的最大值恰好映射到 1，从而将原始向量 $\boldsymbol{v}$ 中的所有元素映射在区间内，将上述参数代入缩放函数表达式，可得完整变换形式为：

$$Q(\boldsymbol{v}) = \alpha\hat{Q}\left(\frac{v-\beta}{\alpha}\right)+\beta \tag{20}$$

在蒸馏量化框架中，需依次完成两个关键步骤。首先，通过蒸馏损失函数实现从教师模型到学生模型的知识迁移，指导学生模型的训练过程。其次，在量化神经网络训练阶段，将蒸馏损失有效地整合至优化目标中。采用基于投影梯度下降的优化方法：训练过程首先按照常规方式计算梯度并更新网络参数，随后将更新后的参数投影至预设的量化解空间。该方法的创新点在于：在每次投影操作后，将产生的量化误差累积至后续迭代的梯度计算中，动态评估每个权重参数是否应当调整至相邻的量化电平，从而在保持训练稳定性的同时提升量化模型的精度表现。

在量化神经网络环境下引入蒸馏损失进行模型优化，采用投影梯度下降法实现训练过程。该方法首先在全精度参数空间执行标准梯度更新，随后将更新后的参数投影至预设的离散量化值集合。关键之处在于，每次投影操作产生的量化误差会被累积并反馈至后续迭代的梯度计算中，以此动态评估每个权重参数是否需要调整至相邻量化电平，该训练过程在全精度空间进行随机梯度下降（SGD），而实际梯度计算则基于量化后模型的前向传播结果，并以蒸馏损失作为核心优化目标。

蒸馏量化过程：

设 ω 是网络权重；

loop；

$\omega^q \longleftarrow$ 量化函数 (w,s)；

执行前向传播并计算蒸馏损失 $l(\omega^q)$；

执行反向传播并计算 $\frac{\partial l(\omega^q)}{\partial \omega^q}$；

使用 SGD 以全精度更新原始权重 $\omega = \omega - v \cdot \frac{\partial l(\omega^q)}{\partial \omega^q}$；

最后在返回量化权重前：$\omega^q \longleftarrow$ 量化函数 (w,s)；

returnω^q。

设 $\boldsymbol{p}=(p_1, \cdots, p_s)$ 为量化点的向量，$Q(\boldsymbol{v}, \boldsymbol{p})$ 表示量化函数。理想情况下，能够找到一组量化点 $\boldsymbol{p}$，使得在使用 $Q(\boldsymbol{v}, \boldsymbol{p})$ 对模型进行量化时，精度损失达到最小。为此，通过计算 Q 相对于 $\boldsymbol{p}$ 的梯度，使用随机梯度下降方法，找到这个关键的量化点 $\boldsymbol{p}$，从而确定最优的量化配置。

在量化神经网络的过程中，如何确定用于替换原始权重的具体量化值 p_i。由于量化操作本身是离散的，其对应的梯度几乎处处为零，导致无法直接通过梯度反向传播对量化过程进行端到端优化：

$$\frac{\partial Q(\boldsymbol{v},\boldsymbol{p})}{\partial v} = 0 \tag{21}$$

该问题在量化神经网络中普遍存在，导致无法通过常规反向传播算法将梯度传递至量化函数之前的网络层。为了解决上述问题，本文采用一种改进的直通估计器（STE）变体进行梯度近似。另一方面，模型所学习的量化电平参数 p_i 在优化过程中保持连续可微特性 $Q(\boldsymbol{v}, \boldsymbol{p})_i$，使得量化函数对任意的 p_j 均可导，其梯度计算表达式如下：

$$\frac{\partial Q(\boldsymbol{v},\boldsymbol{p})_i}{\partial p_j} = \begin{cases} \alpha_i, v_i \text{ 量化为 } p_j \\ 0, \text{其他} \end{cases} \tag{22}$$

其中：α_i 表示第 i 个权重对应的缩放因子。

因此，采用与原始模型训练阶段相同的损失函数，并基于式（22）所定义的梯度近似关系，结合标准反向传播算法，计算损失函数对量化点 p 的梯度。在此基础上，利用随机梯度下降（SGD）算法对量化点 p 进行联合优化，以最小化量化引起的精度损失。

