硬件感知的高效特征融合网络搜索

郭家明[1),2),3)] 张 蕊[1),2)] 支 天[1)] 何得园[2)] 黄 迪[1),2),3)]
常 明[2)] 张曦珊[1),2)] 郭 崎[1)]

[1)](中国科学院计算技术研究所处理器芯片全国重点实验室 北京 100190)
[2)](寒武纪科技 北京 100191)
[3)](中国科学院大学 北京 100049)

摘 要 特征融合网络通过融合多尺度特征来提高目标检测精度,是深度学习目标检测框架中的关键部分.已有的研究工作通过优化融合网络的拓扑结构来提高结果精度,忽略了所需的硬件资源开销以及特征选择和特征融合操作对结果的影响.本文提出了支持多尺度特征融合的注意力感知融合网络(Attention-aware Fusion Network,AFN),通过软硬件协同可实现硬件开销(参数存储、计算时间等)敏感的神经网络自动搜索,从融合网络的特征、路径和操作三方面实现一体化的优化部署.实验结果表明,当主干网络为 ResNet50 时,在实现相似检测精度时,相比现有最先进的搜索网络 NAS-FPN,本文方法的参数量和计算量分别减少 29.6%和 22.3%,相比现有人工设计网络 FPN,本文方法的 AP 可以提高 2.1%.当主干网络为 VGG 时,相比现有最先进的搜索网络 Auto-FPN,本文方法的 AP 提高了 1.7%.

关键词 目标检测;神经结构搜索;硬件开销
中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2022.02420

Hardware-Aware and Efficient Feature Fusion Network Search

GUO Jia-Ming[1),2),3)] ZHANG Rui[1),2)] ZHI Tian[1)] HE De-Yuan[2)] HUANG Di[1),2),3)]
CHANG Ming[2)] ZHANG Xi-Shan[1),2)] GUO Qi[1)]

[1)](*State Key Laboratory of Processors, Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190)
[2)](*Cambricon Technologies, Beijing* 100191)
[3)](*University of Chinese Academy of Sciences, Beijing* 100049)

Abstract Fusion network is a representative module in object detection frameworks to fuse multi-scale features and improve detection accuracy. Previous works of designing fusion network architecture mainly focus on designing the topology of the fusion path to improve the performance of object detection. However, the required hardware resource overhead and the influence of feature selection and feature fusion operations on the detection performance are ignored. In this paper, we propose a feature fusion network named Attention-aware Fusion Network (AFN), which has a strong capacity of fusing multi-scale features for object detection. Through software and hardware cooperation, it can realize the automatic search of the neural network sensitive to hardware cost (parameter storage, calculation time, etc.), and realize the integrated optimization deployment from the three aspects of the fusion network's characteristics, paths and operations. In this paper, we first summarize and propose three key factors that should be considered in the design of

收稿日期:2021-12-28;在线发布日期:2022-05-23.本课题得到国家重点研发计划(2017YFA0700900)、国家自然科学基金(61925208,62102399,61906179,61732020,U20A20227)、中国科学院战略性先导科技专项(XDB32050200)、中国科学院稳定支持基础研究领域青年团队计划(YSBR-029)和科学探索奖资助.**郭家明**,博士研究生,主要研究方向为深度学习.E-mail:guojiaming18s@ict.ac.cn.**张 蕊**,博士,副研究员,主要研究方向为深度学习、强化学习.**支 天**,博士,高级工程师,主要研究方向为集成电路设计、可重构计算.**何得园**,硕士,主要研究方向为深度学习.**黄 迪**,博士研究生,主要研究方向为深度学习.**常 明**,硕士,主要研究方向为深度学习.**张曦珊**,博士,副研究员,主要研究方向为深度学习.**郭 崎**(通信作者),博士,研究员,主要研究领域为智能计算系统.E-mail:guoqi@ict.ac.cn.

feature fusion network: fusion feature selection, fusion path and fusion mode. We also need to consider the hardware overhead of deploying the algorithm to the target platform. However, these design factors compose a huge design space that contains tremendous amounts of design choices. Thus, manually designing the optimal architecture of the fusion neck is very difficult. We employ neural network search (NAS) method to automatically design the feature fusion network. We propose three kind of search unit: feature search unit, fusion path search unit and fusion mode search unit. The feature search unit aims to search for the most appropriate input features for each scale instead of fixing from the top layer of each stage. The fusion path search unit takes all possible cross-scale fusing connections among groups as the search space and search for the optimal connection. the fusion mode search unit contains a variety of candidate fusion operations and decide operations to fuse features of multiple scales. Particularly, this unit is attention-aware by utilizing different kinds of attention mechanisms along with the commonly used add operation as candidate fusion operations. We use NAS algorithm based on evolutionary algorithm, and realize weight reuse and grouping fusion when designing search unit, which reduce the computational cost and memory cost. We also take the hardware cost of feature fusion network on the target hardware as the search target, so we can achieve a good trade-off between precision and computational cost on the target hardware. We evaluate our method on the famous detection dataset COCO and compare the searched feature fusion network with several advanced feature fusion network and give the results of detection precision and the network complexity. We also carry out experiments to evaluate our special design choice of the search units and the search object. The experiment results shows that when the backbone is ResNet50, compared with the existing searched network NAS-FPN, the parameter amount and calculation amount are reduced by 29.6% and 22.3% respectively when achieving similar detection accuracy. Compared with the existing artificially designed network FPN, AP can increase by 2.1%. When the backbone network is VGG, compared to the searched network Auto-FPN, AP can increase by 1.7%.

Keywords　object detection; neural architecture search; hardware overhead

1　引　言

最近，基于卷积神经网络(CNN)的目标检测框架在准确性和效率方面都取得了显著进步[1-5]. 目标检测框架通常由提取特征的主干网络、融合特征的特征融合网络和用于分类和定位的头部网络组成. 由于目标检测需要同时进行识别和定位，直接使用主干网络顶层的低分辨率特征不足以准确定位小目标. 因此，最近的研究工作[3-4,6-7]设计了先进的特征融合网络来融合多尺度特征，增强了定位和识别的能力.

特征融合网络设计的前期工作重点研究多尺度特征的融合路径，包括通过手工设计[6-8]或神经结构搜索(NAS)[9-10]. 然而，不同层次的特征应该与不同大小的物体相关联，所以常用的加法操作作为融合操作可能会导致不同层次特征之间的冲突，并损害融合网络的性能[11]. 一些工作[11-12]采用注意力机制自适应地融合多尺度特征. 尽管它们提高了检测性能，但决定如何产生注意力权重以及在什么维度应用注意力机制具有挑战性，人工设计的单一注意力模式并未充分利用注意力机制的潜力. 此外，现有融合网络的输入特征固定为主干网络中特定的层，比如，主干网络每个阶段的最后一层. 然而，他们忽略了从其他层选择特征可能会获得更好的性能. 在这项工作中，我们专注于从更高的角度设计融合网络的架构. 为了提高融合多尺度特征的能力，不同于现有的只考虑融合路径的工作，我们考虑了三个方面：输入融合特征、融合路径和融合操作.

