面向多功能张量加速器的细粒度结构化稀疏设计

赵桦筝[1]，庞善民[1]，赵英海[2]，华高晖[1]，李晨阳[1]，段战胜[3]，梅魁志[4]
（1. 西安交通大学软件学院，710049，西安；2. 北京华航无线电测量研究所，100013，北京；
3. 西安交通大学自动化科学与工程学院，710049，西安；4. 西安交通大学人工智能学院，710049，西安）

摘要：为解决模型压缩算法与多功能张量加速器（VTA）的适配性问题，通过改进经典的YOLObile分块剪枝方法，完成面向该加速器的自适应细粒度结构化稀疏设计及性能评估。针对VTA的多重循环维度展开特性，对模型的权重张量进行32×32大小的分块；结合时间维度的自蒸馏与空间维度的教师蒸馏，进行多维度特征对齐；通过一阶段式迭代训练方式，改进原有的ADMM算法计算流程，在提升模型部署精度的同时减少训练成本；提出自适应层剪枝率模块，进行总剪枝率的自适应分配，实现端到端的自动化剪枝。实验结果表明：改进方法有效减少了约2.4%的浮点计算量，并在图像分类、目标检测等多项任务中提升了压缩模型的精度，最大增长百分比为2.6%。该方法为深度学习模型在VTA上的稀疏化部署提供了一种高效、轻量级的软件解决方案。

关键词：神经网络轻量化；模型稀疏化；深度学习；多功能张量加速器；模型部署
中图分类号：TP31 **文献标志码**：A
DOI：10.7652/xjtuxb202411017 **文章编号**：0253-987X(2024)11-0176-09

Fine-Grained Structured Sparse Design for Versatile Tensor Accelerator

ZHAO Huazheng[1], PANG Shanmin[1], ZHAO Yinghai[2], HUA Gaohui[1],
LI Chenyang[1], DUAN Zhansheng[3], MEI Kuizhi[4]
(1. School of Software Engineering, Xi'an Jiaotong University, Xi'an 710049, China; 2. Beijing Huahang Institute of Radio Measurement, Beijing 100013, China; 3. School of Automation Science and Engineering, Xi'an Jiaotong University, Xi'an 710049, China; 4. College of Artificial Intelligence, Xi'an Jiaotong University, Xi'an 710049, China)

Abstract: In order to address the compatibility issue between model compression algorithms and the versatile tensor accelerator (VTA), an adaptive fine-grained structured sparse design tailored for this accelerator is proposed by enhancing the classical YOLObile block-wise pruning method and evaluates its performance. In light of the multi-dimensional loop unfolding characteristics of VTA, the model's weight tensors are divided into 32×32 blocks. This approach integrates temporal distillation and spatial distillation to align multidimensional features. Through a single-stage iterative training method, the calculation process of the original ADMM algorithm is refined to improve model deployment accuracy while reducing training costs. An adaptive layer pruning rate module is introduced to dynamically allocate the total pruning rate, facilitating end-to-end automated pruning. The experimental results demonstrate that this improved method effectively reduces floating-point computations by approximately 2.4% and enhances the accuracy of compressed models across various tasks such as image classification and object detection, with a maximum

收稿日期：2024-04-13。 作者简介：赵桦筝（1999—），女，硕士生；梅魁志（通信作者），男，教授，博士生导师。 基金项目：新疆维吾尔自治区重点研发计划资助项目（2022B01008-1）；国家自然科学基金资助项目（62076193）。
网络出版时间：2024-07-12 网络出版地址：https://link.cnki.net/urlid/61.1069.T.20240709.1505.004

growth percentage of 2.6%. This method offers an efficient and lightweight software solution for the sparse deployment of deep learning models on VTAs.

Keywords: neural network compression; model sparsity; deep learning; versatile tensor accelerator; model deployment

随着深度学习的发展，神经网络的复杂度及其对硬件计算资源的需求量与日俱增。现如今，参数量在几十亿甚至万亿级别的大模型层出不穷，在算法功能愈发完善的同时，也向硬件算力和存储空间提出了新的挑战[1]。深度学习模型压缩部署作为深度学习领域研究的重点之一[2]，其目标是解决模型的工程应用问题[3]。在软件算法方面，快速、轻量级的嵌入式端深度学习模型部署[4]，要求提供高效、准确且适用于硬件设备的神经网络推理方案[5]。

目前，通用的硬件架构和复杂的软件编译工具链已经成为深度学习模型部署领域的共识[6-7]。深度学习编译器框架(TVM)[8]作为深度学习编译器的代表，是目前应用最广泛的开源软硬件协同加速引擎。作为 TVM 框架的扩展，多功能张量加速器(VTA)[9]是一个开放、通用、可定制的深度学习加速器，具有完整且适用于 TVM 的编译器堆栈。TVM 和 VTA 共同构成了端到端的软硬件协同加速堆栈解决方案[10]，但当前较少见到针对该方案讨论稀疏化的方法。

剪枝技术通过删除模型中次重要的神经元降低模型的参数量[11]。Nvidia 提出的 2∶4 细粒度结构化稀疏性方案[12]只能针对 Nvidia 芯片完成 50%的稀疏任务。SR-STE 算法[13]提出的 $N∶M$ 结构化稀疏神经网络训练方式仍存在较大精度损失。本文提出的面向张量处理器的细粒度结构化稀疏设计方法，同时兼顾了非结构化细粒度剪枝和结构化粗粒度剪枝方案的优势。