3　硬件模块

3.1　整体系统框架

本文使用一个四核张量处理单元（TPU）构建的 FPGA 计算平台，在其上实现部署。如图 4 所示，其整体架构是将 TPU 部署在 VU13P FPGA 上。

在核心处理单元上，两个 TPU 计算核集成在一片 VU13P FPGA 之中。具体而言，VU13P _ 0 内部部署了 TPU0 与 TPU1，而 VU13P _ 1 内部则部署了 TPU2 和 TPU3。上述设计分布式布局，能够为并行处理大规模矩阵乘加等张量运算提供硬件基础。系统的数据交互与控制链路通过部署在 KU3P FPGA 上的 PCIe XDMA（DMA over PCI Express）使用。KU3P 作为主机与计算单元之间的高速桥接器，通过 PCI Express 总线与测

试主机连接。测试主机安装 Ubuntu 18.04 操作系统，通过调用 PCIe 接口驱动程序，来完成对 FPGA 平台上各 TPU 核的直接内存访问（DMA），从而完成测试向量的下发、计算任务的调度、结果的回收以及最终的性能测试验证。

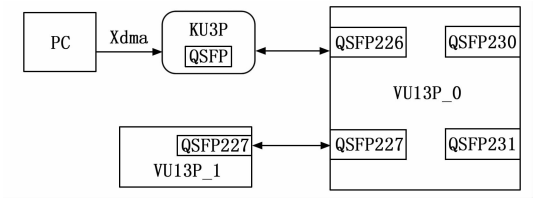

图 4 整体系统框架图

3.2 KU3P 系统框架

如图 5 所示，系统中主机与加速器间通过部署在 KU3P FPGA 上的 PCIe XDMA 实现通信与控制链路。系统构建了一个高效的数据通路，测试主机通过 PCIe 总线发起的数据传输请求，由 XDMA 处理并经链路转发至目标 VU13P FPGA 上的 TPU 计算核心。

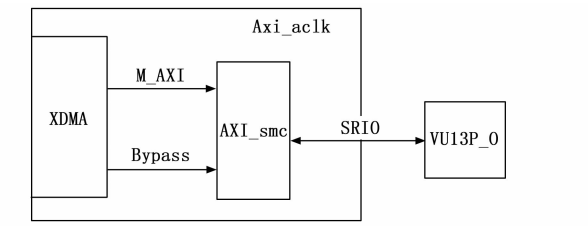

图 5 KU3P 系统框架图

3.3 VU13 P 系统框架

如图 6 所示，VU13 P 系统构建片上网络（NoC，Network-on-Chip）为核心的高速互连架构。片上网络能够为片内多个 TPU 计算核心提供高带宽、低延迟的通信通路。在板级层面，片上网络通过串行 RapidIO（SRIO，serial rapidIo）高速串行协议与 KU3P FPGA 建立连接，从而构成了主机与 TPU 之间稳定的数据交互通道。同时，不同 VU13P 板卡之间的数据通信，也通过板间的 SRIO 链路直接实现。NoC 负责片内计算核间的协同，而 SRIO 则实现了板级设备间的高效数据交换。

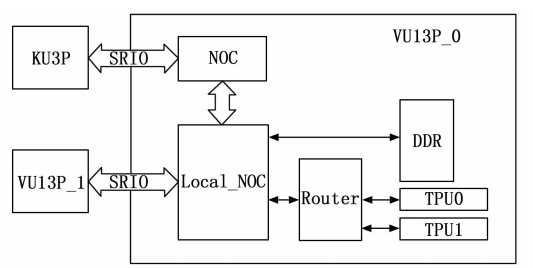

图 6 VU13P 系统框架图

4 实验结果与分析

4.1 实验数据

本文使用的数据集是两种公开的图像数据集，分别为：CIFAR-10 和 CIFAR-100[20]。其中，CIFAR-10 包含 60 000 张 32×32 像素的彩色图片。数据集被规范地划分为训练集（Train）和测试集（Test）两部分，其中训练集 50 000 张，测试集 10 000 张。图片内容涵盖飞机（Airplane）、汽车（Automobile）、鸟类（Bird）、猫（Cat）、鹿（Deer）、狗（Dog）、青蛙（Frog）、马（Horse）、船只（Ship）和卡车（Truck）共计 10 个类别，每个类别拥有 6 000 张图像。这些类别涵盖了常见的动物和交通工具，图像背景相对简洁，目标主体突出，可以很好地用在测试模型上。

