基于硬件感知的多目标神经结构搜索方法

许 柯[1)] 孟 源[2)] 杨尚尚[2)] 田 野[3)] 张兴义[2)]

[1)](安徽大学人工智能学院计算智能与信号处理教育部重点实验室 合肥 230601)

[2)](安徽大学计算机科学与技术学院 合肥 230601)

[3)](安徽大学物质科学与信息技术研究院 合肥 230601)

摘 要 神经结构搜索技术可以在大量候选网络集合中搜索到适用于特定任务的神经网络结构.目前,大多数结构搜索网络的部署是针对英伟达GPU、英特尔CPU或谷歌TPU等硬件设备的.然而,将搜索到的架构迁移到一些AI专用加速器中,如寒武纪加速卡或华为Atlas推理加速器,推理效果却表现不佳.主要存在两方面的问题:在搜索空间设计层面,由于硬件架构设计对不同算子的支持存在差异,复用传统的搜索空间到专用神经网络加速器上,其推理效率不是最优的;在结构搜索层面,由于专用神经网络加速器在并行计算资源和数据流水通道等设计的不同,仅采用参数量、计算量作为搜索目标不能准确度量推理延迟,并且限制了神经结构搜索在精度和延迟上的探索空间.为了解决上述问题,本文提出一种基于硬件感知的多目标神经结构搜索方法,首先通过测试不同类型的卷积算子在目标硬件上的性能表现,使用非支配排序设计出定制化的高效搜索空间.然后,将延迟纳入搜索目标,提出一种启发式的混合粒度交叉算子,通过粗粒度阶段间交叉和细粒度阶段内交叉提高种群在多目标下的收敛性和多样性,更好地权衡神经网络的精度和推理延迟.本文主要针对国产寒武纪加速卡MLU270-F4进行了实验分析与方法验证,在CIFAR-10上搜索得到的MLUNet-S4精度比DARTS高0.14%的同时推理速度提升了4.7倍,相比于NSGANet精度仅下降0.04%的同时速度提升了5.5倍;在ILSVRC2012数据集上MLUNet-C相较于具有相同推理速度的MobileNetV2和MnasNet速度上提升了1.2倍的同时预测精度也分别提升了2.3%和0.2%,效果提升显著.

关键词 图像分类;进化算法;多目标神经结构搜索;硬件感知神经结构搜索;寒武纪加速卡

中图法分类号 TP18 **DOI号** 10.11897/SP.J.1016.2023.02651

Hardware-Aware Multi-Objective Neural Architecture Search Approach

XU Ke[1)] MENG Yuan[2)] YANG Shang-Shang[2)] TIAN Ye[3)] ZHANG Xing-Yi[2)]

[1)](*Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education*, *School of Artificial Intelligence*, *Anhui University*, *Hefei* 230601)

[2)](*School of Computer Science and Technology*, *Anhui University*, *Hefei* 230601)

[3)](*Institute of Physical Science and Information Technology*, *Anhui University*, *Hefei* 230601)

Abstract The technology of neural architecture search (NAS) enables the exploration of a vast array of candidate network sets to identify the most suitable neural network for specific tasks. In recent years, neural architecture search methods have achieved remarkable results. However, most of the discovered network architectures are designed for hardware devices such as NVIDIA GPUs, Intel CPUs, or Google TPUs. However, when these searched neural architectures are

收稿日期:2022-11-17;在线发布日期:2023-08-22.本课题得到科技部科技创新2030"新一代人工智能"重大项目(2018AAA0100105)、国家自然科学基金(U21A20512、62206003)资助.**许 柯**,博士,讲师,主要研究领域为神经网络压缩、神经网络架构搜索、异构计算.E-mail:xuke@ahu.edu.cn.**孟 源**,硕士研究生,主要研究领域为深度学习、多目标进化算法、神经结构搜索.**杨尚尚**,博士,主要研究领域为进化多目标优化、神经结构优化以及图学习在智慧教育的应用.**田 野**,博士,副教授,主要研究领域为进化计算.**张兴义**(通信作者),博士,教授,主要研究领域为进化计算以及人工智能.E-mail:xyzhanghust@gmail.com.相关代码已发布在https://github.com/DestinyMy/MLUNet.

migrated to AI-specific accelerators like the Cambricon accelerators or Huawei Atlas inference accelerators, the inference performance falls short of expectations. The endeavor of exploring neural architecture search tailored for specialized hardware encounters two formidable hurdles. Firstly, when it comes to search space design, it is important to consider the hardware architecture supporting different operators. In the case of domestically developed neural network accelerators, reusing a traditional search space may not result in optimal inference efficiency. Secondly, with regards to architectural search, the distinct AI hardware design features disparate provisions for parallel computing resources and data flow channels. Consequently, the exclusive reliance on parameters and computation as the sole search objectives fails to accurately gauge the latency of inference, thereby constraining the potential for exploration in terms of both accuracy and latency. In response to these challenges, this paper puts forth a hardware-aware multi-objective neural architecture search methodology. The objective is to enhance the coherence of neural network architecture exploration across diverse hardware architectures and constraint objectives. This will be achieved by tailoring bespoke search spaces and structures that align with the specific nuances of the underlying hardware architecture. Specifically, we commence by assessing the efficacy of various convolution operators on the designated hardware platform through a stacked operator approach. Subsequently, we craft a customized and efficient search space utilizing non-dominant sorting techniques. Furthermore, we integrate latency metrics evaluated on the physical hardware platform into the search objective and formulate a heuristic crossover operator that is conducive to hardware compatibility. By introducing a coarse-grained inter-stage crossover and a fine-grained intra-stage crossover, we bolster the convergence and diversity of the population under the constraints of multi-objective, thereby attaining an improved balance between the precision and inference latency of the neural network architectures. This study primarily revolves around the execution of experimental analysis and methodological ablation validation concerning the framework posited for the domestic Cambrian MLU270-F4 accelerator card. Numerous empirical investigations have unequivocally demonstrated that the methodology posited in this study possesses superior transportability and deployability across hardware platforms. Specifically, the MLUNet-S4 exhibited a superior accuracy of 0.14% compared to DARTS, alongside an impressive 4.7-fold enhancement in inference speed on the CIFAR-10 dataset. In contrast to NSGANet, MLUNet-S4 experienced a marginal decline in accuracy of merely 0.04%, while achieving an astonishingly accelerated inference speed of 5.5 times faster. On the large-scale ILSVRC2012 dataset, MLUNet-C triumphed over MobileNetV2 and MnasNet, excelling in both speed and accuracy aspects. Notably, the MLUNet-C attained a remarkable acceleration of 1.2 times faster than its counterparts while notably advancing prediction accuracy by an extraordinary 2.3% and 0.2%, respectively. The related code is available at https://github.com/DestinyMy/MLUNet.

Keywords image classification; evolutionary algorithm; multi-objective neural architecture search; hardware-aware neural architecture search; cambricon accelerator

1 引 言

近十年来,深度学习发展迅速,广泛应用于计算机视觉[1]、自然语言处理等诸多领域[2]. 为了追求更高的任务精度,越来越多的模型操作算子和面向复杂连接的模型结构被提出,这使得手动设计模型的过程变得越来越繁琐和耗时[3]. 因此,神经结构搜索[4-8]被提出用来自动设计面向数据集的高效神经网络结构.

神经结构搜索是一个复杂的优化问题，其目标是在大量候选网络结构中搜索满足特定目标约束的神经网络．神经结构搜索主要由三个部分组成：搜索空间、搜索策略以及评估策略[9]．其中，搜索空间包含了所有可能的候选网络结构；搜索策略则用于对搜索空间中的结构进行选择性采样；评估策略决定了如何评估采样网络的性能，并将评估后的性能作为更新搜索策略的反馈．现有的结构搜索方法按照搜索策略可以分为三种：强化学习[4,10-13]，梯度优化[6,14-16]以及进化算法[17-21]．基于强化学习的结构搜索方法[4,7]使用循环神经网络作为控制器预测神经网络的字符串表征序列，将神经网络的性能作为奖赏反馈引导控制器采样更好的网络结构．由于需要额外的控制器训练，在探索高维搜索空间时，基于强化学习的搜索成本要高于其他策略；而基于梯度优化的方法[6]引入包含所有候选网络的超网，在训练过程中对操作引入注意力机制，之后删除重要性较低的操作．由于搜索空间的连续化，该方法可以使用梯度引导，搜索过程更加高效，但由于超网的规模限制，最终搜索到的网络缺乏多样性；基于进化算法的搜索方法[17-19]首先对网络结构进行编码并初始化一个模型结构集合，然后通过种群交叉、变异，结合任务性能评估迭代优化得到更好的网络结构．采用进化算法搜索网络结构可以更好地实现多目标搜索部署．因此，本文采用进化算法实现各类多目标硬件场景下的神经网络结构搜索．

面向硬件部署的神经网络自动设计又被称为硬件感知的神经结构搜索[22]．目前大多数搜索方法都是面向英伟达GPU[20-21,23]或者英特尔CPU[15,24-25]平台搜索其硬件架构适用的神经网络，部分研究则考虑受众广泛的移动设备[11-12,16,26]以及低功耗的现场可编程门阵列(Field-Programmable Gate Array，简称FPGA)和非国产化的专用集成电路(Application Specific Integrated Circuit，简称ASIC)[27-28]上的网络部署．然而由于硬件架构设计以及数据流传输实现的差异，将基于这些硬件平台搜索得到的神经网络结构迁移部署到专用AI硬件平台上并不是最优的．此外，在众多AI加速器设备中，寒武纪加速卡可以提供数据中心级算力，计算性能可以对标英伟达服务器级GPU，并且搭载独特的主动散热技术，在性能充分发挥的情形下实现更低的功耗．由于国外对国内的芯片封锁制裁，诸多公司已采用寒武纪加速卡替换英伟达GPU来进行数据中心的部署，所以基于寒武纪加速卡进行网络结构搜索研究颇具意义．

为针对各类神经网络加速器搜索到精度和推理延迟权衡更好的神经网络结构，本文提出了一种基于硬件感知的多目标进化神经结构搜索方法，并选择寒武纪加速卡MLU270-F4进行实验分析和方法验证，整体方法框架如图1所示．在搜索空间设计方