1　研究方案

为了在获得较高剪枝结构自由度的同时，使剪枝后的模型结构能够较好地利用硬件资源实现并行计算[14-15]，本文对剪枝算法 YOLObile[16]进行了改进，即将神经网络的权重矩阵按照 $m\times n$ 维度进行分块，并针对不同的剪枝块设计不同的剪枝规则，同时保证同一剪枝块内的剪枝规则相同。YOLObile 算法通过实验得出最佳的分块方案为 8×4 大小[16]，将该方法直接应用在 VTA 张量加速器的硬件加速设计中时，会导致精度损失较大。为了提升粗粒度分块造成的精度损失，本文设计了如图 1 所示的结构化稀疏设计方法，充分保证了剪枝后模型的准确率和较高的运算速度，并将任务领域由目标检测任务扩展至分类任务。

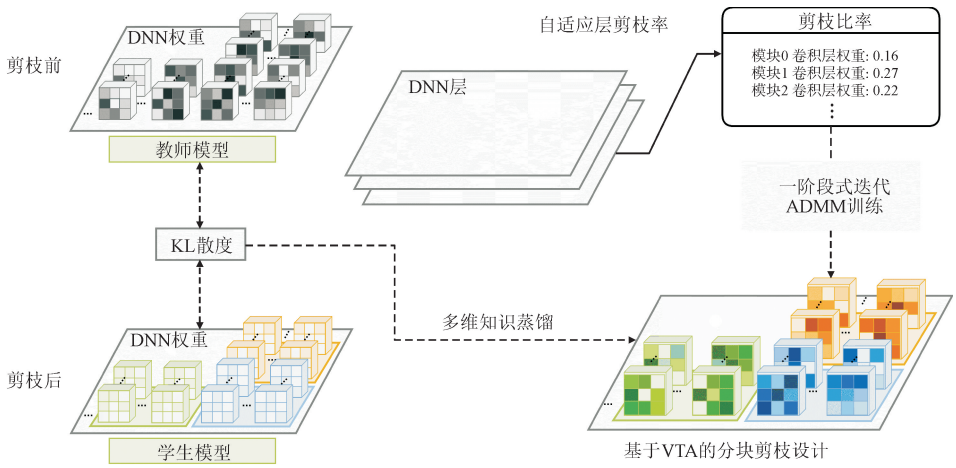

图 1　细粒度结构化稀疏设计方案框架图

Fig. 1　Fine-grained structured sparse design frame

1.1　研究动机

本文首先根据硬件算力对分块粒度大小进行计算。针对 VTA 张量加速器架构，计算模块的计算资源受限，典型配置为 32×32 大小的计算矩阵，也就是最大支持到单次计算 32×32＝1 024 个数据。本文的细粒度结构化稀疏方法保证了每个块内剪枝

规则的一致性，为了降低模型压缩后的精度损失，各块之间使用不同的剪枝规则。本文方法不仅可以应用在卷积核尺寸为 3×3、1×1、5×5 等的卷积层上，也同样适用于线性层。

如图 2 所示，本文以 YOLOv3 模型[17]的 Conv2d.1 张量为例，展示了稀疏化权重数值分布。原始的 Conv2d.1 张量大小为[64，32，3，3]，按照 32×32 大小进行分块后，可得两个大小分别为[32，32，3，3]的子张量。图中灰色阴影方块所代表的数值为 0 的部分可以通过循环指令跳过进行硬件端的加速计算。

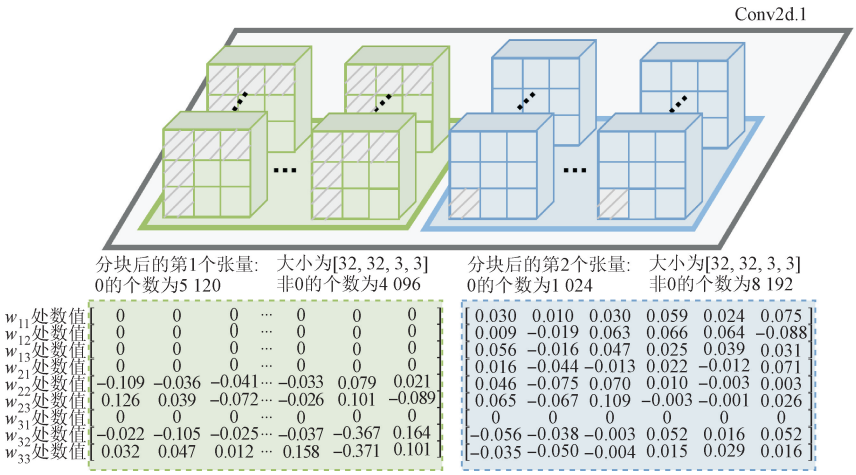

图 2 YOLOv3 模型的稀疏化结果的可视化展示

Fig. 2 Visualization of sparse result in YOLOv3 model

1.2 分块稀疏化设计方案

为了改进张量处理器在 32×32 分块粒度配置下的稀疏模型精度下降问题，首先设计了 1.2.1 小节中的多维知识蒸馏模块和 1.2.2 小节中的一阶段式迭代 ADMM 训练方法提升稀疏模型精度。然后，通过 1.2.3小节中的自适应层剪枝率模块自动化分配各层剪枝率。最后，通过自动权重分配策略平衡各部分损失。