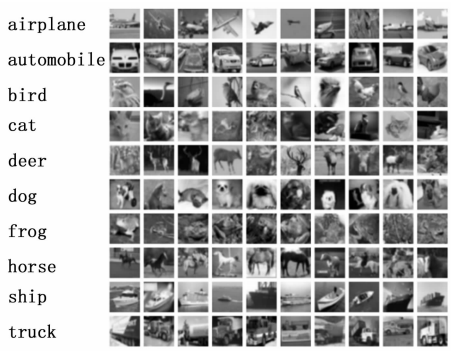

图 7 CIFAR10 数据集图片示例

CIFAR-100 为 CIFAR-10 的扩展版本，包含 60 000 张 32×32 像素的彩色图片，其划分方式与 CIFAR-10 一致，即训练集 50 000 张，测试集 10 000 张。图片包括海狸、毛虫、黑猩猩、枫树、床、蘑菇、橘子、摩托车、键盘、甜椒等共计 100 个不同类别。为了使任务更具有层次性，这 100 个类别又划分成 20 个超类，每个超类中包含 5 个子类。这种双层标签使 CIFAR-100 能够更好地还原真实世界中物体的层次化分类，对模型的细粒度识别能力提出了更高要求。

4.2 实验环境以及训练参数配置

实验环境的硬件配置详见表 1。

表 1 硬件配置

系统	Windows10(64 位)Ubuntu 18.04 Linux
RAM	32 GB
GPU	NVDIA RTX 2090
深度学习框架	PyTorch
GPU 加速库	CUDA 12.2

实验所用训练参数配置详见表 2。

4.3 评价指标

参数减少量和压缩率作为模型压缩效果的评价指标。参数减少量是来衡量参数量减少的绝对值；压缩率

表 2　训练参数配置

初始学习率	0.01
蒸馏温度	3.0
数据批次	256
训练轮数	200
优化器	SGD

则是通过压缩前后模型存储量的比值来评估减少的相对程度。压缩率定义为：

$$\varphi(M,M') = \frac{\mu}{\mu'} \tag{23}$$

其中：μ 表示压缩前模型的存储容量，μ' 表示压缩后模型的存储容量。

在同一个硬件平台上，模型的推理时间为评估其模型加速的指标。加速率定义为压缩前后模型推理时间的比值，具体计算公式如下：

$$\varphi(M,M') = \frac{v}{v'} \tag{24}$$

其中：v 表示压缩前模型在目标平台上的推理时间，v' 表示压缩后模型在目标平台上的推理时间。

4.4　对比实验

本文分别在 CIFAR-10 和 CIFAR-100 数据集上对所提出的方法进行实验，选择了用蒸馏提升量化神经网络的方案作为对比，其中包括：代表性量化蒸馏方法 QDistill[15]、基于大型教师模型的离线量化蒸馏方法 QKD[16]、在线量化蒸馏方法 ONE[17] 以及采用量化指导的 SPEQ[18]，离线蒸馏方法 AFD 及在线蒸馏方法 DML[19]。

表 3 中是不同算法在 CIFAR-10 数据集上基于 ResNet-20 网络的对比结果，在 W4/A4（4-bit 权重和 4-bit 激活）比特精度的量化方法中，Baseline 方法的准确率为 90.2%，本文算法在同等 W4/A4 比特精度条件下，准确率达到了 92.1%，不仅超过了 Baseline 方法 1.9%，也优于其他方法。与 QKD 相比提升 1.3%，和表现最好的 SPEQ 相比还高出 0.5%，与全精度方法对比发现，DML 和 AFD 方法分别达到了 92.7% 和 92.3%的准确率。虽然全精度方法整体表现最优，但是本文算法其性能与最优的全精度方法 DML 仅相差 0.6%，其精度直逼全精度网络的精度。