接下来，我们对这三个方面进行分析：

(1) 现有的融合网络的输入特征固定为主干网络每个阶段的最后一层. 这一般是每个阶段中语义信息最强，空间信息最模糊的层，我们选择其他层可以在语义信息和空间信息之间实现更好的权衡；

(2)不同的拓扑结构(例如,自顶向下或自底向上)可能表现出语义信息强或空间信息强的不同属性,也可能影响特征融合的有效性;

(3)除了常用的相加融合操作之外,还有很多不同种类的注意力机制.注意力机制使用不同的特征来生成权重并沿不同维度应用权重,将在语义或空间的不同方面表现出融合能力.

我们在考虑以上设计因素的同时,还要考虑到算法部署到目标平台时的硬件开销.目前,目标检测算法广泛应用于安防监控、自动驾驶、医疗影像处理、机器人等各个场景.为了保证算法可以在目标硬件上高效运行,我们需要在设计算法时考虑目标平台的硬件特性.然而,要在以上巨大的设计空间中实现精度、计算能效.参数量之间的良好权衡无疑非常困难.因此,我们使用了神经结构搜索算法,它在自动有效地在搜索空间中发现最佳网络方面有显著的效果.

基于上述分析,我们提出了一种新的网络结构搜索框架来设计硬件感知的注意力感知融合网络(Attention-aware Fusion Network,AFN).我们的贡献主要包括以下三个方面:

(1)我们总结并提出了设计特征融合网络要同时考虑的三个关键因素:融合特征选择、融合路径、融合模式;

(2)我们使用了神经网络搜索算法并根据上述关键因素设计三个搜索单元:融合特征搜索单元、融合路径搜索单元和融合模式搜索单元.我们采用了基于进化算法的神经网络搜索算法,并在设计搜索单元时实现了权重复用和分组融合,减小了搜索时的计算开销和内存开销;

(3)我们把在目标硬件上特征融合网络的硬件开销作为搜索目标,因此,在目标硬件上能实现精度和计算开销的良好权衡.

2 相关工作

目标检测.基于卷积神经网络的目标检测框架已经取得了显着的成功,例如,R-CNN[5]、Fast R-CNN[1]、Faster R-CNN[2]、SSD[3]和 RetinaNet[4].现代目标检测系统通常由三部分组成:从图像中提取多尺度特征的主干网络、融合提取特征的特征融合网络和用于分类和定位的头部网络.其中特征融合网络是解决物体检测中多尺度问题的基础模块.许多工作通过精心设计融合网络的结构获得了性能的提升.FPN[9]构建了一个自顶向下的结构,横向连接到主干网络的不同阶段,从而融合所有尺度的特征.PANet[8]相比 FPN 增加了额外的自底向上的连接结构.Bi-FPN[7]将自顶向下和自底向上的特征金字塔堆叠多次,并且在融合多尺度特征时增加了额外的参数对特征进行加权.现有的特征融合网络工作主要设计连接的拓扑来更好的融合特征.不同的是,我们所提出的方法通过考虑三个关键因素从更高的角度设计特征融合网络:融合特征选择、融合路径、融合操作.

注意力机制.受人类视觉系统处理特定物体而不是整个场景的启发,研究人员结合注意力机制来提高卷积神经网络的性能.注意力机制旨在根据特征的重要性自适应地增强特征,并在许多任务中带来较大提升.一些方法[13-15]利用注意力机制来提高图像分类的性能.最近,注意力机制被应用于目标检测的特征融合网络中.ASFF[11]利用空间维注意力对不同位置的特征加权来实现更好的融合.NETNet[12]使用空间维注意力对干扰特征进行擦除来避免不同尺度特征冲突.NAS-FPN[9]在融合特征时利用注意力机制对特征的不同通道进行加权.Non-Local[16]操作使用自注意力机制使新特征每个位置都可以是原特征所有位置的加权,捕获了空间上的长距离依赖关系.Libra-RCNN[17]在 FPN 的基础上使用 Non-Local 操作对融合后的特征进行增强.与它们不同的是,我们考虑结合不同类型的注意力机制,并通过网络结构搜索的方法对他们进行合理组合.在消融实验中,我们发现对注意力机制的合理组合相比单一注意力机制能实现更高的精度.

神经结构搜索.神经结构搜索(NAS)旨在为特定任务和数据集自动设计神经网络.NAS 方法已广泛应用于图像分类.一些工作[18-19]通过训练从搜索空间采样的所有网络来搜索最佳架构,并通过使用强化学习或进化方法等样本策略来加快过程.最近,一些网络结构搜索方法[20-22]通过训练一个包含所有架构的超参数化网络来减少所需的计算资源.在这项工作中,我们同样采用这种方法.NAS 方法也被应用于寻找用于目标检测的网络结构.DetNAS[23]在目标检测框架中搜索主干网络的结构.NAS-FPN[9]和 Auto-FPN[10]主要搜索特征融合网络的融合路径.SM-NAS[24]通过进化方法搜索目标检测框架中的所有模块.与它们相比,我们提出的 AFN 侧重于从精心设计的搜索空间中有效地搜索特征融合网络,包括搜索输入特征、融合路径和注意力感知

融合操作.

硬件感知的神经网络设计. 考虑到某些硬件平台有限的计算资源，一些工作希望设计轻量高效的神经网络来计算资源受限情况下算法的实时运行. 一些工作利用专家经验人工设计轻量级神经网络. MobileNet[25]和 MobileNetV2[26]通过使用深度可分离卷积减小神经网络的计算和存储开销开销，大大加快了网络在 CPU 上的运行效率. ShuffleNet[27]使用分组卷积和通道随机排列来实现减小计算量的同时尽量不损失精度. 也有一些研究如 Yolo[28]、ThunderNet[29]设计了轻量级的目标检测算法实现在计算资源受限的终端上的高效检测. 另一些方法采用神经网络搜索算法设计在目标硬件上高效的网络. 对于这些工作，一个重要的问题是如何估计神经网络在目标硬件上的延时. MnasNet[30]使用实时测试候选网络的延时. ChamNet[31]、FBNet[32]、NetAdapt[33]、DFNet[34]等则使用查找表对网络的延迟进行估计. 这些工作主要针对通用物体识别网络，或者是目标检测算法中的主干网络. 与上述方法不同的是，我们使用神经结构搜索算法设计硬件感知的特征融合网络结构，并采用神经网络预测候选结构的延迟.

3　方　法

在这项工作中，我们使用基于进化算法的神经结构搜索方法来解决网络结构设计问题. 在本节中，我们首先介绍 AFN 的结构和搜索空间. 然后我们描述如何根据目标硬件平台的特性搜索硬件感知的特征融合网络.

3.1　基于注意力的特征融合网络

我们提出的 AFN 旨在融合多尺度特征并提高目标检测框架的分类和定位能力. 图 1 显示了我们提出的网络搜索框架的整体结构. AFN 将来自主干网络的特征作为输入，生成融合特征. AFN 是可扩展的，可以堆叠在任何主干网络和检测头网络之间.