1.2.1 多维知识蒸馏模块

知识蒸馏通常是指使用高精度的教师模型指导低精度的学生模型[18]，在结构相似的情况下效果尤为明显，而剪枝得到的稀疏模型与原模型在结构上高度相似，非常符合知识蒸馏的应用条件。为了同时学习空间维度上未剪枝教师模型的输出及时间维度上前一个训练批次该模型本身的输出，本文设计了多维知识蒸馏模块，如图 3 所示。图中 s 代表学生模型，t 代表教师模型。该模块可分为空间维度的教师模型蒸馏损失$\mathcal{L}_{\mathrm{dis}}$以及时序维度的自蒸馏损失$\mathcal{L}_{\mathrm{dis}}^{\mathrm{self}}$两部分。相较于前者，后者在非目标类别上的概率信息更为丰富。

本文提出的自蒸馏方法按照批次 b 将训练阶段的时间维度划分为b_i 和b_{i-1} 两个小批次，并在第 i 个批次计算二者的软输出值 p_b^i 和 p_b^{i-1} 的 KL 散度，具体计算过程如下

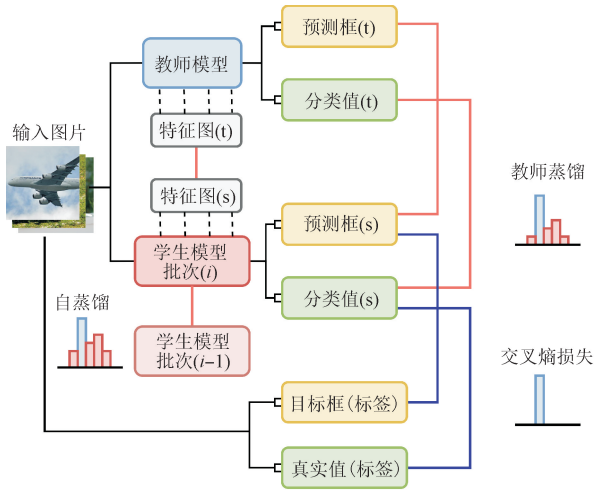

图 3 多维知识蒸馏模型示意图

Fig. 3 Diagram of multi-dimensional knowledge distillation model

$$\begin{cases}\mathcal{L}_{\mathrm{dis}}^{\mathrm{self}} = T^2 f(P,Q)\\ P = \dfrac{1}{S_{i,j}}\left(\exp\left(\dfrac{1}{T}p_b^i\right)\right)\\ Q = \dfrac{1}{S_{i-1,j}}\left(\exp\left(\dfrac{1}{T}p_b^{i-1}\right)\right)\end{cases} \tag{1}$$

式中：f 为 KL 散度函数；i 为批次数；j 为单个批次的矩阵数；p 为模型的输出，在 YOLO 系列模型中即为 YOLOLayer层的输出；$S_{i,j}=\sum_{i=1}^{m}\exp(p_{j,r}^i)$；$S_{i-1,j}=$

$\sum_{i=1}^{m}\exp(p_{j,r}^{i-1})$，其中 m 为模型输出张量单个矩阵的行数；T 为温度，主要起到平滑正则化作用。

教师蒸馏又可进一步划分为类别损失 $\mathcal{L}_{\text{dis}}^{\text{cls}}$、坐标损失 $\mathcal{L}_{\text{dis}}^{\text{box}}$ 以及特征损失 $\mathcal{L}_{\text{dis}}^{\text{feat}}$ 这 3 部分。分类损失首先计算得到样本属于每个类别的概率，并通过 KL 散度分别度量原模型与稀疏模型分类类别分布之间的差异性；坐标损失通过计算预测框中心坐标及锚框宽高，衡量稀疏模型预测的边界框与真实边界框之间的差异；特征损失通过对剪枝前后模型的中间层的特征图进行提取并蒸馏，使稀疏模型能够更好地保留原模型中的细节特征。综上，可得最终的多维知识蒸馏损失 $\mathcal{L}_{\text{dis}}$，算式如下

$$\mathcal{L}_{\text{dis}}=\lambda_{\text{self}}\mathcal{L}_{\text{dis}}^{\text{self}}+(\lambda_{\text{cls}}\mathcal{L}_{\text{dis}}^{\text{cls}}+\lambda_{\text{box}}\mathcal{L}_{\text{dis}}^{\text{box}}+\lambda_{\text{feat}}\mathcal{L}_{\text{dis}}^{\text{feat}}) \tag{2}$$

式中：λ_{self}、λ_{cls}、λ_{box}、λ_{feat} 分别为自蒸馏损失项、分类损失项、坐标损失项、特征损失项的平衡系数。

1.2.2　一阶段式迭代 ADMM 训练

传统的分块剪枝算法[16]是先得到剪枝掩码再剪枝并训练的两阶段式方案。为了节省训练计算成本，避免重复训练剪枝前的预训练模型，以及针对稀疏模型的重新微调训练，本文尝试采用 SR-STE[13]算法进行修改，将传统的两阶段式剪枝方式改为一阶段式剪枝方式，即在剪枝得到稀疏权重和剪枝掩码后，直接通过反向传播更新近似梯度来训练稀疏网络，一步到位，不再进行剪枝后的微调。