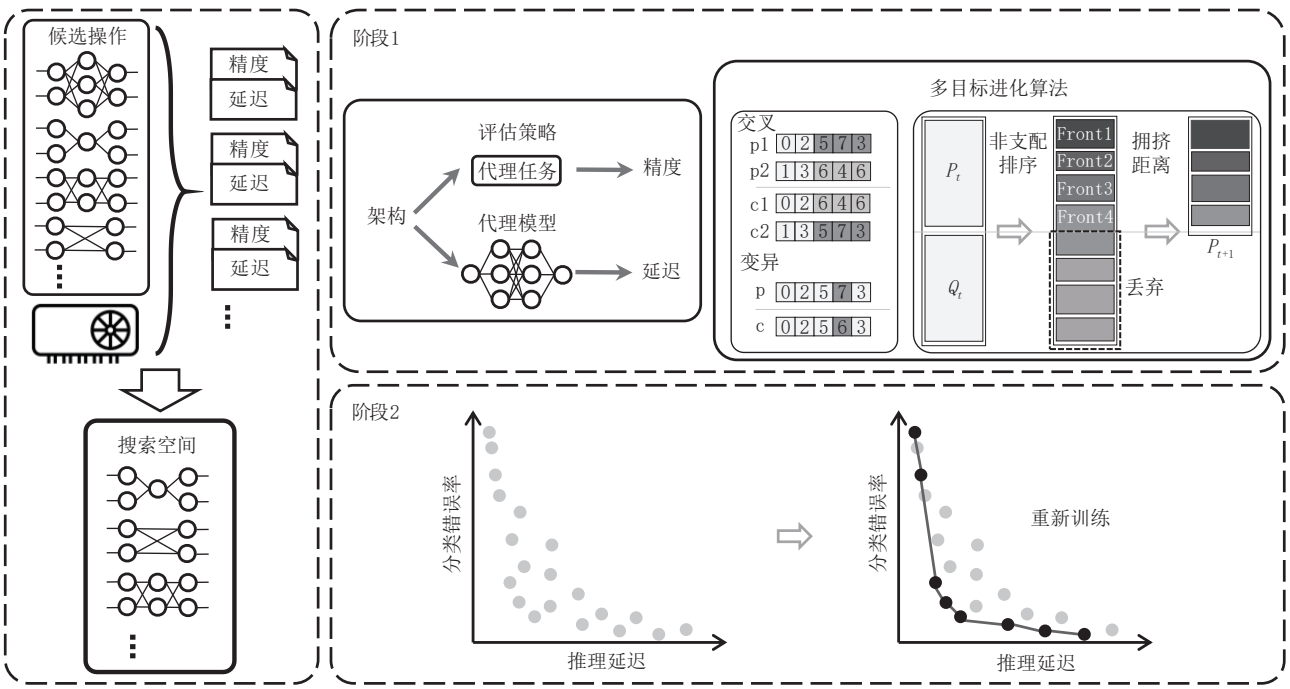

图1　基于硬件感知的多目标神经结构搜索方法框架

面，通过测试操作算子在目标硬件上的延迟表现，设计了专用的硬件感知搜索空间，在降低内存访问代价的同时可以保证预测精度；在网络结构的进化过程中提出混合粒度交叉算子，结合NSGA-II框架[29]，保证种群多样性并逐步向帕累托最优前沿面靠近，其中在推理延迟指标的评估过程中引入多层感知机作为预测器，通过精确的延迟预测有效地探索搜索空间中延迟更优的区域；搜索阶段结束后，重新完整训练最后一代种群帕累托最优前沿中的网络个体．此外，多组对比实验也验证了本文所提方法的有效性．

本文的主要贡献可分为如下三点：

(1)本文提出一种基于硬件感知的多目标神经结构搜索方法，可以实现各类硬件架构上的高效模型设计，并选择寒武纪加速卡设计了专用且高效的神经网络搜索空间，降低网络的内存访问代价从而成倍提高其在寒武纪MLU270-F4上的推理速度并保证网络的预测精度．

(2)本文提出一种启发式的混合粒度交叉算子，通过粗粒度的阶段间交叉和细粒度的阶段内交叉，提高了种群的多样性和收敛性，可以更好地权衡专用硬件加速器上神经网络的精度和推理延迟．

(3)通过和现有多种经典手工神经网络和神经结构搜索算法的实验结果对比可以发现，本文提出的方法可以搜索到对精度和推理延迟权衡更好的网络模型，并设计了搜索空间、交叉算子以及延迟预测器的消融实验，验证了所提方法的有效性．

本文第1节介绍神经结构搜索的相关背景与研究动机．第2节介绍相关工作，包括寒武纪加速卡、基于进化算法的神经结构搜索以及硬件感知的神经结构搜索算法．第3节介绍本文提出的硬件感知的多目标进化结构搜索算法．第4节通过对比实验和消融实验验证所提方法的有效性．最后总结全文．

2 相关工作

2.1 寒武纪加速卡MLU270-F4

MLU270-F4芯片采用寒武纪MLUv02架构，加速的核心是神经功能单元(Neural Function Unit，简称NFU)．相比于依赖频繁数据读取的高并行、高能耗GPU架构，寒武纪硬件架构整体采用了更加细粒度的流式处理结构[30]，可以更好地减少计算单元对输入带宽的依赖，实现神经网络计算过程中更高效的数据复用；存储层次设计方面，MLUv02架构采用分层级存储方式，包含多级缓存和共享存储结构，并依据存储结构设计了专用的缓存控制器和全局控制引擎，可充分利用处理器内部存储进行性能优化，除了在片上配置共享缓存之外，在每个计算单元中也配置专属存储器，使计算单元独享其带宽并减少对共享缓存的访问．图2展示了GPU和寒武纪的硬件架构，其中GPU由若干包含单指令多线程(Single-Instruction Multiple-Thread，简称SIMT)计算核心的SIMT核心簇以及互联网络构成，而寒武纪采用全局控制引擎和包含若干处理单元(Processing Element，简称PE)的NFU计算单元，通过直接内存访问(Direct Memory Access，简称DMA)控制器调节输入神经元的输入缓冲阵列(Neuron Buffer Input，简称NBin)和输出神经元的输出缓冲阵列(Neuron Buffer Output，简称NBout)．可以看出，寒武纪芯片可以更高效率的执行类似标准卷积运算的高数据复用型操作，而诸如深度可分离卷积这类低数据复用型算子则执行效率并不高，该现象在搜索空间的选择实验结果中也得到了证实．

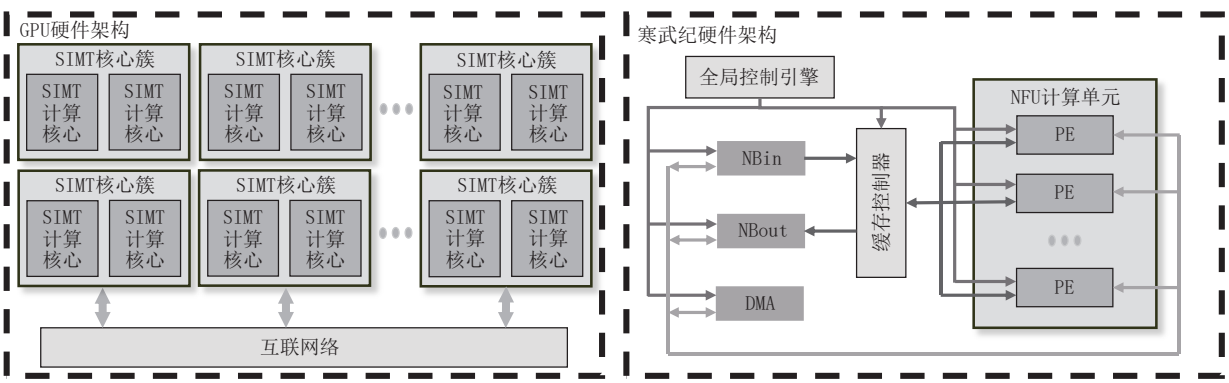

图2 GPU与寒武纪硬件架构对比图

2.2　基于进化算法的神经结构搜索

进化算法是实现多目标神经结构搜索的典型方法．从个体编码策略的角度，基于进化算法的神经结构搜索可分为两类：宏架构编码和微架构编码．前者将完整的神经网络结构编码为个体，而后者仅编码Cell结构，完整的神经网络通过Cell的堆叠构建．对于宏架构编码，Real等人[18]采用有向无环图来表示整个神经网络，这是进化算法首次应用于神经结构搜索，其中子代个体通过多种变异操作产生．由于搜索空间规模较大，该算法需在250台计算机上并行运行15天．Xie等人[17]将网络结构分为多个阶段以减小搜索空间规模，并通过遗传算法进行搜索．相比于多分支结构，链式神经网络[31]更适合使用交叉算子进行结构的重组，但不便表示跨层连接．因此，许多研究通常将多个基本操作封装成块[32-33]，比如ResNet块、DenseNet块等．不同于宏架构编码，大量实验表明许多有效的人工设计的神经网络是由相同Cell进行堆叠构建的[34-36]，受这一发现的启发，许多研究仅搜索Cell结构以进一步减少搜索空间[37-38]．Liu等人[39]提出了分层Cell搜索空间，其中基础操作包括卷积、池化等，更高层的候选操作由相邻的下层操作组合构成．为了保持种群中神经结构的多样性，该工作保留了种群中所有个体应用锦标赛选择，并将锦标赛规模设为种群规模的5%．Real等人[19]将Cell分为Normal Cell和Reduction Cell，并使用种群中个体保留的代数作为选择依据，用子代个体替换种群中保留代数最大的个体来探索更大的搜索空间．

尽管在各种数据集上取得了令人印象深刻的改进，但上述进化算法的计算效率极低，比如Real等人[19]提出的方法在450块GPU上并行运行需要7天．为了减少进化神经结构搜索中真实评价的个体数量，Sun等人[40]使用代理模型筛选出有潜力的个体进行真实评价；Suganuma等人[41]抛弃早期训练阶段中性能较差的个体．为了减少种群中个体的评估代价，Liu等人[42]使用原始数据集的子集进行个体训练；Zhang等人[43]通过继承父代个体中尽可能多的权重参数减少子代的训练代价．

基于进化算法的硬件感知结构搜索中，Li等人[21]结合偏序思想，使用进化算法权衡网络的精度和英伟达GPU推理延迟；Yang等人[44]为解决小模型陷阱问题，对网络精度和参数量以及精度增长速度和参数量分别进行非支配排序，保留有潜力的大模型；Cai等人[45]使用进化算法在训练后的超网中进行结构搜索[46]，在超网训练阶段并未涉及进化搜索．综上所述，上述结构搜索方法均针对英伟达GPU、英特尔CPU或者常见的移动设备，鲜有方法基于寒武纪加速卡；并且考虑到硬件架构设计上的差异，上述工作的结果在寒武纪加速卡中并不具备良好的迁移性．因此，本文将在寒武纪加速板卡MLU270-F4上实现高效的多目标神经网络结构搜索和部署．

2.3　硬件感知的神经结构搜索算法

近些年来专为深度神经网络设计的硬件加速器逐渐流行，它们可以显著地降低神经网络推理阶段的延迟和能量消耗，而硬件加速器的性能（比如延迟，能量消耗，芯片面积等）和神经网络的属性（比如层数，参数量等）息息相关，所以在神经结构搜索过程中考虑硬件约束也是必不可少的．为了衡量模型的效率，NASNet等工作[7,10]优化FLOPs，然而FLOPs低或者参数量低的网络结构并不意味着网络推理迅速[11,47]．NetAdapt[47]使用目标硬件平台的经验性延迟表贪婪地在给定延迟约束下最大化网络模型的精度；MnasNet[11]同样使用延迟表，使用强化学习为移动设备搜索最优神经网络；FBNet[16]使用延迟表估计网络延迟，并提出延迟感知的损失函数，通过优化梯度来搜索网络结构；ChamNet[15]则通过在搜索过程中结合预测模型，为目标硬件找到最佳网络结构；基于梯度的神经结构搜索方法ProxylessNAS[12]使用直接测量的硬件延迟指标，分析每个操作算子的延迟并构建预测模型，将延迟作为损失函数中可微分的正则项引导搜索过程，但是MobileNetV3[13]指出多目标损失函数在处理小模型时并不友好，直接应用延迟约束更为妥当；Once-for-all[45]使用一个预训练的超网模型，通过对超网中不同规模的子模型的筛选进行不同硬件平台的网络部署．然而，上述方法是针对英伟达GPU、英特尔CPU或者移动设备等进行结构搜索，鲜有方法基于专用神经网络加速器进行网络部署．实验发现，基于上述硬件搜索得到的网络结构对于硬件架构相异的加速器，比如寒武纪加速卡，并不具备良好的迁移性．因此针对硬件特性挖掘搜索空间设计专用网络结构是很有必要的．

3　本文方法

在本节，我们首先针对多目标进化神经结构搜索问题进行建模，并通过实验论证了模型推理

延迟与参数量和计算量的关系；然后详细介绍面向硬件感知的多目标神经结构搜索方法，具体包括硬件定制化搜索空间的设计以及使用基于混合粒度交叉算子的NSGA-II改进算法进行搜索空间探索.