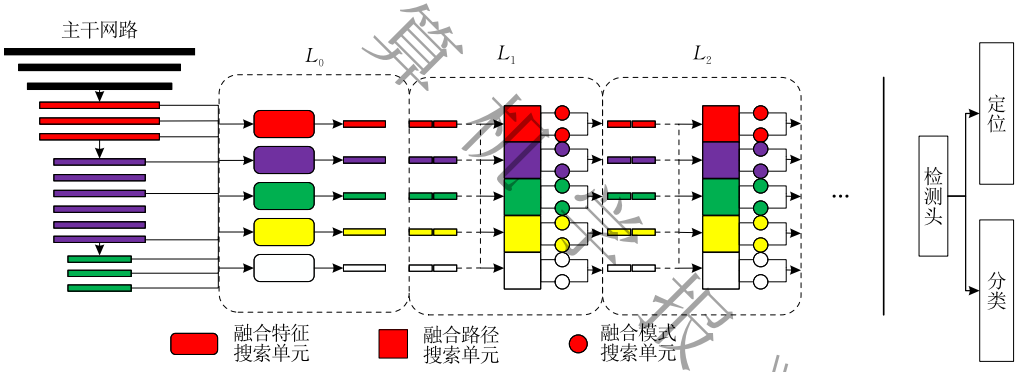

图 1　融合网络搜索框架的整体结构(它由融合特征搜索单元、融合路径搜索单元和融合模式搜索单元这三个搜索单元组成)

为了搜索网络结构，AFN 由三种单元组成：融合特征搜索单元、融合路径搜索单元和融合模式搜索单元. 这三种单元构成了 AFN 中的层. 将整个 AFN 结构记为 S_{AFN}，假设 AFN 包含 $T+1$ 层：$S_{\mathrm{AFN}}=\{L_0,L_1,\cdots,L_T\}$，每层产生 N 种不同尺度的特征：$L_i=\{F_1^i,F_2^i,\cdots,F_N^i\}$.

第一层 L_0 将来自主干网络的 M 个候选特征作为输入，并针对不同的尺度输出最合适的 N 个特征. 我们通过融合特征搜索单元搜索 L_0 的结构.

此外，接下来的 T 层 $L_i(i=1,\cdots,T)$ 融合了从前一层 L_{i-1} 产生的 N 个特征并保持特征数量不变.

为了提高融合多尺度特征的能力，L_i 的结构由融合路径搜索单元和融合模式搜索单元决定. 融合路径搜索单元选择最优的跨尺度连接，同时融合模式搜索单元确定多个注意力机制中的最优融合操作. 通过这三种基本单元，我们将整个搜索空间划分为可单独改变的一系列选择，这样我们在搜索空间内的不同的网络结构之间实现了权重共享，从而减小了搜索时的计算开销和存储开销. 接下来，我们将详细描述这三种基本单元.

3.1.1　融合特征搜索单元

现有的特征融合网络通常从主干网络每个阶段的最后一层选择输入融合特征. 然而，最后一层的特征往往语义信息强而空间信息模糊，而随着层数减少，卷积感受野减小，空间信息保留更多，而语义信息逐渐减弱. 我们认为在选择输入特征时就考虑空间信息和语义信息的权衡将实现更好的融合效率. 因此，融合特征搜索单元旨在从主干的不同层中选择多尺度的最佳特征.

通常，我们假设生成具有相同空间大小的特征图的层处于同一网络阶段. 为了提供更多的选择，我

们从每个阶段的第一层、中间层和最后一层中选择特征作为候选特征.这三种层代表了每个阶段的空间信息最强、语义信息和空间信息平衡和语义信息最强的特征.将来自第 i 个主干阶段的特征表示为 $P_i^c=\{P_i^0,P_i^{\left\lfloor\frac{n_i-1}{2}\right\rfloor},P_i^{n_i-1}\}$,其中 n_i 为第 i 阶段的层数,上标表示层的索引.

这里,我们以 ResNet-50[35] 为例,从 3～5 阶段中选择特征作为输入,因此候选特征可以表示为 $P^{\text{candidate}}=\{P_3^c,P_4^c,P_5^c\}$.假设 $P^{\text{candidate}}$ 中总共有 M 个候选特征,它是 AFN 第一层 L_0 的输入.L_0 产生 N 特征 $L^0=\{F_1^0,F_2^0,\cdots,F_N^0\}$ 对应于 N 不同的尺度.空间大小从 F_1^0 到 F_N^0 以因子 2 逐渐下采样.尺度 s 的每个输出特征 $F_s^0(s=1,\cdots,N)$ 使用融合特征搜索单元 U_s^f 选择,该单元应用选择操作 $O^f\in\mathcal{O}_C^f$ 对候选特征 $P^{\text{candidate}}$,表示为 $U_s^f:F_s^0=O^f(P^{\text{candidate}})$.

为了生成具有更丰富多尺度信息的特征,选择操作要么选择一个候选特征,要么选择其中两个特征的和(特征将首先下采样或上采样为目标大小).

对于 M 候选特征,可能的操作数为 $C=\binom{m}{2}+M$.

最后,选定的特征后面总是跟着一个 3×3 卷积、一个批归一化层(Batch normalization layer)和一个 ReLU 激活函数.

3.1.2 融合模式搜索单元

融合模式搜索单元决定融合多尺度特征的操作.除了常用的加法运算外,还有很多不同的注意力机制在不同方面表现出很强的融合能力.因此,我们将不同种类的注意力机制与加法操作一起视为候选融合操作,表示为 $A=\{A_1,A_2,\cdots,A_Q\}$,其中 Q 是融合操作的数量.融合模式搜索单元的整体结构如图 2 所示.

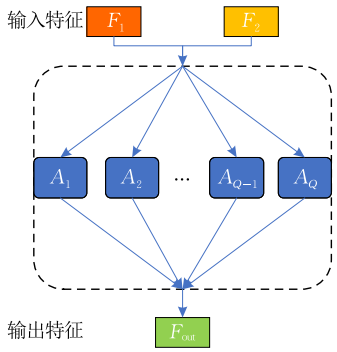

图 2 融合模式搜索单元

为了充分发挥注意力机制的潜力,我们考虑了两个关键方面来设计候选注意力机制:

(1)使用哪些特征生成权重,特征本身还是包含更多空间或语义信息的另一个尺度的特征?

(2)在哪个维度应用注意力机制,更适合语义融合的通道维度还是更适合空间信息融合的空间维度?

我们了构建以下 5 种融合操作.这些融合操作涵盖了应用注意力机制的所有维度选择,以及用于生成注意力权重的所有特征选择.这些融合操作的细节如图 3 所示.

①空间维自注意力(*Spatial-wise self-attention*).受 ASFF 的启发,这个操作使用 1×1 卷积计算两个特征的注意力权重,并且在这两个权重之间应用 Softmax 函数.这两个注意力权重与特征具有相同的空间大小,但它们的通道数都是 1.然后将特征与相应的注意力权重相乘并相加.