通过 2.1.2 小节中的消融实验发现，引入 SR-STE 一阶段式剪枝训练后，模型产生了较大的精度损失，为此本文对一阶段式剪枝进行了改进，同时调整了惩罚参数 ρ，以便更好地应用到模型剪枝领域，并得到了较好的实验结果。

ADMM 算法中，通过引入增广拉格朗日量来求解原始的具有非凸约束的优化问题[19]。如式(3)所示，引入分段函数 $g_i(W_i)$，以不含独立约束条件的形式求解原来的非凸优化问题，表达式为

$$\begin{cases}\min\limits_{\{W_i\}\{b_i\}}\ \mathcal{F}(W_i,b_i)+\sum\limits_{i=1}^{N}g_i(W_i)\\ \text{s.t.}\ \ g_i(W_i)=\begin{cases}0, & \text{card}(W_i)\leqslant l_i\\ +\infty, & \text{其他}\end{cases}\end{cases} \tag{3}$$

式中：$\mathcal{F}(\cdot)$ 为激活函数；W 为权重参数；b 为偏置参数；$g(\cdot)$ 为引入的分段函数；card$(\cdot)$ 为返回其矩阵参数中非零元素数量；N 为总网络层数；l_i 为第 i 层的层剪枝率。

引入增广拉格朗日量，提取缩放对偶变量，即可得到 ADMM 算法迭代式如下

$$(W_i^{k+1},\ b_i^{k+1})=\arg\min_{\{W_i\},\{b_i\}}L_\rho(W_i,b_i,Z_i^k,U_i^k) \tag{4}$$

$$Z_i^{k+1}=\arg\min_{\{Z_i\}}L_\rho(W_i^{k+1},b_i^{k+1},Z_i,U_i^k) \tag{5}$$

$$U_i^{k+1}=U_i^k+W_i^{k+1}-Z_i^{k+1} \tag{6}$$

式中：$L(\cdot)$ 为拉格朗日函数；Z 为优化问题的目标函数；U 为拉格朗日乘子。

使用对偶上升法的思路，针对 $(W_i^{k+1},\ b_i^{k+1})$、Z_i^{k+1}、U_i^{k+1} 这 3 组变量，分别在固定另外两个变量的条件下，更新其中一个变量，重复迭代循环直到数值稳定，模型也就达到收敛了。其中，对于对偶子问题的求解，采用梯度下降法计算如下

$$\min_{\{W_i\}\{b_i\}}\mathcal{F}(W_i,b_i)+\sum_{i=1}^{N}\frac{\rho_i}{2}\|W_i-Z_i^k+U_I^k\|_F^2 \tag{7}$$

即，ADMM 损失 $\mathcal{L}_{\text{admm}}=\sum\limits_{i=1}^{N}\frac{\rho_i}{2}\|W_i-Z_i^k+U_i^k\|_F^2$。基于此，本文设计了如图 4 所示的一阶段式迭代 ADMM 训练优化流程。

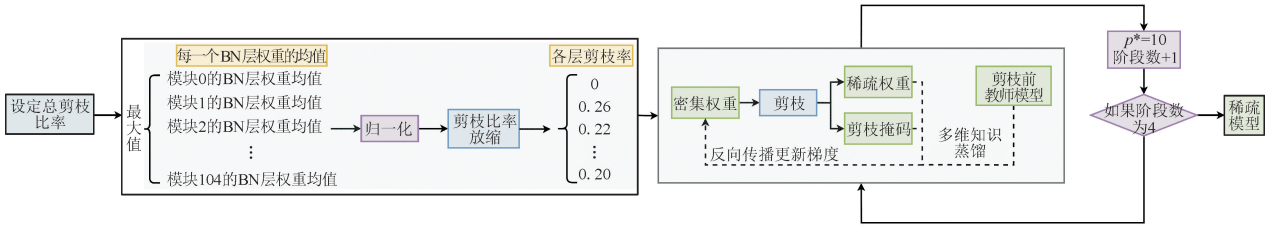

图 4　一阶段式迭代 ADMM 训练流程图

Fig. 4　Flow chart of one-stage iterative ADMM training

在设定了总剪枝率后，通过 1.2.2 小节中提出的自适应层剪枝率模块获得各层的剪枝率 l_i。将总的训练流程分为 4 个连续不断的训练阶段，每一个阶段都使用前一个得到的最终权重作为模型初始化权重，设定惩罚参数 ρ 的初始值为 10^{-4}，ρ 在 $10^{-4}\sim10^{-1}$ 范围内变化，第 i 个阶段的惩罚参数 $\rho_i=10\rho_{i-1}$，通过动态调整惩罚参数，在训练初期尽量使其逼近损失函数 $\mathcal{F}(\cdot)$ 的最小值，在训练后期进一步正则化权重，从而

达到较好的模型精度。在每一个批次的训练中，都使用剪枝后得到的稀疏权重和剪枝掩码计算近似梯度，反向传播更新剪枝前密集权重的梯度。

1.2.3 自适应层剪枝率模块

为了评估度量剪枝率对稀疏模型的性能影响，本文定义对某一层进行单层剪枝后测定的全类平均精度为该层的层精度敏感度。图 5 所示为不同剪枝率下 YOLOv3 层精度敏感度变化趋势图，用不同颜色标记了不同总剪枝率的设定分类，并在层叠图中进行了分层绘制。例如，当剪枝率设定为 0.7 时，测得第 29 层单层剪枝后的全类平均精度指标为0.46，即其精度敏感度为 0.46。