3.1 多目标进化神经结构搜索问题建模

神经结构搜索中常见的优化目标为网络预测精度，但面向网络模型的硬件部署通常需要平衡至少一个与之冲突的目标，比如参数量，计算量等. 针对不同部署场景设计具有不同复杂性的高性能架构问题一般视为多目标双层优化问题[48]，该问题可以采用公式1建模表达：

$$\begin{aligned}&\text{minimize } \boldsymbol{F}(\boldsymbol{x})=\left(f_1(\boldsymbol{x};\boldsymbol{w}^*(\boldsymbol{x})),f_2(\boldsymbol{x})\right)^T,\\&\text{subject to } \boldsymbol{w}^*(\boldsymbol{x})\in\arg\min\mathcal{L}(\boldsymbol{w};\boldsymbol{x}),\\&\qquad\boldsymbol{x}\in\boldsymbol{\Omega}_x,\boldsymbol{w}\in\boldsymbol{\Omega}_w\end{aligned}\tag{1}$$

其中，$\boldsymbol{\Omega}_x$为神经网络搜索空间，$\boldsymbol{x}$为候选神经网络，里层变量$\boldsymbol{w}\in\boldsymbol{\Omega}_w$为其关联的权重. 目标向量$\boldsymbol{F}$由网络验证集分类错误率($f_1$)和参数量或计算量(FLoating point OPerations，简称FLOPs)等(f_2)构成. $\mathcal{L}(\boldsymbol{w};\boldsymbol{x})$为网络在训练集上的交叉熵损失.

通常，自动搜索到的神经网络会比手工设计的具有更高的预测精度以及更低的参数量[6,18]，但大部分搜索方法并没有显式地考虑到硬件部署的推理延迟. 虽然参数量和FLOPs已经被广泛用于神经网络的自动设计[10,20,49-50]，但是最近的研究表明，理论上参数量或者FLOPs的改进并不总是在硬件设备上转化为更好的推理延迟，并且在某些情况下并不理想[11,47]. 为了对比神经网络的参数量和FLOPs与专用神经网络加速器推理延迟的相关性，本文选择寒武纪加速卡在设计的搜索空间中随机采样了1000个网络结构，绘制了如图3所示的散点图，横轴表示网络的参数量或者FLOPs，纵轴表示网络在寒武纪加速卡上的真实推理延迟. 从图3中可以看到，网络的参数量和FLOPs与推理延迟大致为正相关，但存在很多截然相反的情况. 我们在图中使用虚线标注了同样的参数量或者FLOPs时网络的推理延迟跨度，可以看出即使参数量或者FLOPs很大时同样存在推理速度很快的网络，而规模很小的网络推理速度也并不总是优于更大型的网络结构，优化网络的参数量或者FLOPs并不一定会给推理延迟带来增益. 因此，本文为了得到在硬件设备上确实更加友好的神经网络结构，将第二个目标设为硬件实际推理延迟，即目标向量$\boldsymbol{F}$由网络验证集分类错误率(f_1)和硬件实际推理延迟(f_2)构成.

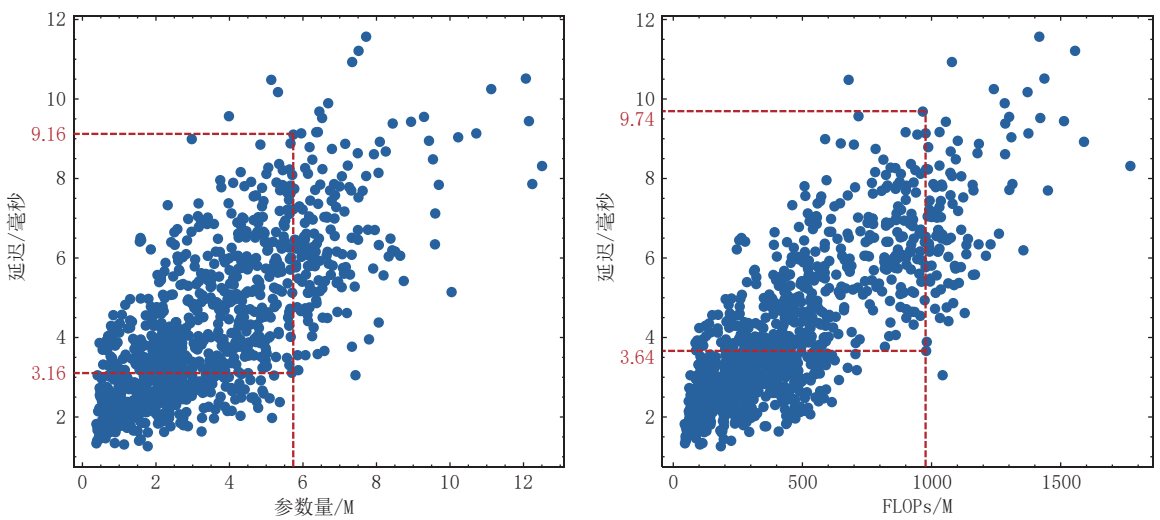

图3 神经网络参数量和FLOPs与推理延迟的相关性

使用进化算法优化神经网络结构所必需的是网络的编码策略，该策略定义了基因型-表型，其中基因型是网络编码，表型是不同的神经网络架构. 图4展示了基于进化算法的神经结构搜索的基本框架[17-20,40,48,51]，通常包括七个步骤：(1) 基于编码策略随机生成若干个神经网络编码构建初始种群；(2) 将网络编码映射到对应的神经网络架构进行评价得到适应度值；(3) 根据适应度值采用合适的策略选择父代个体生成交配池；(4) 根据交配池使用交叉、变异等操作生成子代；(5) 对子代个体进行评价得到适应度值；(6) 合并父代种群和子代种群，采用合适的环境选择策略选取个体作为下一代种群；

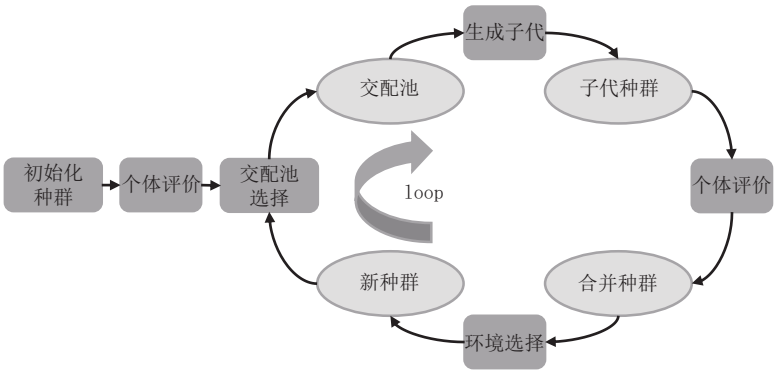

图4　基于进化算法的神经结构搜索基本框架

(7) 判断退出条件，若不满足则返回步骤(3)，否则输出最后一代种群.

基于进化算法的神经结构搜索中，个体性能的评估是整个流程中最耗时的部分. 早期的方法一般是将所有个体进行完整的训练得到个体的预测精度，比如AmoebaNet[19]一次搜索过程需要花费3150天，之后的研究工作一般使用代理任务或代理模型加速评价过程. 代理任务一般从数据集、训练代数或者模型大小上入手，比如使用训练集子集训练网络，或者以较少代数的训练结果近似最终的结果等；代理模型一般先建立网络结构与其性能之间的映射关系，之后通过该映射关系获取未知网络的性能. 区别在于，代理模型需要大量数据进行训练，对于分类错误率而言，完整训练大量神经网络作为数据集代价颇高；而对于推理延迟而言，数据收集花费的时间是可以承受的，并且带来的加速增益也十分可观，比如在搜索过程中不考虑迁移部署到目标硬件设备上的时间，平均一个网络的推理时间大约为5分钟，但使用预测器进行评估后，花费的时间只有1.2秒左右. 因此，本文分别使用代理任务和代理模型对分类错误率和推理延迟进行代理评价，其中，代理任务通过减少个体的训练轮数实现，代理模型通过多层感知机神经网络构建网络结构与目标硬件设备推理延迟的映射关系实现.

3.2　硬件感知的多目标神经结构搜索方法

3.2.1　神经网络加速器定制化搜索空间设计

现有的神经结构搜索方法在搜索空间的设计层面并没有考虑到硬件的内存访问代价以及操作算子在不同硬件平台上运行的速度差异性，大多根据前人工作设计，搜索到的神经网络在特定硬件上并不是最优的. 本文通过测试不同类型的卷积算子在目标神经网络加速器上的性能，使用非支配排序设计了定制化的搜索空间，设计流程如算法1所示.

首先收集现有流行的卷积神经网络操作算子，包括标准卷积、残差卷积、空间可分离卷积、空洞卷积、深度可分离卷积、倒置残差卷积等等，之后线性堆叠这些操作算子构建测试网络，网络构建参考NSGANet与ResNet，将网络分为三个阶段，并设置初始通道为64，前两个阶段后使用最大池化层进行下采样并且通道加倍，每个阶段线性堆叠测试算子，如公式2所示：

算法1.　硬件定制化搜索空间设计流程.

输入：候选操作算子集合$CandidateOP$，集合大小为N

初始化目标向量$objs=\varnothing$

FOR $i=1,\cdots,N$ DO

　使用公式2生成测试网络net_i

　在训练集$DSet_{train}$上训练net_i

　在验证集$DSet_{val}$上得到net_i精度acc_i

　在$DSet_{val}$上推理得到net_i推理延迟$latency_i$

　将性能对$(acc_i, latency_i)$添加到$objs$中

END FOR

对$objs$做非支配排序得到帕累托前沿$FrontNo$

结合$FrontNo$筛选算子得到专用搜索空间SS

输出：专用搜索空间SS

$$\begin{cases} Net \leftarrow \{64, stage_1, mp, stage_2, mp, stage_3, classifier\} \\ stage_k \leftarrow \overbrace{op_i}^{n} \end{cases} \tag{2}$$

其中，$stage_k$表示Net中使用候选操作op_i重复线性堆叠构成的第k个阶段，n表示堆叠数量。

本文中取$n=4$，接着在CIFAR-10数据集上训练100代，并得到这些测试网络的预测精度以及在目标硬件(MLU270-F4)上的推理延迟，最终使用非支配排序对这些测试网络进行排序，结合帕累托最优前沿选择优质操作算子构建操作搜索空间. 结果如表1所示. 表1中，0号操作CBR-k3表示卷积核

大小为3的标准卷积加上批量归一化以及ReLU激活函数构成的卷积块，后续的操作都是基于这种模块构建的，主要目的是降低搜索复杂度. 值得注意的是，MobileNet[52]中提出的深度可分离卷积在寒武纪加速卡上表现较差，具体数据分析可见4.3.1节搜索空间有效性分析，这也证明了同样的操作算子并不一定适用于多种硬件平台，对特定硬件选择更为适用的算子是很有必要的. 同时，EfficientNet等工作[50,53]指出通道对于神经网络推理延迟的影响十分明显，但DARTs、NASNet等工作[6-7,10,17-18,20]均采取固定初始通道个数的方式构建神经网络，为增加搜索空间的饱满度，本文将初始通道也作为一个搜索因子，即 $Init\_Channel=\{16, 24, 32, 40, 48, 64\}$.