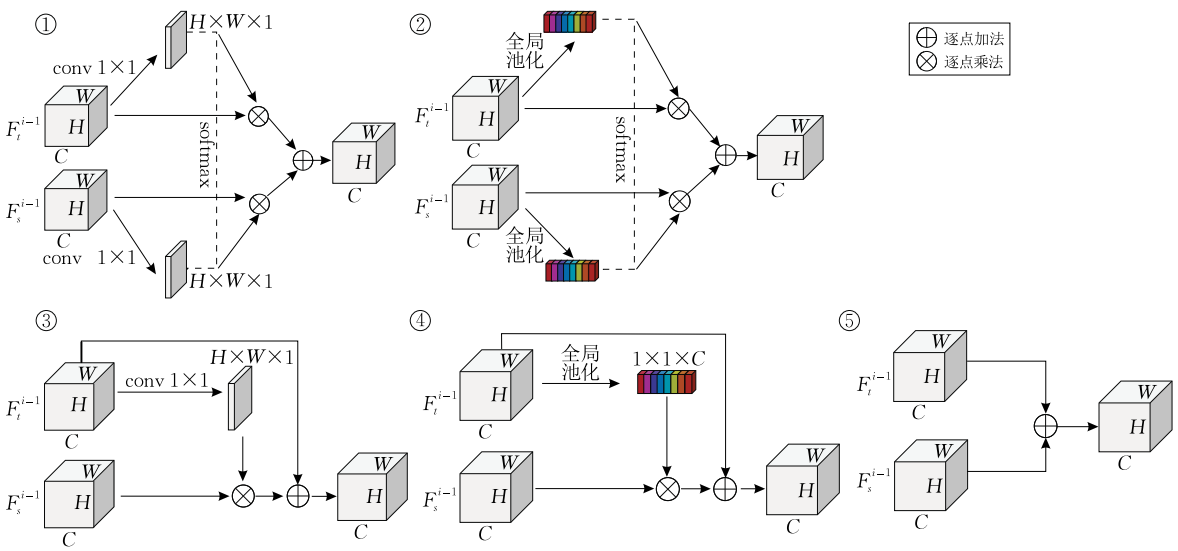

图 3 融合模式搜索单元中用于融合特征 F_s^{i-1} 和 F_t^{i-1} 的候选融合操作(F_s^{i-1} 是和输出特征有相同尺度的特征.F_t^{i-1} 从其他尺度上采样或下采样得到的特征.对于第③种和第④种操作,输入特征的位置可以被交换)

②通道维自注意力(*Channel-wise self-attention*). 这个操作与空间维自注意力(*Spatial-wise self-attention*)类似. 唯一的区别是注意力权重是通过全局池化生成的，因此它们具有与特征相同的通道数，但它们的空间大小为 1.

③空间维注意力(*Spatial-wise attention*). 在这个操作中，一个特征将乘以一个注意力图，这个图是使用另一个特征经过 1×1 卷积计算出来的. 注意力图与特征具有相同的空间大小，但通道数为 1. 最后将两个特征相加.

④通道维注意力(*Channel-wise attention*). 这个操作与空间维注意力(*Spatial-wise attention*)类似. 不同之处在于注意力权重是通过全局池化生成的，并且具有与特征相同的通道数，但注意力权重的空间大小为 1.

⑤加法(*Add*). 这个操作会直接将两个特征相加.

在层 L_i 中具有输出尺度 s 的融合操作 A_s 需要特征 F_s^{i-1} 和 F_t^{i-1}，$t\neq s$ 作为输入.

融合操作 A_s 是通过一个 n 选 1 操作 $O^a\in\mathcal{O}_Q^a$ 得到的.

那么，层 L_i 中尺度 s 对应的融合模式搜索单元 U_s^a 可以表示为 U_s^a: $A_s=O^a(\{A_j\}_{j=1}^Q)$.

3.1.3　融合路径搜索单元

在融合跨尺度特征的过程中，不同的融合路径可能表现出语义信息强或空间信息强的不同属性. 因此，我们设计了融合路径搜索单元来决定应该融合前一层的哪些特征.

融合路径搜索单元的设计有两个关键点：

(1) 分析现有特征融合网络的各种结构[6-8]，我们注意到一个设计原则，即融合操作的输入总是包括与输出特征具有相同尺度的前一层的特征. 该原则通过保留当前尺度的信息并从另一个尺度的特征聚合跨尺度信息来提高融合效率；

(2) 如融合模式搜索单元中所述，我们使用的融合操作都是双输入操作. 因此，为了在减少计算开销的同时融合更多尺度，我们在融合之前将特征通道分为几组. 基于上述几点，我们首先将每个尺度特征的通道划分为 K 组. 那么对于 L_i 层，$k(k=1,\cdots,K)$ 组中尺度为 s 的特征可以表示为 $F_{s,k}^{i-1}$. 融合路径搜索单元 U_s^p 采用 n 选 1 的操作 $O^p\in\mathcal{O}_{(N-1)*K}^p$ 来选择不同尺度的特征 $F_{t,k}^{i-1}$，$t\neq s$. 然后特征 $F_{s,k}^{i-1}$ 和特征 $F_{t,k}^{i-1}$ 被融合模式搜索单元融合. 为了融合相同尺度的输出特征，我们加上了一个融合卷积. 图 4 表示了将通道分成两组时的融合过程，其中不同颜色的方块代表了不同尺度的特征.

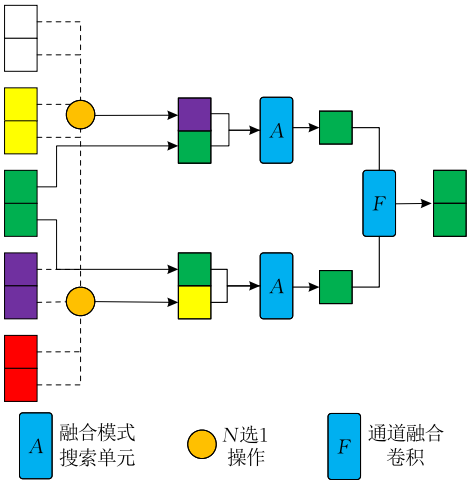

图 4　特征维度被划分为两组时的融合过程

之前的工作如 NAS-FPN[9] 和 Auto-FPN[10] 旨在寻找融合路径. 因此，它们直接融合了不同尺度的整个特征. 不同的是，我们提出的融合路径搜索单元对特征通道进行分组并限制搜索空间，这可以降低搜索和部署时计算复杂度并有效且高效地获得跨尺度特征.

3.2　搜索算法

我们采用了基于进化算法的神经结构搜索方法在上述搜索空间中找到最优结构. 如上所述，三种单元构成了整个搜索空间. 每个单元包含一个 N 选 1 的选择操作，所有这些操作形成一个路径，即一种网络结构.

由于所有这些路径共享相同的参数并组成一个超网络，我们可以训练超网络并把它作为我们搜索空间中所有网络结构(路径)的性能估计器. 受 GreedyNAS[22] 的启发，我们采用多路径采样策略，并贪心地过滤比较差的路径来训练超网络. 当超网络训练完成后，我们将超网络的性能作为评分函数的一部分，并执行进化算法以获得最优网络结构. 接下来我们将详细介绍搜索过程.

参考 DARTS[20] 算法，我们将原始训练数据划分为两个不相交的集合：训练集 D_{train} 和验证集 D_{var}. 在我们的搜索过程中，训练集 D_{train} 用于训练超网络的参数，验证 D_{val} 用于验证超网络的性能. 我们为每个类别的对象随机选择 20 张图像构成验证集 D_{val}，其余图像构成了训练集 D_{train}.

我们的搜索过程分为三个阶段：预训练阶段、贪心训练阶段和进化算法阶段.

在预热阶段,搜索单元中的所有操作都被均匀采样并使用训练集 D_{train} 训练. 预训练阶段对整个搜索过程非常重要. 在搜索过程开始时,所有操作的权重都没有得到充分训练. 如果在这时开始贪心训练,那么收敛更快而不是性能更好的操作将在贪心训练中更具有优势,这样,搜索就会偏离最优值. 在预训练阶段之后,所有的操作都应该得到充分的训练.