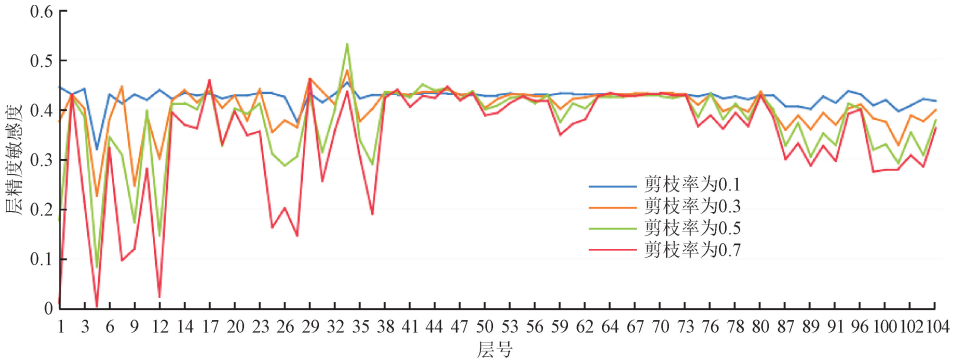

图 5 YOLOv3 层精度敏感度趋势图

Fig. 5 YOLOv3 layer precision sensitivity trend diagram

由图 5 可得，不同的剪枝率下，层精度敏感度变化趋势是一致的，即层敏感度是由模型结构决定的固有属性，不随剪枝率的变化而变化，因此可以通过分析预训练模型结构及权重数值特征，得到恰当的层剪枝率分配策略。基于上述结论，本文进一步提出了自适应的层剪枝率模块。本文采用传统的用于求解非凸约束优化问题的交替向乘子法（ADMM）算法选取最优剪枝权重。为了保证总剪枝率的设定需求，先前的工作大多通过多次调参设置，选取最优的层剪枝比率。

BN 层在卷积层之后进行了数据归一化，BN 层中的 α 参数包含了数据分布信息，可依据 BN 层的权重数据进行每层重要性衡量。基于此，本文提出了如图 6 所示的自适应层剪枝率模块。

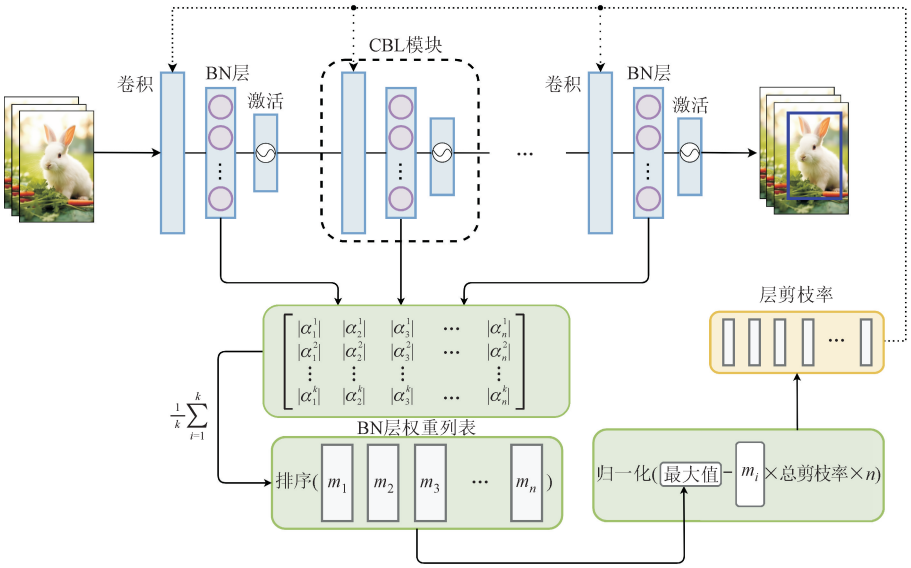

图 6 自适应层剪枝率模块示意图

Fig. 6 Adaptive layer pruning rate module diagram

将每个 BN 层权重的均值作为评估各层重要性的指标，并通过 Softmax 函数将其映射为取值区间在[0，1]范围内且和为 1 的概率分布，然后按照指定的剪枝比率进行放缩，最终得到自适应的各层剪枝率。选层策略主要可分为按照 BN 层权重的最大值排序以及按照均值排序两种方案，本文通过实验

发现按照均值排序效果更优。

为了更好地平衡多维知识蒸馏损失、一阶段式迭代 ADMM 训练损失以及传统训练损失在梯度下降过程中的作用，本文设计了如下所示的自适应损失权重分配策略

$$\begin{cases}\mathcal{L}=\lambda\left(\dfrac{\mathcal{L}_{\text{dis}}}{s}\mathcal{F}+\dfrac{\mathcal{F}}{s}\mathcal{L}_{\text{dis}}+\mathcal{L}_{\text{admm}}\right)\\ s=\mathcal{F}+\mathcal{L}_{\text{dis}}+\mathcal{L}_{\text{admm}}\end{cases}\tag{8}$$

式中：$\mathcal{L}$为总损失；s 为上述三项损失之和；$\mathcal{F}$为传统训练损失；λ 为超参数。

通过实验发现，深度学习模型训练损失函数 $\mathcal{F}$ 与知识蒸馏损失$\mathcal{L}_{\text{dis}}$ 的梯度调整需要相互制约，因此在前面分别乘上了对方的比例系数，而$\mathcal{L}_{\text{admm}}$ 极小，不需要进行比例调整。