表1 适用于寒武纪加速卡的专用搜索空间

操作编号	操作表示	操作类型	卷积核大小	膨胀系数	通道扩展率
0	CBR-k3	标准卷积	3	-	-
1	RB-k3-d1	残差卷积	3	1	-
2	BN-k3-d2	瓶颈卷积	3	2	-
3	BN-k5-d2	瓶颈卷积	5	2	-
4	IRB-k3-d1-e3	倒置残差卷积	3	1	3
5	IRB-k3-d1-e6	倒置残差卷积	3	1	6
6	IRB-k5-d1-e3	倒置残差卷积	5	1	3
7	IRB-k5-d1-e6	倒置残差卷积	5	1	6

除了设计操作搜索空间，网络框架对推理延迟的影响也不容忽视. 之前的结构搜索研究中，网络框架大多是基于Cell，如图5左所示. 这种结构可以搜索到精度高、规模小的网络结构，但是由于Cell内部节点之间的密集连接，会大大增加硬件的内存访问代价，从而增加网络的推理延迟；并且最终的网络是通过对Cell结构堆叠固定次数构建，但堆叠次数与Cell内部结构的不恰当组合通常会造成精度的明显下降、显著的计算量增加或者延迟的大幅增长. 为了解决上述问题，文本设计了一种高效的基于Layer的网络框架，如图5中所示，将整个网络分为三个阶段，与NSGANet、AmoebaNet类似，区别在于，除了对输入进行预处理的Stem层以及输出处理的分类层，其余全部搜索得到. 每个阶段允许不同的操作层数，同时每层允许不同的操作算子，操作之间线性连接，减少内存访问代价的同时增加网络的多样性；数据经过前两个阶段后对特征进行下采样和通道加倍，该操作通过将对应算子步长设为2实现，因为池化层会丢失大量特征信息. 图5右所示为网络编码与其在CIFAR数据集上的对应网络结构，使用Stem层对输入图像进行预处理之后，依次堆叠操作算子，最后经过分类层输出预测类别，CIFAR和ImageNet网络构建的主要差异在于Stem层，具体构建方法和NASNet[4]、DARTs[6]类似.

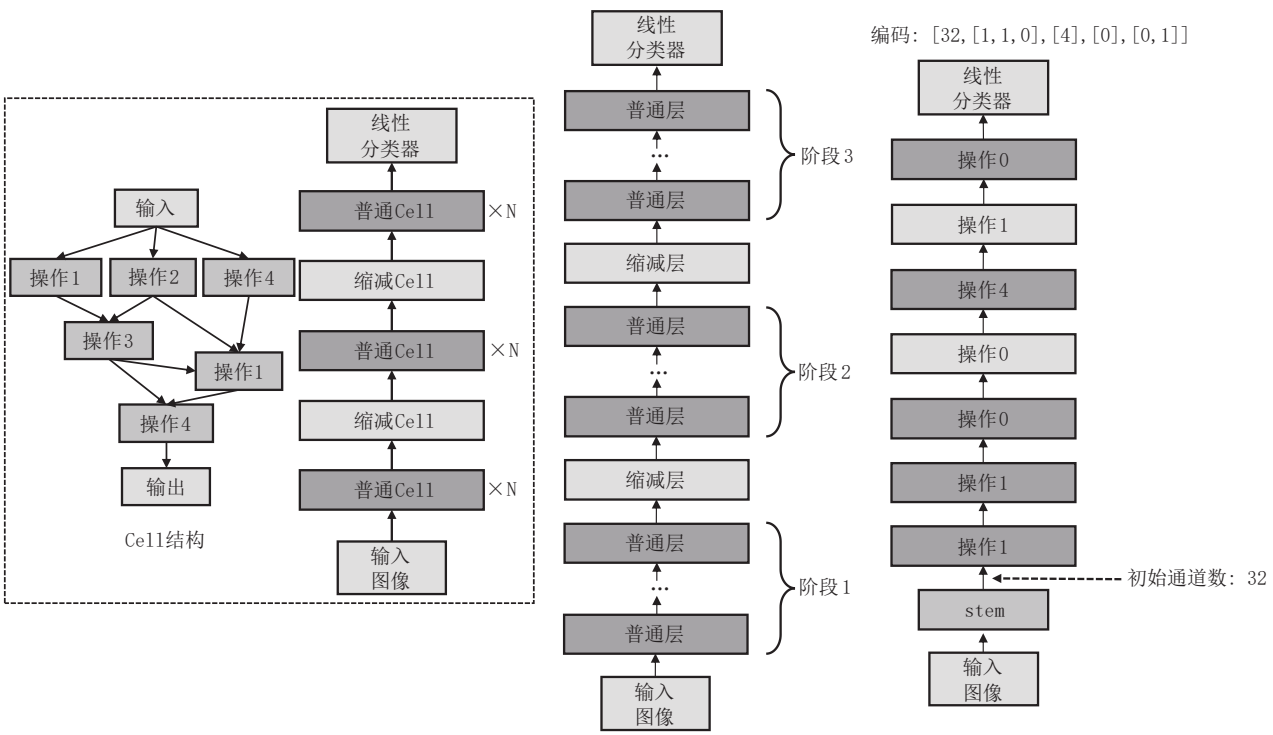

图5 基于Cell的网络结构以及本文提出的网络框架

3.2.2　混合粒度交叉算子优化

交叉算子是进化算法中最主要的操作算子，对种群的搜索性能起着重要的作用．现有进化神经结构搜索方法中的个体交叉方式通常为单点交叉[17,20]，其操作流程可概括为随机交叉点，之后交叉前一段的个体编码或对编码每位随机交叉．这种交叉方式具有较大的随机性，并没有考虑到个体的优劣，盲目的强制交叉只会带来时间的损耗，发现更优个体的概率并不高．

算法2.　混合粒度交叉算子操作流程．

输入：父代个体p_1, p_2，交叉概率p_c
将p_1按阶段表示为$\{stage_1^1, stage_2^1, stage_3^1\}$
将p_2按阶段表示为$\{stage_1^2, stage_2^2, stage_3^2\}$
初始化两个子代个体$q_1=\{\}, q_2=\{\}$
$rand=random()$
IF $rand<p_c$ THEN
　FOR $i=1, \cdots, 3$ DO
　　将$stage_i^1$按操作表示为$(op_1^1, op_2^1, \cdots, op_m^1)$
　　将$stage_i^2$按操作表示为$(op_1^2, op_2^2, \cdots, op_n^2)$
　　IF $acc(p_1)>acc(p_2)$ THEN
　　IF $m>n$ THEN
　　$stage_i^2=(op_1^2, op_2^2, \cdots, op_n^2, op_{n+1}^1, \cdots, op_m^1)$
　　ELSE
　　$cp=random(1, m+1)$
　　$stage_i^2=(op_1^1, op_2^1, \cdots, op_{cp}^1, op_{cp+1}^2, \cdots, op_n^2)$
　　ELSE
　　IF $m>n$ THEN
　　$cp \leftarrow random(1, n)$
　　$stage_i^1=(op_1^2, \cdots, op_{cp}^2, op_{cp+1}^1, \cdots, op_m^1)$
　　ELSE
　　$stage_i^1=(op_1^1, op_2^1, \cdots, op_m^1, op_{m+1}^2, \cdots, op_n^2)$
　　将$stage_i^1$添加到q_1中
　　将$stage_i^2$添加到q_2中
　END FOR
ELSE
　$q_1=p_1, q_2=p_2$
　$rand\_s=random(1, 4)$
　交换q_1, q_2中第$rand_s$个阶段
输出：子代个体q_1, q_2

不同于传统的交叉方式，本文根据个体之间的性能对比关系提出一种启发式的混合粒度交叉算子，交叉流程如算法2所示．算法中为了表示简便省略了下采样层，$stage_i^1$、$stage_i^2$分别表示p_1、p_2第i个阶段，$stage$由若干操作线性堆叠构建．若随机数$rand<p_c$，对每个阶段执行阶段内细粒度交叉；否则随机一个阶段执行阶段间粗粒度交叉．两种不同粒度的交叉方式保证了种群的多样性，阶段内细粒度交叉同时会提高种群的收敛性．使用阶段内细粒度交叉时，为了方便描述这里假设$m>n$，我们可以考虑两种情况：当p_1的性能优于p_2时，我们直觉地认为较大的可能性是长度导致的，因为较深的神经网络精度一般会优于较浅的神经网络，所以将$stage_i^1$的多余位拼接到$stage_i^2$上；若p_2的性能反而优于p_1，说明$stage_i^2$中的操作算子是更优的，这时将$stage_i^2$交叉点前编码赋值给$stage_i^1$对应位来增强p_1性能，增加后代更优的概率．因此，混合粒度交叉算子可以从模型算子选择和模型深度两个层面保证交叉算子产生后代的质量．

算法3.　基于改进NSGA-II的结构搜索算法．

输入：种群大小N，最大进化代数G，交叉概率p_c
初始化包含N个个体的初始种群P_0
对P_0进行评估得到目标向量$objsp$
对$objsp$做非支配排序得到帕累托前沿$FrontNop$
根据$objsp$和$FrontNop$计算拥挤距离$CrowdDisp$
FOR $i=1, \cdots, G$ DO
　$MatingPool=$
　　$BinaryTourSelect(P_{i-1}, FrontNop, CrowdDisp)$
　FOR p_1, p_2 IN $MatingPool$ DO
　　$q_1, q_2=Hybridgrancrossover(p_1, p_2, p_c)$//算法2
　　$q_1, q_2=Mutate(q_1, q_2)$
　　将q_1, q_2添加到子代种群Q_i中
END FOR
对Q_i进行评估得到目标向量$objsq$
对$objsq$做非支配排序得到帕累托前沿$FrontNoq$
根据$objsq$和$FrontNoq$计算拥挤距离$CrowdDisq$
$R_i=P_{i-1} \cup Q_i$
$P_i, FrontNop, CrowdDisp=$
　$EnvironmentSelect(R_i, FrontNor, CrowdDisr)$
END FOR
输出：最后一代种群P_G

为了更好地权衡神经网络预测精度和推理延迟，本文将混合粒度交叉算子嵌入到NSGA-II[29]框架中优化神经网络结构，以提高种群的收敛性．NSGA-II是一种基于帕累托支配的鲁棒元启发式多目标进化算法，通过快速非支配排序方法，在环境选择中结合精英选择策略，在降低算法复杂度的同时，加快了种群的收敛．算法3为基于改进NSGA-II的神经结构搜索算法的主要流程．首先，初始化N