在贪心训练阶段,我们采用 GreedyNAS[22] 方法. 在训练过程中,我们保留一个大小为 $|\mathcal{P}|$ 的候选池 $\mathcal{P}$. 在每次训练迭代中,我们以 ϵ 的概率从候选池中采样 m 条路径,并以 $1-\epsilon$ 的概率从整个搜索空间中采样. 这 m 个路径将在 D_{val} 上进行测试. 结果将按目标函数 F_{greedy} 排序. 然后将前 k 条路径并用于更新候选池 $\mathcal{P}$. 当候选池已满时,将丢弃 F_{greedy} 值最低的路径.

在贪心训练阶段完成后,我们获得了一个超网络作为不同网络结构的性能估计器. 然后我们以候选池中的前 n 条路径作为初始种群执行进化算法. 我们通过使用目标函数 $F_{\text{evolution}}$ 来评估网络结构. 在这些评估的架构中,我们选择 top-n 作为父代来生成子网络. 其中子网络由变异和交叉各产生一半网络. 我们在迭代中重复这个过程,神经结构搜索结果是最后一个种群中目标函数得分最高的网络结构.

对于所有实验,我们将候选池的大小 $|\mathcal{P}|$ 设置为 500,样本路径数为 10,所选路径数为 5. 在贪心训练过程中,我们将池采样概率从 0 线性提高到 0.8. 预热阶段和贪心训练阶段伪代码如算法 1 所示. 使用进化算法获得网络结构的伪代码如算法 2 所示.

算法 1. 超网络训练.

输入:超网络 S 以及其参数 w,训练数据 D_{train},验证数据 D_{val},采样路径数 m,保留路径数 k,最大迭代次数 T,预热阶段迭代次数 T_{warm},目标池采样概率 p_{end}

初始化候选池 $\mathcal{P}=\{\}$,池采样概率 $p=0$,

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

  IF $i\leq T_{\text{warm}}$ THEN

    从整个搜索空间均匀采样 k 个路径 $\{path_j\}_{j=1}^{k}$

    FOR $j=1,\cdots,k$ DO

      从 D_{train} 中采样一批数据 d_{train}

      在 d_{train} 上计算 $path_j$ 对应的损失函数 L_{train}^{j}

      使用梯度更新 $path_j$ 对应的权重

  ELSE

    $p=i/(T-T_{\text{warm}})*p_{\text{end}}$

    以 p 的概率从候选池 $\mathcal{P}$ 中采样,以 $(1-p)$ 的概率从整个搜索空间采样,一共得到 m 条路径

    从 D_{var} 采样一批数据 d_{var}

    在 d_{var} 计算每个路径对应的 F_{greedy} 并排序

    选出 top-k 路径 $\{path_j\}_{j=1}^{k}$

    使用 $\{path_j\}_{j=1}^{k}$ 更新候选池 $\mathcal{P}$

    FOR $j=1,\cdots,k$ DO

      从 D_{train} 中采样一批数据 d_{train}

      在 d_{train} 上计算 $path_j$ 对应的损失函数 L_{train}^{j}

      使用梯度更新 $path_j$ 对应的权重

输出:超网络 S 以及其参数 w,候选池 $\mathcal{P}$

算法 2. 进化算法.

输入:超网络 S 以及其参数 w,验证数据 D_{val},候选池 $\mathcal{P}$,最大迭代次数 T,种群数量 N,初始种群数量 N_0

从候选池 $\mathcal{P}$ 中选出 top-N_0 作为初始种群 $\{path_i\}_{i=1}^{N_0}$

FOR $i=1,\cdots,N_0$ DO

  在 D_{val} 上计算 $path_i$ 对应目标函数 $F_{\text{evolution}}$

根据 $F_{\text{evolution}}$ 选出 $\{path_i\}_{i=1}^{N_0}$ 中的 top-N:$Pop=\{path_i\}_{i=1}^{N}$

FOR $t=1,\cdots,T$ DO

  $count=0$

  WHILE $count<N/2$ DO

    从 Pop 中随机采两条路径并交叉得到 $path_{\text{crossover}}$

    $Pop=Pop\bigcup\{path_{\text{crossover}}\}$

    IF $path_{\text{crossover}}\notin Pop$ DO

      $count+=1$

  $count=0$

  WHILE $count<N/2$ DO

    从 Pop 中随机一条路径并变异得到 $path_{\text{mutation}}$

    $Pop=Pop\bigcup\{path_{\text{crossover}}\}$

    IF $path_{\text{crossover}}\notin Pop$ DO

      $count+=1$

  在 D_{val} 上计算 Pop 中所有路径的 $F_{\text{evolution}}$

  选取 top-N 作为新的 Pop

输出:Pop 中 $F_{\text{evolution}}$ 最高的路径

3.3 硬件感知的搜索目标

3.3.1 硬件感知的目标函数

为了实现搜索得到的网络结构实现硬件开销和检测准确率的良好权衡,我们需要将硬件开销加入我们的搜索目标. 如 3.2 节所述,在搜索过程中,我们使用目标函数来表示搜索目标并对网络结构进行排序. 因此,我们设计的目标函数同时考虑精度和目标硬件上网络的存储和计算开销. 在贪心训练超网络时目标函数为

$$F_{\text{greedy}}=-L_{\text{val}}-\alpha_t T(Arch)-\alpha_M M(Arch),$$

其中 L_{val} 为网络在 D_{val} 上的平均损失函数,$T(Arch)$ 为网络在目标硬件上的计算时间开销,$M(Arch)$ 为网络参数的存储开销.

在执行进化算法时目标函数为

$$F_{\text{evolution}}=mAP_{\text{val}}-\alpha_t T(Arch)-\alpha_M M(Arch),$$

其中 mAP_{val} 为函数在网络在 D_{val} 的平均检测精度.

3.3.2　时间开销预测

上述目标函数中，平均损失函数 L_{val}，平均检测精度 mAP_{val} 可以在搜索过程中得到，参数存储开销也可以直接计算得到，而对于计算时间开销 $T(Arch)$ 我们很难直接得到. 虽然我们能直接得到网络结构对应的计算量，但在 GPU 或者神经处理芯片上计算量和计算时间往往不是线性相关. 因此，我们训练了预测神经网络 P 用于估计网络的计算时间开销 $T(Arch)$. 神经网络 P 的网络结构如图 5 所示.

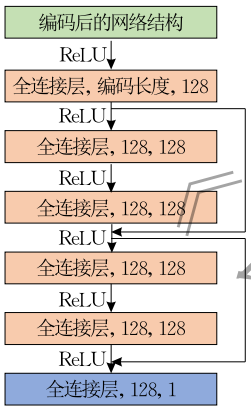

图 5　时间开销预测网络的网络结构

我们在搜索前从搜索空间中随机采样 N 个网络结构，分别将这些网络结构在目标硬件平台上运行 M 次并记录运行时间，取 M 次的平均值记录下来，作为训练数据. 然后使用采集到的数据训练神经网络 P. 损失函数为

$$L_{\mathrm{P}}=\frac{1}{N}\sum_{i=1}^{N}\left(\mathrm{P}(Arch_i)-\frac{1}{M}\sum_{j=1}^{M}Latency_j(Arch_i)\right)^2.$$

在搜索过程中，我们可以直接使用训练收敛的网络 P 的输出估计计算时间开销 $T(Arch)$.