2　实验结果与分析

2.1　消融实验

为了探究面向张量处理器 VTA 的细粒度结构化稀疏设计中各个模块的有效性，本节在 COCO128 数据集上，使用 YOLOv3 模型并将总剪枝率设定为 30%，进行了如下消融实验。

2.1.1　多维知识蒸馏模块

表 1 为多维知识蒸馏模块消融实验结果，采用精度、F_1、FLOPS 压缩率衡量本文方法的有效性。针对目标检测任务，由于 8×4 分块粒度较细，加入多维知识蒸馏模块后，全类平均精度指标有所提升但不明显，而对于粒度较粗的 32×32 分块，加入多维知识蒸馏模块后，剪枝后的模型精度由 0.801 提升至 0.822，但仍有较大的优化空间。

表 1　多维知识蒸馏模块消融实验结果

Table 1　Ablation experiment of multi-dimensional knowledge distillation module

方案	分块大小	知识蒸馏	精度	F_1	FLOPS 压缩率
1	未剪枝	×	0.830	0.747	0
2	8×4	×	0.828	0.750	1.417
3	8×4	√	0.829	0.725	1.417
4	32×32	×	0.801	0.700	1.426
5	32×32	√	0.822	0.741	1.426

注：表内“×”表示无该模块，“√”表示包含该模块。

2.1.2　SR-STE 模块

为了减少训练代价，本文采用 SR-STE 算法[13]将传统的两阶段式流程优化为直接进行稀疏训练的一阶段式流程，结果见表 2。

表 2　SR-STE 算法一阶段式剪枝消融实验结果

Table 2　SR-STE one-stage pruning ablation experiment

方案	分块大小	知识蒸馏	SR-STE	精度	F_1	FLOPS 压缩率
1	未剪枝	×	×	0.830	0.747	0
2	8×4	×	×	0.828	0.750	1.417
3	32×32	√	×	0.822	0.741	1.426
4	32×32	√	√	0.766	0.660	1.426

注：表内“×”表示无该模块，“√”表示包含该模块。

由于一阶段式训练轮次的减少，模型产生了较大的精度损失，全类平均精度下降了 6.8%，为此本文将通过后续 2.1.3 小节及 2.1.4 小节中的模块对经典的 SR-STE 算法进行改进。

2.1.3　自适应层剪枝率模块

表 3 所示为自适应层剪枝率模块的消融实验结果，本文通过如下消融实验以数值化形式验证自适应层剪枝率模块的有效性。可以看出，当分块大小为 32×32 时，加入自适应层剪枝率模块的方案 4 对比未加入的方案 3，精度提高了 2.5%，且此时的 FLOPS 压缩率也提升了 1.8%，实验结果表明了自适应层剪枝率分配的有效性。

表 3　自适应层剪枝率模块消融实验结果

Table 3　Adaptive layer pruning rate module

方案	分块大小	知识蒸馏	自适应剪枝	精度	F_1	FLOPS 压缩率
1	未剪枝	×	×	0.830	0.747	0
2	8×4	×	×	0.828	0.750	1.417
3	32×32	√	×	0.766	0.660	1.426
4	32×32	√	√	0.785	0.674	1.451

注：表内“×”表示无该模块，“√”表示包含该模块。

2.1.4　一阶段式迭代 ADMM 训练

表 4 所示是一阶段式迭代 ADMM 训练的消融实验结果，当分块大小为 32×32 时，通过惩罚参数递增的迭代训练方式，实验 4 对比未改进的实验 3，精度提高了 5.7%，验证了该模块的有效性。

表 4　一阶段式迭代 ADMM 训练消融实验结果

Table 4　One-stage iterative ADMM training ablation experiment

方案	分块大小	知识蒸馏	自适应剪枝	迭代剪枝	精度	F_1	FLOPS 压缩率
1	未剪枝	×	×	×	0.830	0.747	0
2	8×4	×	×	×	0.828	0.750	1.417
3	32×32	√	√	×	0.785	0.674	1.451
4	32×32	√	√	√	0.830	0.754	1.451

注：表内“×”表示无该模块，“√”表示包含该模块。

2.2 对比实验

2.2.1 目标检测任务剪枝率对比实验

为了验证目标检测任务中不同剪枝率下的剪枝效果，本文进行了剪枝率对比实验。目标检测任务剪枝率对比实验结果见表 5。

表 5 目标检测任务剪枝率对比实验结果

Table 5 Comparison of pruning rate in object detection task

剪枝率/%	剪枝方法	精度	
		YOLOv3 模型	YOLOv4 模型
0	未剪枝	0.783	0.846
30	YOLObile	0.766	0.829
	本文	0.774	0.837
40	YOLObile	0.760	0.827
	本文	0.764	0.830
50	YOLObile	0.758	0.815
	本文	0.759	0.817

如表 5 所示，本文分别使用 YOLOv3[17]、YOLOv4[20]模型在 VOC07+12 数据集上进行了验证，并选用分块剪枝的代表性方法 YOLObile[16]作为基准进行对标。针对 YOLOv3 模型，当总剪枝率设定为 30% 时，全类平均精度的增长百分比为 1.0%。表明针对目标检测任务，本文提出的方案在常见的 YOLO 系列模型架构和常用的目标检测数据集上具有良好的精度。