个个体的种群，并计算种群中每个个体的目标值，通过目标值对个体进行非支配排序并计算个体之间的拥挤距离．然后进行迭代优化，结合排名和拥挤距离使用二进制锦标赛选择父代个体．父代个体通过混合粒度交叉和变异生成子代个体，变异为对结构编码中的每一位按照一定概率进行突变．最后通过环境选择生成新的种群进行下一次迭代的进化，达到预设的迭代次数后，输出最后一代种群．

4 实验分析

4.1 实验设置

本文在公开图像数据集上进行实验，包括CIFAR以及ISLVRC2012．表2给出了数据集所对应的详细信息．实验结果采用网络的泛化性能和寒武纪加速卡的推理延迟来评估网络的性能，神经网络的泛化性能为该网络训练后在给予未知数据情况下网络的预测准确度，推理延迟为神经网络在硬件设备上数据批量大小为1时推理所消耗的时间[17]，将CIFAR-10或ISLVRC2021数据集中的测试集用作延迟计算，每次输入一张图像并记录下网络处理图像花费的时间，直至整个测试集迭代完毕取平均值作为网络处理每张图像的平均推理时间，将该过程执行三次，最后将三次测量的推理时间取平均作为该网络的推理延迟．

表2 实验数据集

数据集	类别数	训练集	验证集	测试集
CIFAR-10	10	50000	-	10000
CIFAR-100	100	50000	-	10000
ISLVRC2012	1000	1281167	50000	100000

本文方法的实现基于Python以及PyTorch 1.8，实验分为两个阶段：搜索阶段，由于进化算法是基于种群的，为了节省搜索时间，本文使用了代理任务进行网络的训练，种群大小设置为20，进化代数为25，交叉概率为0.2，变异概率为编码长度的倒数，每阶段的网络深度范围为[1,10]，神经网络训练25轮，使用随机梯度下降优化结合余弦退火学习率策略将学习率从0.1逐步衰减到0.001，权重衰减系数设置为0.0003，梯度裁剪最大范数为5.0，训练批量大小为128，测试批量大小为500；训练阶段，选取最后一代种群的最优帕累托前沿中的网络进行完整的训练，在CIFAR上网络的训练轮数提升为600，最大学习率和最小学习率分别为0.025、0，权重衰减系数设置为0.0005，训练批量大小为80，dropout概率为0.4；在ImageNet上网络的训练轮数为250，训练批量大小为512，学习率设置为0.3，权重衰减系数设置为0.00003，学习率每次迭代衰减为原来的0.97，dropout概率为0．训练神经网络时使用数据增强cutout以及辅助分类器，辅助分类器输出占比为0.4．

延迟预测器超参数设置与NSGANetV2[54]保持一致，数据集从专用搜索空间中随机采样8000个网络结构、真实延迟数据对，其中7000个数据对作为训练集，1000个作为测试集，预测器输入为初始通道数、网络层数量、个体中不同操作数量、网络参数量、FLOPs以及寒武纪加速卡核心数量．

4.2 实验结果与分析

为了评估本文提出的多目标神经结构搜索方法的有效性，本文分别与经典手工设计网络和神经结构搜索方法进行了比较．其中，手工设计的网络包括有ResNet[34]、DenseNet[35]、MobileNet[52]、ShuffleNet[55]等；神经结构搜索方法包括有NASNet[10]、DARTS[6]、LEMONADE[56]、NSGANet[48]、AmoebaNet[19]、FBNet[16]、MnasNet[11]等．图像数据集CIFAR和ISLVRC2012上精度和寒武纪加速卡延迟的对比结果分别如表3、表4所示，对比算法的预测精度是在对应论文的实验结果中获取，推理延迟同样使用实验设置中描述的测量方法获得．

从表格中可以看到，本文提出的方法在CIFAR和ISLVRC2012数据集上的效果优于绝大部分现有的神经结构搜索方法．其中在CIFAR上的实验，本文搜到的网络在与其他模型精度相当或更好的前提下，推理速度至少快3倍，甚至一个量级，比如MLUNet-S4和NSGANetV1-A2在精度水平相当的情况下，前者比后者快5.5倍，MLUNet-S1和PNAS在精度水平相当的情况下，前者比后者快13.5倍，主要的原因在于这些结构搜索方法均是基于Cell进行搜索的．值得注意的是，MLUNet-S5无论在参数量还是FLOPs指标上都是远大于对比算法的，但推理速度提升了至少2倍，精度也超过了所有对比算法；MLUNet-S4在FLOPs几乎相同的情况下，推理速度提升了近5倍，而精度也超过大多数对比算法；MLUNet-S3在参数量几乎相同的情况下，推理速度得到了相比于MLUNet-S4更为显著的提升，精度只下降了0.2%．因此，从上述对比结果也可以看出，采用参数量和FLOPs间接度量专用硬件上的推理延迟是不准确的，甚至存在相关性小

表3　CIFAR数据集不同方法对比结果

模型名称	搜索策略	延迟(毫秒)	参数量(M)	FLOPs(M)	CIFAR-10(%)	CIFAR-100(%)
ResNet110[34]	手工设计	3.4516	1.73	255.81	93.57	—
DenseNet (k=12)[35]	手工设计	—	—	—	95.90	79.80
DenseNet (k=24)[35]	手工设计	—	—	—	96.26	80.75
DenseNet-BC (k=24)[35]	手工设计	25.0959	15.32	5546.86	96.38	82.40
DenseNet-BC (k=40)[35]	手工设计	23.7961	25.62	9431.86	96.54	82.82
AE-CNN+E2EPP[40]	进化算法	—	—	—	94.70	77.98
Block-QNN-S[57]	强化学习	—	—	—	95.62	79.35
PNAS[5]	序列化 基于模型优化	24.2859	4.48	729.63	96.59	82.37
NASNet-A[10]	强化学习	20.8766	3.83	624.23	97.35	83.42
ENAS[58]	强化学习	20.6725	3.86	626.75	97.11	82.73
DARTs-V1[6]	梯度优化	16.3553	3.17	518.94	97.06	—
DARTs-V2[6]	梯度优化	17.6785	3.35	547.47	97.17	82.46
P-DARTs[59]	梯度优化	19.0651	3.43	550.79	97.50	82.80
PC-DARTs[60]	梯度优化	18.6992	3.63	576.06	97.43	82.64
CDARTs[61]	梯度优化	20.2644	3.86	611.23	97.52	84.31
LEMONADE[56]	进化算法	—	—	—	96.95	—
NSGANetV1-A1[48]	进化算法	14.4114	2.39	395.96	96.51	80.77
NSGANetV1-A2[48]	进化算法	20.5015	3.44	563.87	97.35	82.58
AmoebaNet-A[19]	进化算法	18.5613	3.15	506.29	96.88	81.07
AmoebaNet-B[19]	进化算法	12.7547	2.71	425.45	97.45	—
MLUNet-S5	**进化算法**	**5.3315**	**10.81**	**1158.14**	**97.66**	**83.04**
MLUNet-S4	**进化算法**	**3.7530**	**4.43**	**505.46**	**97.31**	**83.12**
MLUNet-S3	**进化算法**	**2.3581**	**3.14**	**318.02**	**97.11**	**82.13**
MLUNet-S2	**进化算法**	**2.0733**	**2.60**	**270.96**	**96.85**	**81.24**
MLUNet-S1	**进化算法**	**1.7988**	**2.22**	**240.42**	**96.79**	**80.68**

于零的情况;并且本文提出的专用搜索空间的确能够有效地降低硬件的内存访问代价,从而减少网络的推理延迟.

在ISLVRC2012上的实验,MLUNet较中间NASNet等基于Cell的结构搜索方法相比速度同样可以快2倍以上,和手工设计的网络以及硬件感知的搜索方法相比可以支配绝大多数方法,比如MLUNet-C精度比ResNet34高0.9%同时延迟低0.4毫秒,比MobileNetV2快近1毫秒同时精度提高2.3%,相比于结构搜索方法FBNet-B而言,精度高

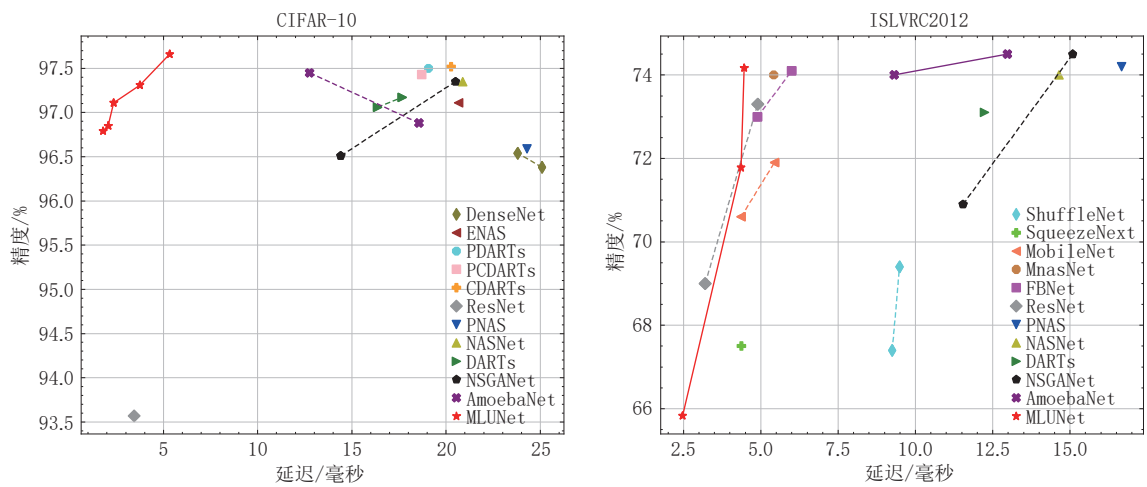

图6　CIFAR-10和ISLVRC2012上网络性能的帕累托前沿散点图

表4 ISLVRC2012数据集不同方法对比结果

模型名称	搜索策略	延迟(毫秒)	参数量(M)	FLOPs(M)	Top1(%)	Top5(%)
ResNet18[34]	手工设计	3.1998	11.69	1822.18	69.0	-
ResNet34[34]	手工设计	4.8982	21.80	3675.63	73.3	91.4
ShuffleNetV1[55]	手工设计	9.2502	3.44	274.54	67.4	-
ShuffleNetV2[62]	手工设计	9.4865	2.28	150.60	69.4	-
SqueezeNext[49]	手工设计	4.3680	3.37	907.32	67.5	88.2
MobileNetV1[52]	手工设计	4.3599	4.23	583.92	70.6	89.5
MobileNetV2[63]	手工设计	5.4417	3.50	320.24	71.9	91.0
NASNet-A[10]	强化学习	14.6475	5.56	620.15	74.0	91.6
DARTS[6]	梯度优化	12.2327	4.72	544.59	73.3	91
PNAS[5]	序列化 基于模型优化	16.6668	6.39	714.94	74.2	-
NSGANetV1-A1[48]	进化算法	11.5394	3.70	417.32	70.9	90.0
NSGANetV1-A2[48]	进化算法	15.0865	4.90	562.21	74.5	92.0
AmoebaNet-A[19]	进化算法	12.9763	4.63	540.56	74.5	92.0
AmoebaNet-B[19]	进化算法	9.3132	3.97	450.99	74.0	91.5
Mnasnet[11]	强化学习	5.4163	4.38	330.13	74.0	91.8
FBNet-A[16]	梯度优化	4.8936	5.20	315.37	73.0	-
FBNet-B[16]	梯度优化	6.0015	4.81	313.69	74.1	-
ChamNet-B[15]	进化算法	-	-	-	73.8	-
MLUNet-A	**进化算法**	**2.4693**	**6.71**	**399.94**	**65.8**	**86.47**
MLUNet-B	**进化算法**	**4.3595**	**8.39**	**780.89**	**71.8**	**90.38**
MLUNet-C	**进化算法**	**4.4633**	**9.98**	**962.24**	**74.2**	**91.45**

0.1%的同时快近1.5毫秒.值得注意的是,在CIFAR和ISLVRC2012上MLUNet在比NSGANet延迟快3到4倍甚至更高的同时精度相差不多甚至更优,我们认为主要的原因在于搜索空间的差异以及本文提出的启发式混合粒度交叉算子具有更强的探索能力,在ISLVRC2012上MLUNet-C更优于Mnasnet和FBNet则是由于本文提出的操作搜索空间比MBConvs搜索空间更适用于寒武纪加速卡;并且就MLUNet-C在参数量和FLOPs指标上与其他算法的对比情况来看,我们仍能得到与MLUNet-S5相同的结论.