表 3　不同算法在 ResNet-20 网络上的测试结果对比

算法	比特精度(W/A)	测试准确率/%
Baseline	4/4	90.2
DML	F/F	92.7
AFD	F/F	92.3
QDistill	4/4	89.9
QKD	4/4	90.8
SPEQ	4/4	91.6
本文算法	**4/4**	**92.1**

表 4 表示了不同算法在 CIFAR-100 数据集上在 ResNet-50 网络上的对比结果。本文算法在 W4/A4 比特精度表现出良好的性能优势。Baseline 方法的准确率为 66.3%，本文算法在同等 W4/A4 比特精度条件下，准确率达到了 69.4%，与 Baseline 相比，本文算法提升了 3.1%；与 QKD 相比提升 2.6%；是与表现较好的 SPEQ 方法相比，本文算法仍高出 0.8%。本文算法的性能与全精度方法 AFD 仅相差 0.5%。

表 4　不同算法在 ResNet-50 网络上的测试结果对比

测试准确率/%	算法	比特精度(W/A)
Baseline	4/4	66.3
DML	F/F	70.3
AFD	F/F	69.9
QDistill	4/4	63.3
QKD	4/4	66.8
SPEQ	4/4	68.6
本文算法	**4/4**	**69.4**

因此，本文算法对于更加复杂的数据集和分类任务，能够在保持低比特精度的同时，实现接近全精度的性能表现。

4.5　端侧推理时延对比

根据整体系统架构搭建硬件测试平台，KU3P FPGA 通过金手指供电并插入主板的 PCIe 插槽，其 QSFP 接口与 VU13P _ 0 的 U13 QSFP 端口相连，而 VU13P _ 0 的 U19 QSFP 端口连接至 VU13P _ 1 的 U13 QSFP 接口。系统上电顺序依次为：PC 主机 → VU13P _ 0 → VU13P _ 1，若 SRIO 链路连接正常，对应 LED 指示灯将亮起。通过仿真软件对 FPGA 开发板进行 Flash 烧录，完成量化模型的端侧部署，最终实现对本文算法在端侧平台上针对压缩前和压缩后 ResNet-20 模型进行实际测试验证，将卷积层输出通道并行展开计算[20-22]。

在 CIFAR-10 数据集上，压缩前模型测试准确率为 92.7%，模型存储空间大小为 1.59 MB，功耗为 4.59 W。经过本文算法的 4-bit 量化压缩后，模型准确率下降到 91.8%，保持了优异的识别性能，而模型大小则降低至 0.41 MB，功耗为 2.46W。本方法实现了 3.88 倍的实际压缩率。这就意味着可以在几乎不损失模型精度的情况下，模型的存储空间占用减少了约 74%，能够显著降低了模型部署的存储成本。实验结果显示，量化后模型的单张图片推理时间从 82.3 ms 降低到 5.67 ms，对实时应用场景具有重要意义，也为在资源受限的边缘设备上部署更复杂的模型成为可能。

表 5　蒸馏量化各阶段模型性能和推理时延

	测试准确率/%	模型大小/MB	时延/ms	功耗/W
压缩前	92.7	1.59	82.3	4.59
压缩后	91.8	0.41	5.67	2.46

5 结束语

针对如何在边缘设备部署的卷积神经网络轻量化问题，本文提出了一种融合知识蒸馏与低比特量化的协同优化方法。该方法通过引入知识蒸馏损失来指导量化训练过程，能够有效地缓解低比特量化带来的性能损失。在 CIFAR-10 与 CIFAR-100 数据集上进行了 ResNet 系列网络的 4 位量化实验，其模型精度接近于全精度的测试准确率，但本文所提出的方法显著压缩了模型大小和其推理时延，验证了在 FPGA 平台部署其可行性。本文方法在精度保持与推理加速方面取得良好效果，仍存在需要完善的地方，当前方法对极低比特（如 2-bit）量化的适应性仍需进一步研究。随着边缘计算需求的持续增加，本方法在工业视觉、自动驾驶、物联网终端等实时性要求高的场景中具有广阔应用前景。

参考文献：

[1] LECUN Y，BOSER B，DENKER J，et al. Handwritten digit recognition with a back-propagation network [C] //Proceedings of the International Conference on Neural Information Processing Systems，Cambridge，MA，USA：MIT Press，1989：396－404.