4　实　验

4.1　数据集

在本文中，我们使用 COCO 数据集[36]进行实验. COCO 数据集是通用目标检测任务最为常用的数据集，它包含 80 种不同物体. 我们使用它的 2017 版本，它的训练集 train2017 集合中包含 118K 图像，验证集 val2017 中包含 5K 图像（又名 minival）. 在搜索过程中，我们从 train2017 数据集中每类随机抽取 20 张图像来构建搜索验证数据集. 在重训练期间，搜索到的模型使用整个 train2017 数据集进行训练，并使用 minival 进行评估.

4.2　实施细节

我们检测模型中使用的每个主干网络，都使用在 Imagenet 上预训练的权重用于初始化.

我们硬件感知的融合网络搜索算法是使用 PyTorch[37-38]开发的. 我们在一台装有 8 个 NVIDIA V100 GPU 的机器上进行实验. 在搜索和训练期间，我们使用[0.8,1.2]之间的随机尺度对训练数据进行缩放，批大小为 8. 对于焦点损失（Focal loss）的超参数我们使用 $\alpha=0.25$ 和 $\gamma=1.5$. 我们主要使用 RetinaNet[4]的开源实现进行实验，并使用与 NAS-FPN[9]相同的超参数进行训练. 对于我们的特征融合网络 AFN，每个特征的输出通道数为 256.

对于所有的融合路径搜索单元，我们将组数设置为 2. 对于主干网络是轻量级网络如 mobile-net 的融合网络，我们将网络在 CPU 上的运行时间作为目标函数的一部分，对于其他网络在 GPU 上的运行时间作为目标函数的一部分. 如果要应用于其他硬件，我们的目标函数可以做简单的替换. 如图 6 所示，我们在 Nvidia Tesla T4 随机选取候选网络结构并测试运行延时，真实测试的延时和网络预测的延时在测试集上有很强的相关性，因此我们使用预测网络输出作为搜索的目标函数的一部分能很好的实现对网络在目标硬件上计算延时的约束.

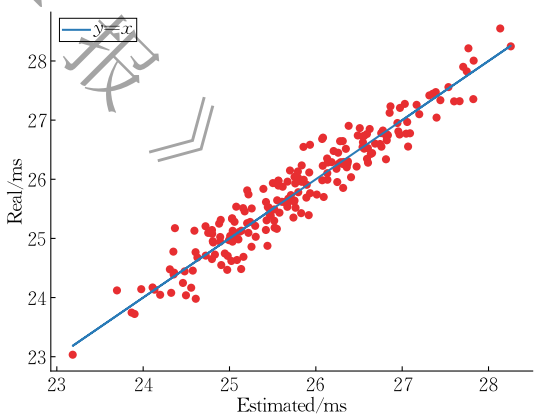

图 6　预测延时和真实延时对比

搜索详情. 在学习网络参数时，我们使用初始学习率为 0.08 的 SGD 优化器，并使用余弦退火将学习率从 0.08 逐渐减小到 0.0008，动量为 0.9，权重衰减为 0.0001. 我们总共进行了 60 个 epoch 的网络结构搜索. 对于进化算法，我们设置种群规模为 50，迭代次数为 20.

训练细节. 当架构搜索完成后，我们将离散架构解码为上文描述的结构. 我们使用具有 0.9 动量

和 0.0001 权重衰减的 SGD 优化器. 该模型训练了 50 个 epoch. 0.08 的初始学习率应用于前 30 个 epoch,并在 30 和 40 个 epoch 时除以 10.

4.3 物体检测结果

与最先进模型比较. 我们将搜索得到的与其他最先进的特征融合网络进行比较,包括手工设计的结构,如 FPN[6]、PANet[8] 和搜索得到的网络结构如 Auto-FPN[10]、NAS-FPN[9]. 为了公平比较,我们使用 RetinaNet-FPN 作为基线,使用 ResNet-50 作为主干网络. 对于其他融合方法,仅替换融合网络结构,训练设置保持不变. 我们也在实验中调整了 AFN 的层数 T. 如表 1 所示,在 mAP 方面,当输入大小设置为 640×640 时,AFN-3layers 相比 FPN 实现了 2.1%的提升,相比 PANet 实现了 1.4%的提升,与具有类似 FLOPs 和参数数量的 Libra-Net 相比提高了 1.1%,相比相同堆叠次数的 Bi-FPN 实现了 1.1%的提升. AFN-5layers 相比相同堆叠次数的 Bi-FPN 提高了 1.5%. 当输入大小设置为 1024×1024 时,AFN-5layers 与 FPN 相比实现了 3.4%的改进. 对于 640×640 和 1024×1024 输入大小,AFN-5layers 显示出与 NAS-FPN 相近的 mAP,但 FLOPs 和参数数量远小于 NAS-FPN.

表 1 在 COCO 数据集上和最先进的特征融合网络对比

Method	BackBone	image-size	param/M	FLOPs/B	mAP
FPNLite	MobileNetV2	640×640	3.46	11.34	29.6
AFNLite	MobileNetV2	640×640	3.84	11.59	31.6
FPN	ResNet50	640×640	37.75	95.68	37.0
PA-Net	ResNet50	640×640	40.12	98.04	37.7
Libra	ResNet50	640×640	38.03	106.27	38.0
NAS-FPN	ResNet50	640×640	59.74	138.75	39.9
Bi-FPN-3layers	ResNet50	640×640	44.98	109.09	38.0
Bi-FPN-5layers	ResNet50	640×640	54.43	121.63	38.2
AFN-3layers(ours)	ResNet50	640×640	38.29	101.86	39.1
AFN-5layers(ours)	ResNet50	640×640	42.05	107.82	39.7
FPN	ResNet50	1024×1024	37.75	244.94	40.1
AFN-5layers(ours)	ResNet50	1024×1024	42.22	276.62	43.5
NAS-FPN	ResNet50	1024×1024	59.74	355.25	44.2
SSD512	VGG16	512×512	36.04	154.40	29.3
SSD512-Auto-FPN	VGG16	512×512	31.88	—	31.8
SSD512-AFN(ours)	VGG16	512×512	39.16	154.01	33.5

Auto-FPN 在 SSD512 的顶部实现了他们的 Auto-fusion 模块,其中 VGG16 作为主干网络. 因此,为了进行比较,我们同样在 SSD512 的顶部实现了 AFN,并在重新训练模型时使用了与 SSD512 相同的超参数. 而对于搜索过程,预训练阶段的长度设置为 20 个 epoch,搜索 epoch 总数为 120. 实验表明,与未融合多尺度特征的 SSD512 相比,AFN 获得了 4.2%的 mAP 提升,并且以相比于搜索的 Auto-FPN 特征融合网络有 1.7%的 mAP 提升. 总之,与最先进的融合网络相比,我们提出的 AFN 实现了更好的准确性和延迟权衡. 我们还直观地在散点图中显示了比较. 我们使用相同的颜色来表示相同类型的检测器设置:蓝色表示 RetinaNet-MobileNetV2,橙色表示 RetinaNet-ResNet50,绿色表示 SSG512-VGG16. 如图 7 和图 8 所示,AFN 在准确性和效率之间实现了更好的权衡. 当保持相似的计算量和模型大小时,AFN 实现了更高的精度.