为了更直观地展示结果的对比,本文用推理延迟和预测精度指标构建帕累托前沿的散点图,如图6所示,可以看到,在CIFAR数据集上MLUNet支配所有的对比方法,并在延迟指标上达到几乎一个数量级的下降幅度,而在ISLVRC2012数据集上,MLUNet虽然支配绝大多数对比方法,优势较前者来说略小.我们认为原因主要有以下两点:第一,搜索阶段是在CIFAR-10数据集上进行的,由于数据集图像分布、分辨率等属性的差异,网络的性能在CIFAR-10上比在ISLVRC2012上会更为突出;第二,搜索空间中网络框架的设计没有充分利用到ISLVRC2012中图像大分辨率中蕴含的特征信息,和NASNet、DARTs类似的Stem层会损失部分图像特征信息,降低了网络的预测能力.值得注意的是,在CIFAR-10上AmoebaNet系列网络的性能散点图并不是类似于MLUNet的帕累托前沿形状,其原因主要在于网络的参数量等指标并不能客观的反映其在硬件上的推理速度.

图7为MLUNet-S1到S5的网络结构可视化结果,从S1到S5搜索得到的网络精度和推理延迟逐步增加.通过分析搜索结果的算子排布规律,具有相同操作属性的算子往往排列更加紧密,比如MLUNet-S1网络将五个残差卷积(RB-k3-d1)操作排列在一起,能更好地利用寒武纪架构的流水化设计及统一的访存架构,进而在不影响精度的前提下提高模型的推理性能.此外,在高精度、高延迟的模型中,倒置残差卷积占比较大,这说明倒置残差卷积具有良好的特征提取能力,并且分组卷积会给网络的延迟带来负面影响;整体上,网络架构在第一个采样阶段的网络层数一般少于其他两个阶段,因为网络前期特征的分辨率较大,需要处理大量数据,层数过多会增加计算消耗.

综上,本文以寒武纪MLU270-F4为例,着重设

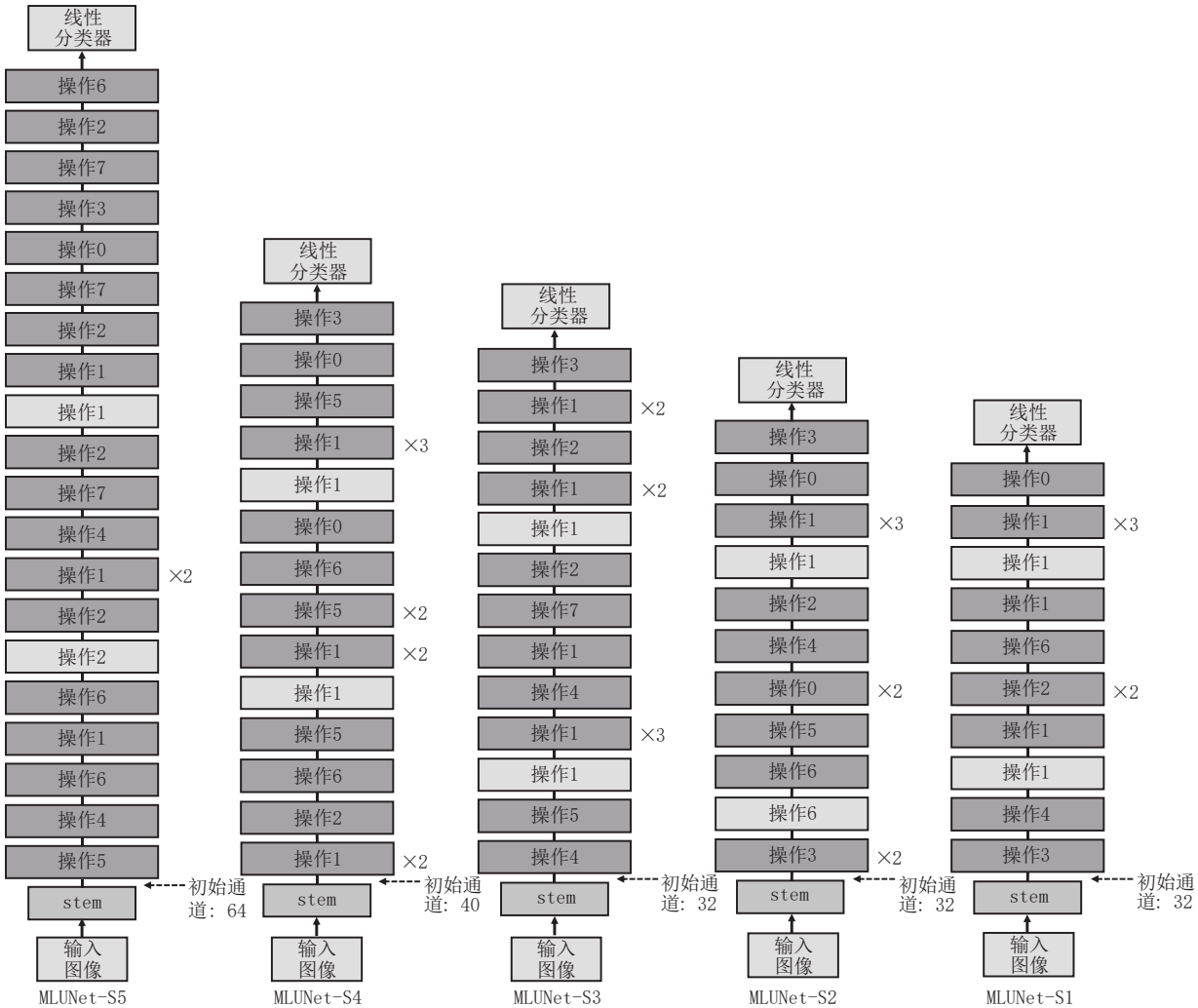

图7　CIFAR-10数据集上搜索到的MLUNet结构可视化

计了面向专用硬件加速器的神经网络结构. 首先，搜索空间算子的选择对硬件架构的推理延迟具有显著差异. 因此，借助本文提出的硬件定制化算子设计流程可以实现不同硬件架构在算子空间选择上的适配，最终有效减少搜索结构的硬件推理延迟；其次，模型的参数量和计算量指标对于最终的架构延迟并不能呈现强线性相关性. 因此，引入硬件感知的延迟预测器和多目标搜索算法能更好地指导结构搜索；最后，设计启发式的多目标混合粒度交叉算子可以提高神经网络结构搜索的收敛性和多样性.

4.3　消融实验与分析

为了探讨本文提出的方法的作用，我们分别验证了搜索空间、混合粒度交叉算子和延迟预测器的有效性.

4.3.1　硬件专用搜索空间有效性分析

本文测试了70多种操作在寒武纪加速卡MLU270-F4上的精度和推理延迟表现，其中操作算子类型包括标准卷积、空间可分离卷积、残差卷积、深度可分离卷积、空洞卷积、倒置残差卷积等，表5给出了部分操作的性能表现.

从表中可以看出，在推理延迟方面，无论是对于卷积核大小、膨胀系数还是通道扩展系数的敏感程度，英伟达2080Ti相比于寒武纪加速卡MLU270-F4都可以忽略不计，比如倒置残差卷积IRB-k5-d1-e3与IRB-k5-d2-e3，在2080Ti上的延迟变化可忽略不计，而在MLU270-F4上延迟增加了约1倍，IRB-k3-d1-e3与IRB-k3-d1-e6也可得到同样的结论.

值得注意的是，深度可分离卷积没有被选入搜索空间，相比于预测精度92.35%、寒武纪加速卡推理延迟0.8857毫秒的标准卷积CBR-k3，深度可分离卷积SepCBR-k3-d1预测精度只有89.79%，延迟为1.9319毫秒，而前者的参数量和FLOPs分别为

后者的7.1倍和6.7倍，造成这种现象的原因主要在于深度可分离卷积虽然将输入特征分为多个组并行计算减少了参数量和计算量，但减弱了其特征提取能力，分组的思想不利于特征之间的交互，并且会增加网络的内存访问代价；硬件层面上，NFU单元采用的细粒度流式处理结构可以有效地减少计算单元的输入带宽依赖，不同于英伟达GPU高并行、大批量作业模式，寒武纪加速卡为神经网络的推理添加了多种并行度，比如通道、输入映射的复用等等，保证了数据的流通，减少了与共享缓存之间的访问代价，但不同于标准卷积，处理深度可分离卷积需要计算所有分组，之后对结果进行拼接，丢弃了通道并行度，增加了计算单元对共享缓存的访问，使得处理深度可分离卷积所消耗的时间随着其分组数增加而增加. 而倒置残差卷积虽然同样进行了分组的操作，但通过通道扩展率弥补了特征提取能力，同时延迟略微提升，实验中MobileNetV2的性能也证实了部分MBConvs的可取性.