[2] SIMONYAN K，ZISSERMAN A. Very deep convolutional networks for large-scale image recognition [C] //Proc. of the 3rd International Conference on Learning Representations. OpenReview. net，2015：1－14.

[3] JIANG Y，WANG S，VALLS V，et al. Model pruning enables efficient federated learning on edge devices [J]. IEEE Transactions on Neural Networks and Learning Systems，2023，34 (12)：10374－10386.

[4] SHIN J H，SHAFIEE A，PEDRAM A，et al. Griffin：Rethinking sparse optimization for deep learning architectures [C] //2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA). IEEE，2022：861－875.

[5] 王东炜，刘柏辰，韩 志，等. 基于低秩分解和向量量化的深度网络压缩方法 [J]. 计算机应用，2024，44 (7)：1987－1994.

[6] ZENG S，LIU J. FlightLLM：efficient large language model inference with a complete mapping flow on FPGAs [C] //Proceedings of the 2024 ACM/SIGDA International Symposium on Field Programmable Gate Arrays，2024：223－234.

[7] RAO，CHEN J. SparseCore：stream ISA and processor specialization for sparse computation [C] //Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems，2022：186－199.

[8] PELTEKIS C，TITOPOULOS V. DeMM：a decoupled matrix multiplication engine supporting relaxed structured sparsity [J]. IEEE Computer Architecture Letters，2024，23 (1)：17－20.

[9] HE Y，LIN J. Amc：Automl for model compression and acceleration on mobile devices [C] //Proceedings of the European Conference on Computer Vision (ECCV)，2018：784－800.

[10] DENG X，ZHANG Z. Sparsity-control ternary weight networks [J]. Neural Networks，2022，145：221－232.

[11] 刘百成. 二值卷积神经网络加速器的 VLSI 架构设计 [D]. 合肥：中国科学技术大学，2020.

[12] 李喜林. 基于 FPGA 的神经网络二值化及其推理部署优化研究 [D]. 西安：西安电子科技大学，2021.

[13] 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.

[14] 詹宏毅. 基于 FPGA 的深度可分离卷积神经网络加速器设计研究 [D]. 成都：电子科技大学，2021.

[15] MIRZADEH S I，FARAJTABAR M，et al. Improved knowledge distillation via teacher assistant [C] //Proceedings of the AAAI Conference on Artificial Intelligence，2020，34 (4)：5191－5198.

[16] KIM J，BHALGAT Y. Qkd：quantization-aware knowledge distillation [J]. Arxiv Preprint，2019 ArXiv：1911.12491.

[17] ZHU X，GONG S. Knowledge distillation by on-the-fly native ensemble [J]. Advances in Neural Information Processing Systems，2018：7528－7538.

[18] BOO Y，SHINS S. Stochastic precision ensemble：self-knowledge distillation for quantized deep neural networks [C] //Proceedings of the AAAI Conference on Artificial Intelligence，2021，35 (8)：6794－6802.

[19] ZHANG Y，XIANG T. Deep mutual learning [C] //Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition，2018：4320－4328.

[20] HINTON G E. Learning multiple layers of representation [J]. Trends in Cognitive Sciences，2007，11 (10)：428－434.

[21] 莫梓嘉. 面向边缘智能的深度卷积神经网络训练与推理优化方法研究 [D]. 北京：北京邮电大学，2023.

[22] LECUN Y，KAVUKCUOGLU K，FARABET C. Convolutional networks and applications in vision [C] //Proceedings of 2010 IEEE International Symposium on Circuits and Systems，IEEE，2010：253－256.