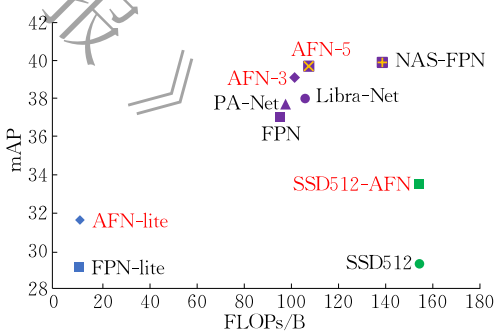

图 7 准确率(mAP)和计算量(FLOPs/B)对比

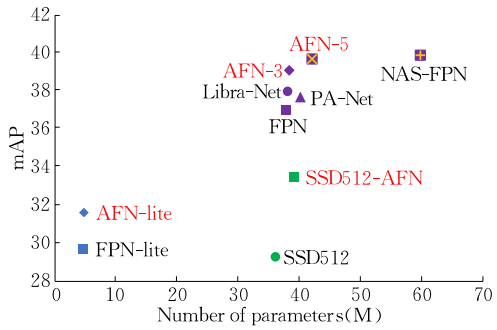

图 8 准确率(mAP)和参数量(M)对比

采用不同的主干网络. 为了进一步验证 AFN 在不同主干上的可扩展性，我们将 ResNet-50 替换为轻量级主干 MobileNetV2. 我们还用深度可分离卷积替换了 AFN 和 head 中的卷积层. 轻量级 AFN 称为 AFNLite. 在 AFNLite 中，每个特征层的输出通道数设置为 128，AFNLite 的层数设置为 3. 结果显示在表 1 中. 与 FPN 的基线相比，所提出的 AFN 在 ResNet-50 上实现了 2.1% 的提高，在 MobileNetV2 上实现了 2.0% 的提高. 综上所述，AFN 可以在不同的主干网络上获得更好的性能，这验证了 AFN 的可扩展性和有效性.

三个搜索单元的消融实验. 在本节中，我们将探讨提出的三个搜索单元在主干网络为 Resnet-50 时对检测性能的影响. 结果示于表 2 中.

表 2　三种搜索单元的消融实验

融合模式	全部	全部	全部	固定（最佳值）	固定（平均值）
融合特征搜索单元	√	√		√	√
融合路径搜索单元	√		√	√	√
融合模式搜索单元	√	√	√		
AP	**39.1**	38.5	38.3	38.8	38.4
AP_{50}	**57.1**	56.7	57.0	57.0	57.1
AP_{75}	**41.5**	40.9	40.1	41.3	40.4
AP_S	**18.6**	18.2	17.9	18.6	17.9
AP_M	**44.8**	43.5	43.7	43.9	44.0
AP_L	**55.0**	54.3	54.9	55.1	54.9

与使用特征搜索单元相比，当我们将主干的特征固定为每个阶段的顶层时，搜索到的架构的 mAP 比 AFN 低 0.8%. 因此，特征搜索单元可以从主干中带来更好的特征选择并实现更高的性能. 当我们使用与 FPN 相同的标准自顶向下融合路径并移除融合路径搜索单元时，mAP 比 AFN 低 0.6%. 该结果表明融合路径搜索单元可以比固定融合路径具有更好的融合效果. 为了分析融合模式搜索单元，我们将融合操作固定为搜索空间中的任意一个. 这些融合模式的平均性能比 AFN 低 0.7%，它们中的最优性能比 AFN 低 0.3%. 结果证明融合模式搜索单元可以为每次融合找到更合适的操作并获得更高的性能.

硬件感知的搜索目标函数消融实验. 在本节中，我们将探讨提出的硬件感知搜索目标函数在主干网络为 Resnet-50，融合网络堆叠层数为 3 层时对检测精度、推理时间、参数量和计算量的影响（推理时间、参数量和计算量只考虑特征融合网络部分）. 表 3 中给出了对比结果. 使用硬件感知的搜索目标函数能够保证在不影响精度的前提下一定程度上降低存储开销并显著降低计算时间开销. 另外，我们发现，硬件感知的搜索目标函数在降低计算时间开销时并没有显著降低计算量（FLOPs/B）. 这说明计算量和计算时间开销并不呈线性相关，而硬件感知的搜索目标函数能够捕捉到硬件特性从而有效地降低计算时间开销.

表 3　硬件感知的搜索目标函数消融实验

	mAP	inference time/ms	FLOPs/B	param/M
AFN 硬件感知	39.1	23.3922	10.49	6.28
AFN	39.1	26.6159	10.77	6.59

定性结果分析. 在本节中，我们展示了 FPN 和 AFN 在 COCO 数据集上比较的定性结果. 所有这些结果都是使用 RetinaNet 检测器获得的. 其中主干网络为 ResNet50. 如图 9 所示，AFN 可以发现一些 FPN 缺失的小物体，如图 9(a)、(c)、(e)和(h). 这也和数值结果相对应：AFN 小物体检测 mAP 为 18.6，FPN 小物体检测 mAP 为 16.4. 此外，AFN 可以去除 FPN 的一些误报边界框，如图 9(b)、(d)、(f)和(g)所示. 一些仅包含部分物体的 FPN 的误报边界框也将被 AFN 删除或纠正，如图 9(f)所示. 另外，相对于真实标注，AFN 只存在少量漏检，很少出现误检情况. 这些结果表明，AFN 可以有效地融合多尺度特征以获得具有强语义和空间信息的特征，从而提高目标检测的性能.

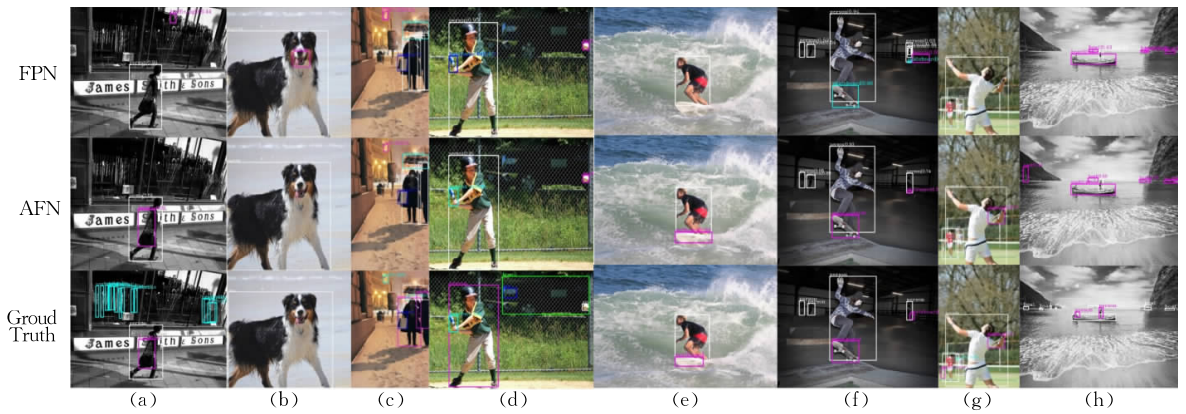

图 9　在 COCO 数据集上对 FPN 和 AFN 的定性对比

5 结 论

在本文中,我们提出了硬件感知的高效特征融合网络搜索框架来搜索既能高效融合多尺度特征又能在目标硬件高效运行的特征融合网络.考虑到输入融合特征、融合路径和融合操作,我们设计了搜索空间以覆盖大部分现有的融合网络.在 COCO 数据集上的实验表明,AFN 可以更好地表示多尺度特征,并且优于最先进的特征融合网络.