表5 部分操作算子的性能对比

操作标识	操作类型	卷积核大小	膨胀系数	通道扩展率	2080Ti延迟(ms)	MLU270-F4延迟(ms)	CIFAR-10(%)
CBR-k5	标准卷积	5	-	-	1.3461	1.1364	92.08
CBR-k13-31	空间可分离卷积	1*3+3*1	-	-	2.1330	1.0201	91.13
SepCBR-k3-d1	深度可分离卷积	3	1	-	2.2817	1.9319	89.79
SepCBR-k3-d2	深度可分离卷积	3	2	-	2.2880	2.2886	88.36
SepCBR-k5-d1	深度可分离卷积	5	1	-	2.2623	2.2967	89.54
RB-k3-d2	残差卷积	3	2	-	2.4735	1.5148	92.56
RB-k3-d12	残差卷积	3	1+2	-	2.5228	1.4924	92.60
RB-k3-d21	残差卷积	3	2+1	-	2.5463	1.5156	92.81
RB-k5-d1	残差卷积	5	1	-	2.4737	1.9735	92.96
RB-k5-d2	残差卷积	5	2	-	2.4571	2.0141	90.97
BN-k3-d1	瓶颈卷积	3	1	-	3.3706	1.5188	92.37
BN-k5-d1	瓶颈卷积	5	1	-	3.2876	1.7571	92.85
IRB-k3-d2-e3	倒置残差卷积	3	2	3	3.0445	4.7910	92.54
IRB-k5-d2-e3	倒置残差卷积	5	2	3	3.0368	9.4507	92.29
IRB-k5-d2-e6	倒置残差卷积	5	2	6	3.0589	24.8202	92.86
CBR-k3	标准卷积	3	-	-	1.3819	0.8857	92.35
RB-k3-d1	残差卷积	3	1	-	2.4136	1.4960	93.31
BN-k3-d2	瓶颈卷积	3	2	-	3.4714	2.1257	93.71
BN-k5-d2	瓶颈卷积	5	2	-	3.5261	3.0522	93.80
IRB-k3-d1-e3	倒置残差卷积	3	1	3	3.1070	3.7502	92.97
IRB-k3-d1-e6	倒置残差卷积	3	1	6	2.9874	6.0254	93.68
IRB-k5-d1-e3	倒置残差卷积	5	1	3	3.0470	4.7889	93.14

表6 基于Cell搜索空间与专用搜索空间搜索到的网络性能对比

搜索空间类别	模型名称	延迟(毫秒)	参数量(M)	FLOPs(M)	Top1(%)
基于Cell搜索空间	CNet-S6	12.0845	11.27	1695	97.61
	CNet-S5	8.8615	8.03	1179.43	97.32
	CNet-S4	10.3651	6.94	1052.45	97.26
	CNet-S3	15.5663	2.92	1306.72	96.74
	CNet-S2	13.3421	2.46	1103.84	96.58
	CNet-S1	8.3296	1.58	219.85	95.34
专用搜索空间	MLUNet-S5	5.3315	10.81	1158.14	97.66
	MLUNet-S4	3.753	4.43	505.46	97.31
	MLUNet-S3	2.3581	3.14	318.02	97.11
	MLUNet-S2	2.0733	2.6	270.96	96.85
	MLUNet-S1	1.7988	2.22	240.42	96.79

为了进一步验证专用搜索空间的有效性，我们将整个方法流程中的专用搜索空间替换为基于Cell的搜索空间[64]，其他组件及参数设置保持不变，最终搜索出的网络和基于专用搜索空间得到的网络在CIFAR-10上的对比结果如表6所示．从表中可以看出，在参数量或FLOPs接近的情况下，基于专用搜索空间得到的网络MLUNet相比于基于Cell搜索空间得到的网络CNet在延迟上都有很大的提升，前者的推理速度相比于后者提升了2倍至6倍，并且精度也有略微提升；而在精度接近的情况下，推理速度的提升甚至接近一个量级，比如MLUNet-S1与CNet-S3，进一步验证了该专用搜索空间的有效性．

4.3.2　混合粒度交叉算子有效性分析

为了探索混合粒度交叉算子的有效性，本文使用单点交叉代替混合粒度交叉算子以及寒武纪加速卡专用搜索空间进行了结构搜索实验，其他参数设置保持一致，并将最终的帕累托最优解在CIFAR-10上训练后绘制精度和推理延迟的帕累托折线图，如图8所示．

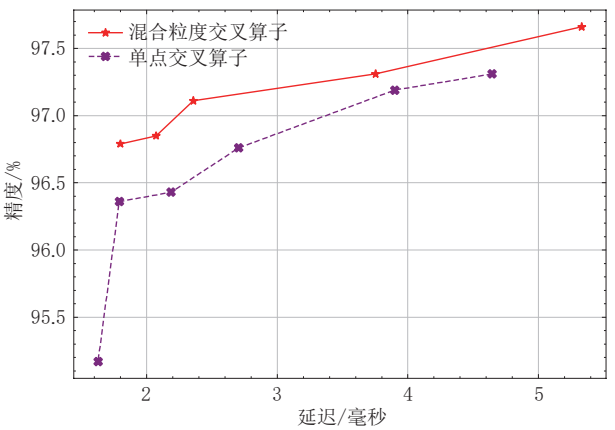

图8　混合粒度交叉算子与单点交叉算子性能对比

图中实线表示使用混合粒度交叉算子搜索到的帕累托最优解在CIFAR-10上的测试精度和寒武纪加速卡推理延迟，虚线表示单点交叉算子．从图中可以看出使用混合粒度交叉算子搜索到的网络结构性能明显支配单点交叉得到的网络效果，精度相当的情况下，推理延迟可以减少1到1.5毫秒，延迟相当的情况下，精度至多可以提高0.5%，验证了混合粒度交叉算子的有效性．

4.3.3　延迟预测器保序性分析

对于延迟预测器的保序性，本文分别采用皮尔逊相关系数(Pearson Correlation Coefficient)和肯德尔排名相关系数(Kendall Rank Correlation Coefficient)进行评估．

本文设计的延迟预测器为4层的前馈神经网络，其中隐含层包含300个节点，每层全连接之后加上ReLU激活函数增强网络的非线性，并在输出层之前将节点以0.2的概率进行dropout，增强网络的泛化能力．我们在英伟达2080Ti上将延迟预测器训练2000轮，最终在测试集上计算模型预测延迟与真实推理延迟排名的相关性，并绘制两种延迟的散点图和拟合直线，如图9所示．

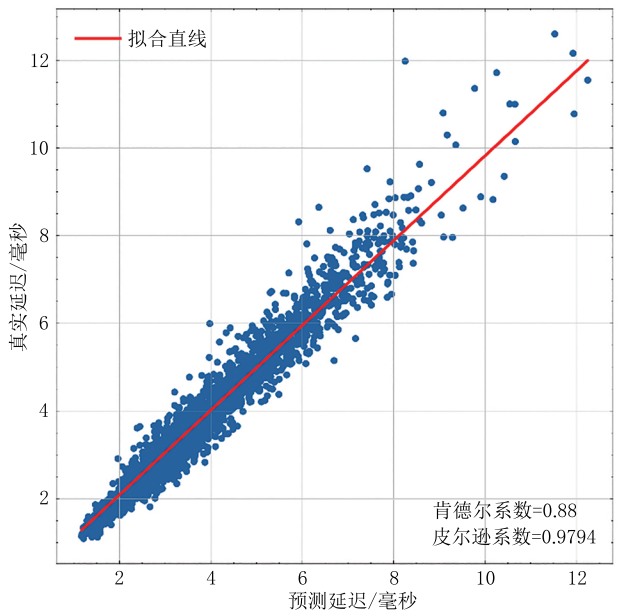

图9　真实延迟和预测延迟之间相关性

图9中横轴为预测延迟，纵轴为真实延迟．两种相关性系数如图9右下所示，我们可以看到皮尔逊相关系数可以达到约0.98，这说明预测延迟与真实延迟的变化趋势十分接近；不同于参数量和推理延迟的弱相关性，预测器的肯德尔排名相关性系数可以达到0.88，结合皮尔逊相关系数，可以大致推断预测延迟在训练时相对于真实延迟添加了一个微小的偏置，但对排名的相关性影响甚微．而从图中拟合直线可以看出，数据基本分布在$y=x$直线附近，拟合直线的斜率约为0.97，预测延迟与真实延迟具有高度相关性，可以协助搜索过程发现更好的网络结构．

5　总结与展望

由于硬件底层数据通路、运行逻辑实现的差异，将基于英伟达GPU、英特尔CPU等硬件设备搜索

到的神经网络迁移到专用AI硬件加速器上是次优的.因此,本文提出了面向硬件感知的多目标进化神经结构搜索方法,并选择国产寒武纪加速卡MLU270-F4进行了实验分析与方法验证.结合大量卷积操作在目标硬件上的性能表现,提出了硬件定制化的高效搜索空间设计方法;为解决传统交叉方式具有较大随机性并且盲目交叉导致生成优质子代概率较小的问题,使用结合启发式混合粒度交叉的NSGA-II框架,提高种群的收敛性和多样性,为神经网络加速器搜索精度和推理延迟权衡了更好的神经网络结构.在CIFAR和ISLVRC2012数据集上的实验以及消融实验验证了该搜索方法的有效性.因为进化算法在解决神经结构搜索问题上相对而言时间较长,并且神经网络在性能更加卓越的同时,模型规模愈发庞大,在算力受限的设备上部署十分困难,未来我们将考虑使用超网结合进化算法的方式进一步提高神经网络的迁移部署能力.

参 考 文 献

[1] Gao Shu-Ping, Zhao Qing-Yuan, Qi Xiao-Gang, Cheng Meng-Fei. Research on the improved image classification method of MobileNet. CAAI Transactions on Intelligent Systems, 2021, 16(1): 11-20 (in Chinese)
(高淑萍, 赵清源, 齐小刚, 程孟菲. 改进MobileNet的图像分类方法研究. 智能系统学报, 2021, 16(1): 11-20)

[2] Zhang Fu-Chang, Zhong Guo-Qiang, Mao Yu-Xu. Neural Architecture Search for Light-weight Medical Image Segmentation Network. Computer Science, 2022, 49(10): 183-190 (in Chinese)
(张福昌, 仲国强, 毛玉旭. 面向轻量化医学图像分割网络的神经结构搜索. 计算机科学, 2022, 49(10): 183-190)

[3] Wang Jun, Feng Sun-Cheng, Cheng Yong. Survey of research on lightweight neural network structures for deep learning. Computer Engineering, 2021, 47(8): 1-13 (in Chinese)
(王军, 冯孙铖, 程勇. 深度学习的轻量化神经网络结构研究综述. 计算机工程, 2021, 47(8): 1-13)

[4] Baker B, Gupta O, Naik N, et al. Designing neural network architectures using reinforcement learning. arXiv preprint arXiv: 1611.02167, 2016

[5] Liu C, Zoph B, Neumann M, et al. Progressive neural architecture search//Proceedings of the European Conference on Computer Vision. Munich, Germany, 2018: 19-34

[6] Liu H, Simonyan K, Yang Y. Darts: Differentiable architecture search//Proceedings of the International Conference on Learning Representations. New Orleans, USA, 2019:1-13

[7] Zoph B, Le Q V. Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578, 2016

[8] Yao Xiao, Shi Ye-Wei, Huo Guan-Ying, Xu Ning. Lightweight model construction based on neural architecture search. Pattern Recognition and Artificial Intelligence, 2021, 34(11): 1038-1048 (in Chinese)
(姚潇, 史叶伟, 霍冠英, 徐宁. 基于神经网络结构搜索的轻量化网络构建. 模式识别与人工智能, 2021, 34(11): 1038-1048)

[9] Li Hang-Yu, Wang Nan-Nan, Zhu Ming-Rui, Yang Xi, Gao Xin-Bo. Recent advances in neural architecture search: A survey. Journal of Software, 2022, 33(1): 129-149 (in Chinese)
(李航宇, 王楠楠, 朱明瑞, 杨曦, 高新波. 神经结构搜索的研究进展综述. 软件学报, 2022, 33(1): 129-149)

[10] Zoph B, Vasudevan V, Shlens J, et al. Learning transferable architectures for scalable image recognition//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 8697-8710

[11] Tan M, Chen B, Pang R, et al. Mnasnet: Platform-aware neural architecture search for mobile//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach, USA, 2019: 2820-2828

[12] Cai H, Zhu L, Han S. Proxylessnas: Direct neural architecture search on target task and hardware//Proceedings of the International Conference on Learning Representations. New Orleans, USA, 2019:1-13

[13] Howard A, Sandler M, Chu G, et al. Searching for mobilenetv3//Proceedings of the IEEE/CVF International Conference on Computer Vision. Seoul, Korea, 2019: 1314-1324