为了进一步可视化不同剪枝率下各类别平均精度指标的变化曲线，本文在不同总剪枝率下进行剪枝，结果见图 7。

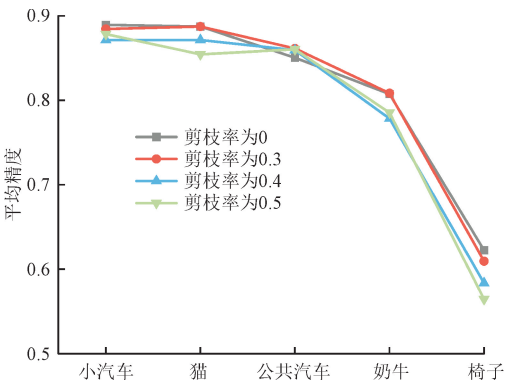

图 7 不同剪枝率平均精度变化图

Fig. 7 Average precision of different pruning rates

如图 7 所示，本文从 VOC2007+2012 数据集的 20 个分类中选取 5 个(小汽车、猫、公共汽车、奶牛、椅子)分为一组绘制折线图。由图可得，随着剪枝率的增大，各类别平均精度呈下降趋势。

2.2.2 分类任务剪枝率对比实验

为了验证分类任务中不同剪枝率情况下的剪枝效果，本文分别使用 ResNet18[21]、ResNet34[21]、MobileNetV2[22]、SENet[23]、DenseNet[24]模型在 CIFAR-100 数据集下进行了剪枝率对比实验，实验结果如图 6 所示。

表 6 分类任务剪枝率对比实验

Table 6 Comparison of pruning rate of classification task

模型	准确率级别	准确率/%				
		未剪枝	剪枝率为 70%		剪枝率为 80%	
			YOLObile[14]	本文	YOLObile[14]	本文
ResNet18	Top1	74.5	74.1	74.6	73.6	73.9
	Top5	92.1	92.0	92.5	91.6	92.4
ResNet34	Top1	74.6	74.3	75.4	73.9	74.5
	Top5	91.6	91.9	92.7	91.5	92.3
MobileNetV2	Top1	75.3	73.8	75.0	72.7	74.6
	Top5	93.2	92.7	93.4	92.2	93.3
SENet	Top1	75.9	74.1	75.6	73.6	75.0
	Top5	92.9	92.5	93.4	92.2	93.3
DenseNet	Top1	77.1	76.2	76.8	74.8	75.8
	Top5	93.3	93.1	94.5	92.8	94.1

从表 6 可得，在 CIFAR-100 数据集下，针对 ResNet18 模型，当总剪枝率为 70% 时，Top1 准确率的增长百分比为 0.67%。针对 MobileNetV2 模型，当总剪枝率为 80% 时，Top1 准确率的增长百分比为 2.6%，Top-5 准确率的增长百分比为 1.2%。本文方案与 YOLObile[16]相比有明显的改进。

3 结 论

本文改进了一种针对 VTA 张量处理器的细粒度结构化稀疏方法。该方法有效解决了传统分块剪枝方法在粗粒度稀疏计算时精度下降的问题，并将任务领域进行了拓展。通过对比实验结果，验证了压缩后的模型推理精度，改进方法在目标检测任务和图像分类任务上均取得了较好的实验结果，同时对剪枝的容忍性更高，可以达到更高的模型稀疏程度。该方法为模型稀疏领域提供了一种高效适配硬件加速器的解决方案，主要包含以下 4 部分贡献：

(1)针对 VTA 的多重循环维度展开特性，对模型的权重张量进行 32×32 大小的分块；

(2)结合时间维度的自蒸馏与空间维度的教师蒸馏，进行多维度特征对齐；

(3)通过一阶段式迭代训练方式，改进原有的 ADMM 算法计算流程，在提升模型部署精度的同时减少训练成本；

(4)提出自适应层剪枝率模块，进行总剪枝率的自适应分配，实现端到端的自动化剪枝。

考虑到针对不同模型压缩方案的普适性问题，在后续的研究中可以设计通用软件压缩工具链[25]，实现快捷通用的模型稀疏化设计。同时可以尝试将其拓展应用于更多不同的网络架构，并将其进一步部署到 VTA 加速的硬件设备上，验证实际 FPGA 硬件推理时的精度及效率[26]。

参考文献：

[1] 高晗，田育龙，许封元，等. 深度学习模型压缩与加速综述 [J]. 软件学报，2021，32(1)：68-92.
GAO Han，TIAN Yulong，XU Fengyuan，et al. Survey of deep learning model compression and acceleration [J]. Journal of Software，2021，32(1)：68-92.

[2] BERTHELIER A，CHATEAU T，DUFFNER S，et al. Deep model compression and architecture optimization for embedded systems：a survey [J]. Journal of Signal Processing Systems，2021，93(8)：863-878.

[3] 符惠桐，王鹏，李晓艳，等. 面向移动目标识别的轻量化网络模型 [J]. 西安交通大学学报，2021，55(7)：124-131.
FU Huitong，WANG Peng，LI Xiaoyan，et al. Lightweight network model for moving object recognition [J]. Journal of Xi'an Jiaotong University，2021，55(7)：124-131.