参 考 文 献

[1] Girshick R. Fast R-CNN//Proceedings of the IEEE International Conference on Ccomputer Vision. Santiago, Chile, 2015: 1440-1448

[2] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards real-time object detection with region proposal networks// Advances in Neural Information Processing Systems. Montreal, Canada, 2015: 91-99

[3] Liu W, Anguelov D, Erhan D, et al. SSD: Single shot MultiBox detector//Proceedings of the European Conference on Computer Vision. Amsterdam, The Netherlands, 2016: 21-37

[4] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection//Proceedings of the IEEE International Conference on Computer Vision. Venice, Italy, 2017: 2980-2988

[5] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Columbus, USA, 2014: 580-587

[6] Lin T Y, Dollár P, Girshick R, et al. Feature pyramid networks for object detection//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, USA, 2017: 2117-2125

[7] Tan M, Pang R, Le Q V. EfficientDet: Scalable and efficient object detection. CoRR, abs/1911.09070, 2019

[8] Liu S, Qi L, Qin H, et al. Path aggregation network for instance segmentation//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 8759-8768

[9] Ghiasi G, Lin T Y, Le Q V. NAS-FPN: Learning scalable feature pyramid architecture for object detection//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach, USA, 2019: 7036-7045

[10] Xu H, Yao L, Zhang W, et al. Auto-FPN: Automatic network architecture adaptation for object detection beyond classification//Proceedings of the IEEE/CVF International Conference on Computer Vision. Seoul, Korea(South), 2019: 6649-6658

[11] Liu S, Huang D, Wang Y. Learning spatial fusion for single-shot object detection. CoRR, abs/1911.09516, 2019

[12] Li Y, Pang Y, Shen J, et al. NETNet: Neighbor erasing and transferring network for better single shot object detection// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle, USA, 2020: 13349-13358

[13] Hu J, Shen L, Sun G. Squeeze-and-excitation networks// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 7132-7141

[14] Woo S, Park J, Lee J Y, et al. CBAM: Convolutional block attention module//Proceedings of the European Conference on Computer Vision(ECCV). Munich, Germany, 2018: 3-19

[15] Wang F, Jiang M, Qian C, et al. Residual attention network for image classification//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, USA, 2017: 3156-3164

[16] Wang X, Girshick R, Gupta A, et al. Non-local neural networks//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 7794-7803

[17] Pang J, Chen K, Shi J, et al. Libra R-CNN: Towards balanced learning for object detection//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach, USA, 2019: 821-830

[18] Zoph B, Le Q V. Neural architecture search with reinforcement learning. CoRR, abs/1611.01578, 2016

[19] Real E, Aggarwal A, Huang Y, et al. Regularized evolution for image classifier architecture search. CoRR, abs/1802.01548, 2018

[20] Liu H, Simonyan K, Yang Y. DARTS: Differentiable architecture search. arXiv preprint arXiv:1806.09055, 2018

[21] Cai H, Zhu L, Han S. ProxylessNAS: Direct neural architecture search on target task and hardware. arXiv preprint arXiv:1812.00332, 2018

[22] You S, Huang T, Yang M, et al. GreedyNAS: Towards fast one-shot NAS with greedy supernet//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle, USA, 2020: 1999-2008

[23] Chen Y, Yang T, Zhang X, et al. Detnas: Backbone search for object detection//Advances in Neural Information Processing Systems. Vancouver, BC, Canada, 2019: 6638-6648

[24] Yao L, Xu H, Zhang W, et al. SM-NAS: Structural-to-modular neural architecture search for object detection// Proceedings of the AAAI Conference on Artificial Intelligence. New York, USA, 2020, 34(7): 12661-12668

[25] Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient convolutional neural networks for mobile vision applications. CoRR, abs/1704.04861, 2017

[26] Sandler M, Howard A, Zhu M, et al. MobileNetV2: Inverted residuals and linear bottlenecks//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 4510-4520

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

[28] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection. CoRR, abs/1506.02640, 2015

[29] Qin Z, Li Z, Zhang Z, et al. ThunderNet: Towards real-time generic object detection on mobile devices//Proceedings of the IEEE/CVF International Conference on Computer Vision. Seoul, Korea, 2019: 6718-6727

[30] 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

[31] 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

[32] 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

[33] 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 (ECCV). Munich, Germany, 2018: 285-300

[34] Noori M, Mohammadi S, Majelan S G, et al. DFNet: Discriminative feature extraction and integration network for salient object detection. Engineering Applications of Artificial Intelligence, 2020, 89: 103419

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

[36] Lin T Y, Maire M, Belongie S, et al. Microsoft COCO: Common objects in context//Proceedings of the European Conference on Computer Vision. Cham, Switzerland: Springer, 2014: 740-755

[37] Chen K, Wang J, Pang J, et al. MMDetection: Open MMLab detection toolbox and benchmark. CoRR, abs/1906.07155, 2019

[38] Paszke A, Gross S, Massa F, et al. PyTorch: An imperative style, high-performance deep learning library//Advances in Neural Information Processing Systems. Vancouver, Canada, 2019: 8024-8035

GUO Jia-Ming, Ph. D. candidate. His research interest is deep learning.

ZHANG Rui, Ph. D., associate professor. Her research interests include deep learning, reinforcement learning.

ZHI Tian, Ph. D., senior engineer. Her research interests include integrated circuit design, reconfigurable calculation.

HE De-Yuan, M. S. His research interest is deep learning.

HUANG Di, Ph. D. candidate. His research interest is deep learning.

CHANG Ming, M. S. His research interest is deep learning.

ZHANG Xi-Shan, Ph. D., associate professor. Her research interest is deep learning.

GUO Qi, Ph. D., professor. His research interest is intelligent computing system.

Background

The task of object detection is to find the positions of all objects of interest in the image and distinguish the categories. For a long time, object detection has had important applications. With the development of deep learning, researchers introduced deep convolutional neural networks to achieve end-to-end target detection.

Modern target detection frameworks generally consist of a backbone neural network used to extract features, a feature fusion network used to fuse features of different scales, and a detection head network used to locate and classify features based on features. The backbone neural network generally directly reuses the network structure used for object classification. Researchers often improve the performance of target detection by optimizing the feature fusion network. However, existing works of designing feature fusion network architecture mainly focus on designing the topology of connections to fuse

features.

We propose a new feature fusion network named Attention-aware Fusion Neck (AFN) which has a strong capacity for fusing multi-scale features for object detection. To design the architecture of a fusion network, AFN considers three key factors: input fusion features, fusion path, fusion operation. For fusion operation, AFN introduces many different kinds of attention mechanisms besides the commonly used add operation. The search space of AFN covers most of the existing architectures of fusion neck, so AFN could be treated as the generalization of most existing fusion networks. Although complex fusion paths and fusion operations can improve the performance of object detection, they bring too much hardware resource overhead for terminal devices. To achieve a better trade-off between object detection precision and hardware resources, we employ the neural architecture search method. We propose well-designed search modules in AFN and take the hardware overhead (parameter storage, computing time, etc.) of the target platform as a consideration in the search process. Experiments results indicate that combined with different backbone networks, the feature fusion network we searched has better performance than the existing feature fusion network and is more efficient than previous searched feature fusion networks.