[14] Chu X, Zhang B, Xu R. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search//Proceedings of the IEEE/CVF International Conference on Computer Vision. Montreal, Canada, 2021: 12239-12248

[15] Dai X, Zhang P, Wu B, et al. Chamnet: Towards efficient network design through platform-aware model adaptation//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach, USA, 2019: 11398-11407

[16] Wu B, Dai X, Zhang P, et al. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach, USA, 2019: 10734-10742

[17] Xie L, Yuille A. Genetic cnn//Proceedings of the IEEE/CVF International Conference on Computer Vision. Venice, Italy, 2017: 1379-1388

[18] Real E, Moore S, Selle A, et al. Large-scale evolution of image classifiers//Proceedings of the International Conference on Machine Learning. Sydney, Australia, 2017: 2902-2911

[19] Real E, Aggarwal A, Huang Y, et al. Regularized evolution for image classifier architecture search//Proceedings of the AAAI Conference on Artificial Intelligence. Hawaii, USA, 2019, 33(01): 4780-4789

[20] Lu Z, Whalen I, Boddeti V, et al. Nsga-net: neural architecture search using multi-objective genetic algorithm//Proceedings of the Genetic and Evolutionary Computation

Conference. Prague, Czech Republic, 2019: 419-427

[21] Li X, Zhou Y, Pan Z, et al. Partial order pruning: for best speed/accuracy trade-off in neural architecture search//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach, USA, 2019: 9145-9153

[22] Guo Jia-Ming, Zhang Rui, Zhi Tian, He De-Yuan, Huang Di, Chang Ming, Zhang Xi-Shan, Guo Qi. Hardware-aware and efficient feature fusion network search. Chinese Journal of Computers, 2022, 45(11): 2420-2432 (in Chinese)
(郭家明，张蕊，支天，何得园，黄迪，常明，张曦珊，郭崎. 硬件感知的高效特征融合网络搜索. 计算机学报，2022，45(11)：2420-2432)

[23] Chen W, Wang Y, Yang S, et al. You only search once: A fast automation framework for single-stage dnn/accelerator co-design//Proceedings of the Design, Automation & Test in Europe Conference & Exhibition. Grenoble, France, 2020: 1283-1286

[24] Zhang L L, Yang Y, Jiang Y, et al. Fast hardware-aware neural architecture search//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. Seattle, USA, 2020: 692-693

[25] Loni M, Sinaei S, Zoljodi A, et al. DeepMaker: A multi-objective optimization framework for deep neural networks in embedded systems. Microprocessors and Microsystems, 2020, 73: 102989

[26] Guo Z, Zhang X, Mu H, et al. Single path one-shot neural architecture search with uniform sampling//Proceedings of the European Conference on Computer Vision. 2020: 544-560

[27] Zhang Y, Fu Y, Jiang W, et al. Dna: Differentiable network-accelerator co-search. arXiv preprint arXiv:2010.14778, 2020

[28] Wang C C, Chiu C T, Chang J Y. Efficientnet-elite: Extremely lightweight and efficient cnn models for edge devices by network candidate search. arXiv preprint arXiv:2009.07409, 2020

[29] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182-197

[30] Liu S, Du Z, Tao J, et al. Cambricon: An instruction set architecture for neural networks//Proceedings of the 2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA). Seoul, Korea, 2016: 393-405

[31] Elsken T, Metzen J H, Hutter F. Neural architecture search: A survey. The Journal of Machine Learning Research, 2019, 20(1): 1997-2017

[32] Sun Y, Xue B, Zhang M, et al. Automatically designing CNN architectures using the genetic algorithm for image classification. IEEE Transactions on Cybernetics, 2020, 50(9): 3840-3854

[33] Suganuma M, Shirakawa S, Nagao T. A genetic programming approach to designing convolutional neural network architectures//Proceedings of the Genetic and Evolutionary Computation Conference. Berlin, Germany, 2017: 497-504

[34] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Las Vegas, USA, 2016: 770-778

[35] Huang G, Liu Z, Van Der Maaten L, et al. Densely connected convolutional networks//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Hawaii, USA, 2017: 4700-4708

[36] Szegedy C, Vanhoucke V, Ioffe S, et al. Rethinking the inception architecture for computer vision//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Las Vegas, USA, 2016: 2818-2826

[37] Ying C, Klein A, Christiansen E, et al. Nas-bench-101: Towards reproducible neural architecture search//Proceedings of the International Conference on Machine Learning. Long Beach, USA, 2019: 7105-7114

[38] Dong X, Yang Y. Nas-bench-201: Extending the scope of reproducible neural architecture search//Proceedings of the International Conference on Learning Representations. New Orleans, USA, 2020:1-16

[39] Liu H, Simonyan K, Vinyals O, et al. Hierarchical representations for efficient architecture search//Proceedings of the International Conference on Learning Representations. Toulon, France, 2017:1-13

[40] Sun Y, Wang H, Xue B, et al. Surrogate-assisted evolutionary deep learning using an end-to-end random forest-based performance predictor. IEEE Transactions on Evolutionary Computation, 2019, 24(2): 350-364

[41] Suganuma M, Kobayashi M, Shirakawa S, et al. Evolution of deep convolutional neural networks using Cartesian genetic programming. Evolutionary computation, 2020, 28(1): 141-163

[42] Liu P, El Basha M D, Li Y, et al. Deep evolutionary networks with expedited genetic algorithms for medical image denoising. Medical Image Analysis, 2019, 54: 306-315

[43] Zhang H, Jin Y, Cheng R, et al. Sampled training and node inheritance for fast evolutionary neural architecture search. arXiv preprint arXiv:2003.11613, 2020

[44] Yang Z, Wang Y, Chen X, et al. Cars: Continuous evolution for efficient neural architecture search//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle, USA, 2020: 1829-1838

[45] Cai H, Gan C, Wang T, et al. Once-for-all: Train one network and specialize it for efficient deployment//Proceedings of the International Conference on Learning Representations. New Orleans, USA, 2020:1-15

[46] Luo X, Liu D, Huai S, et al. HSCoNAS: Hardware-software co-design of efficient dnns via neural architecture search//Proceedings of the 2021 Design, Automation & Test in Europe Conference & Exhibition. 2021: 418-421

[47] Yang T J, Howard A, Chen B, et al. Netadapt: Platform-aware neural network adaptation for mobile applications//Proceedings of the European Conference on Computer Vision. Munich, Germany, 2018: 285-300

[48] Lu Z, Whalen I, Dhebar Y, et al. Multiobjective evolutionary design of deep convolutional neural networks for image classification. IEEE Transactions on Evolutionary Computation, 2020, 25(2): 277-291

[49] Gholami A, Kwon K, Wu B, et al. Squeezenext: Hardware-aware neural network design//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. Salt Lake City, USA, 2018: 1638-1647

[50] Tan M, Le Q. Efficientnet: Rethinking model scaling for convolutional neural networks//Proceedings of the International Conference on Machine Learning. Long Beach, USA, 2019: 6105-6114

[51] Sun Y, Xue B, Zhang M, et al. Automatically evolving cnn architectures based on blocks. arXiv preprint arXiv: 1810.11875, 2018

[52] Howard A G, Zhu M, Chen B, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017

[53] Lu Z, Pu H, Wang F, et al. The expressive power of neural networks: A view from the width. Advances in Neural Information Processing Systems, 2017, 30:1-9

[54] Lu Z, Deb K, Goodman E, et al. Nsganetv2: Evolutionary multi-objective surrogate-assisted neural architecture search//Proceedings of the European Conference on Computer Vision. 2020: 35-51

[55] Zhang X, Zhou X, Lin M, et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 6848-6856

[56] Elsken T, Metzen J H, Hutter F. Efficient multi-objective neural architecture search via lamarckian evolution. arXiv preprint arXiv:1804.09081, 2018

[57] Zhong Z, Yan J, Liu C L. Practical network blocks design with q-learning. arXiv preprint arXiv:1708.05552, 2017, 6

[58] Pham H, Guan M, Zoph B, et al. Efficient neural architecture search via parameters sharing//Proceedings of the International Conference on Machine Learning. Stockholm, Sweden, 2018: 4095-4104

[59] Chen X, Xie L, Wu J, et al. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation//Proceedings of the IEEE/CVF International Conference on Computer Vision. Seoul, Korea, 2019: 1294-1303

[60] Xu Y, Xie L, Zhang X, et al. Pc-darts: Partial channel connections for memory-efficient architecture search//Proceedings of the International Conference on Learning Representations. New Orleans, USA, 2020:1-13

[61] Yu H, Peng H, Huang Y, et al. Cyclic differentiable architecture search. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022:211-228

[62] Ma N, Zhang X, Zheng H T, et al. Shufflenet v2: Practical guidelines for efficient cnn architecture design//Proceedings of the European Conference on Computer Vision. Munich, Germany, 2018: 116-131

[63] Sandler M, Howard A, Zhu M, et al. Mobilenetv2: Inverted residuals and linear bottlenecks//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 4510-4520

[64] Yang S, Tian Y, Xiang X, et al. Accelerating evolutionary neural architecture search via multifidelity evaluation. IEEE Transactions on Cognitive and Developmental Systems, 2022, 14(4): 1778-1792.

XU Ke, Ph. D., lecturer. His main research interests include neural network compression, neural architecture search and heterogeneous computing.

MENG Yuan, master candidate. His main research interests include deep learning, multi-objective evolutionary algorithm and neural architecture search.

YANG Shang-Shang, Ph. D., His main research interests include evolutionary multi-objective optimization, neural architecture optimization and application of graph learning in smart education.

TIAN Ye, Ph. D., associate professor. His main research interest includes evolutionary computing.

ZHANG Xing-Yi, Ph. D., professor. His main research interests include evolutionary computing and artificial intelligence.

Background

This paper focuses on the development of neural networks on hardware devices. Neural networks have excellent performance and have been widely used in many areas. However, in order to pursue higher task accuracy, the network becomes more and more complex, which makes the manual design of neural networks tedious and time-consuming. Therefore, researchers use neural architecture search to design efficient network architectures. Due to the differences in the architecture design, parallel resources and data pipeline design of hardware devices, it is not optimal to transfer and deploy networks based on NVIDIA GPU, Intel CPU and other hardware devices to domestic accelerator devices. In many domestic accelerator devices, the Cambricon accelerator can

provide data center-level computing power and achieve lower power consumption when the performance is fully utilized. Many companies have adopted Cambricon accelerators to replace NVIDIA GPUs for data center deployment due to foreign chip blockade sanctions, so it is significant to conduct neural architecture search research based on Cambricon accelerators.

In recent years, neural architecture search can be generally divided into three categories: based on reinforcement learning, based on gradient descent, and based on evolutionary algorithms; and evolutionary algorithms have strong robustness and good global optimization ability when solving multi-objective optimization problems. Based on NSGA-II, this paper designs a heuristic hybrid granularity crossover operator to improve the convergence and diversity of the population. This paper mainly solves the following two problems: (1) the networks searched by the previous neural architecture search methods cannot be well transferred and deployed to the domestic Cambricon accelerator; (2) the traditional crossover operator has large randomness, and blind forced crossover is not conducive to the generation of high-quality individuals.

This work is supported by the Key Project of Science and Technology Innovation 2030 supported by the Ministry of Science and Technology of China (2018AAA0100105), National Natural Science Foundation of China (U21A20512, 62206003).