[4] BA L J，CARUANA R. Do deep nets really need to be deep? [C]//Proceedings of the 27th International Conference on Neural Information Processing Systems. Cambridge，MA，USA：MIT Press，2014：2654-2662.

[5] KIM T，KWON Y，LEE J，et al. CPrune：compiler-informed model pruning for efficient target-aware DNN execution [C]//Computer Vision -ECCV 2022. Cham，Switzerland：Springer Nature，2022：651-667.

[6] LI Zhengang，YUAN Geng，NIU Wei，et al. NPAS：A compiler-aware framework of unified network pruning and architecture search for beyond real-time mobile acceleration [C]//2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway，NJ，USA：IEEE，2021：14250-14261.

[7] GUAN Hui，LIU Shaoshan，MA Xiaolong，et al. CoCoPIE：enabling real-time AI on off-the-shelf mobile devices via compression-compilation co-design [J]. Communications of the ACM，2021，64(6)：62-68.

[8] CHEN Tianqi，MOREAU T，JIANG Ziheng，et al. TVM：an automated end-to-end optimizing compiler for deep learning [C]//Proceedings of the 13th USENIX conference on Operating Systems Design and Implementation. USA：USENIX Association，2018：579-594.

[9] MOREAU T，CHEN Tianqi，VEGA L，et al. A hardware-software blueprint for flexible deep learning specialization [J]. IEEE Micro，2019，39(5)：8-16.

[10] Mudigere D，HAO Yuchen，HUANG Jianyu，et al. Software-hardware co-design for fast and scalable training of deep learning recommendation models [C]//Proceedings of the 49th Annual International Symposium on Computer Architecture. New York，USA：ACM，2022：993-1011.

[11] VADERA S，AMEEN S. Methods for pruning deep neural networks [J]. IEEE Access，2022，10：63280-63300.

[12] HUANG Sitao，PEARSON C，NAGI R，et al. Accelerating sparse deep neural networks on FPGAs [C]//2019 IEEE High Performance Extreme Computing Conference (HPEC). Piscataway，NJ，USA：IEEE，2019：1-7.

[13] ZHOU Aojun，MA Yukun，ZHU Junnan，et al. Learning N：M fine-grained structured sparse neural networks from scratch [EB/OL]. (2021-04-18)[2024-04-01]. https://arxiv.org/abs/2102.04010.

[14] CHANG S E，LI Yanyu，SUN Mengshu，et al. Mix and match：a novel FPGA-centric deep neural network quantization framework [C]//2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA). Piscataway，NJ，USA：IEEE，2021：208-220.

[15] 林景栋，吴欣怡，柴毅，等. 卷积神经网络结构优化综述 [J]. 自动化学报，2020，46(1)：24-37.
LIN Jingdong，WU Xinyi，CHAI Yi，et al. Structure optimization of convolutional neural networks：a survey [J]. Acta Automatica Sinica，2020，46(1)：24-37.

[16] CAI Yuxuan，LI Hongjia，YUAN Geng，et al. YOLObile：real-time object detection on mobile devices via compression-compilation co-design [C]//Proceedings of the AAAI Conference on Artificial Intelligence. Palo Alto，CA，USA：AAAI Press，2021：955-963.

[17] FARHADI A，REDMON J. Yolov3：an incremental improvement [C]//Computer Vision and Pattern Rec-

ognition. Berlin/Heidelberg, Germany: Springer, 2018: 1-6.

[18] HINTON G, VINYALS O, DEAN J. Distilling the knowledge in a neural network [EB/OL]. (2015-03-09) [2024-04-01]. https://arxiv.org/abs/1503.02531.

[19] NIU Wei, LI Zhengang, MA Xiaolong, et al. GRIM: a general, real-time deep learning inference framework for mobile devices based on fine-grained structured weight sparsity [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022, 44(10): 6224-6239.

[20] BOCHKOVSKIY A, WANG C Y, LIAO H Y M. YOLOv4: optimal speed and accuracy of object detection [EB/OL]. (2020-04-23)[2024-04-01]. https://arxiv.org/abs/2004.10934.

[21] BAE W, YOO J, YE J C. Beyond deep residual learning for image restoration: persistent homology-guided manifold simplification [C]//2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). Piscataway, NJ, USA: IEEE, 2017: 1141-1149.

[22] SANDLER M, HOWARD A, ZHU Menglong, et al. MobileNetV2: inverted residuals and linear bottlenecks [C]//2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway, NJ, USA: IEEE, 2018: 4510-4520.

[23] HU Jie, SHEN Li, SUN Gang. Squeeze-and-excitation networks [C]//2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway, NJ, USA: IEEE, 2018: 7132-7141.

[24] HUANG Gao, LIU Zhuang, VAN DER MAATEN L, et al. Densely connected convolutional networks [C]//2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway, NJ, USA: IEEE, 2017: 2261-2269.

[25] FANG Gongfan, MA Xinyin, SONG Mingli, et al. DepGraph: towards any structural pruning [C]//2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway, NJ, USA: IEEE, 2023: 16091-16101.

[26] SUN Yongshuai, GUO Mengyu, LIANG Dacheng, et al. Exploiting dynamic bit sparsity in activation for deep neuralnetwork acceleration [C]//2021 IEEE 14th International Conference on ASIC (ASICON). Piscataway, NJ, USA: IEEE, 2021: 1-4.

(编辑 亢列梅)