Computer Science and Application 计算机科学与应用, 2024, 14(10), 1-9
Published Online October 2024 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2024.1410197

Hans汉斯

注意力机制引导的神经架构搜索在图像分类中的应用

许　斌[1]，岑　颖[2]，刘健龙[1]，焦　旋[1]

[1]江西理工大学信息工程学院，江西　赣州
[2]广东茂名幼儿师范专科学校，广东　茂名

收稿日期：2024年8月25日；录用日期：2024年9月23日；发布日期：2024年9月30日

摘　要

神经架构搜索(Neural Architecture Search, NAS)是一种能够在预定义搜索空间内找到最优神经网络架构的方法，在图像分类等任务上具有重要的意义。注意力引导的架构搜索(Attention-guided Micro and Macro-Architecture Search, AGNAS)能根据注意力机制对重要操作赋予高的权重，避免了传统NAS方法中操作被错误选择的问题。本文分别从注意力机制和操作单元两个方面进一步对AGNAS方法进行改进。注意力机制方面，将通道注意力更换为瓶颈注意力(Bottleneck Attention Module, BAM)模块，即在通道注意力模块后增加空间注意力模块，以同时关注通道和空间重要特征。操作单元方面，引入ShuffleUnit操作，增加操作选择的多样性，进一步提升网络的特征表达能力。CIFAR-10数据集上的实验结果表明，与原始AGNAS相比，本文的改进方法能获得更低的分类错误率和更高的稳定性；与其他一些架构搜索方法相比，本文方法在分类错误率、参数量和搜索时间等方面的总体性能上更具优势。

关键词

神经架构搜索，图像分类，瓶颈注意力模块，ShuffleUnit

Application of Attention-Guided Neural Architecture Search in Image Classification

Bin Xu[1], Ying Cen[2], Jianlong Liu[1], Xuan Jiao[1]

[1]School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou Jiangxi
[2]Guangdong Preschool Normal College in Maoming, Maoming Guangdong

Received: Aug. 25th, 2024; accepted: Sep. 23rd, 2024; published: Sep. 30th, 2024

Abstract

Neural Architecture Search (NAS) is a method that can find the optimal neural network architecture

文章引用: 许斌, 岑颖, 刘健龙, 焦旋. 注意力机制引导的神经架构搜索在图像分类中的应用[J]. 计算机科学与应用, 2024, 14(10): 1-9. DOI: 10.12677/csa.2024.1410197

within a predefined search space and is of great significance in tasks such as image classification. Attention-Guided Micro and Macro-Architecture Search (AGNAS) can assign high weights to important operations based on attention mechanisms, avoiding the problem of incorrect selection of operations in traditional NAS methods. In this paper, we improve the AGNAS method from two aspects: attention mechanism and operation unit. Regarding attention mechanism, channel attention is replaced with bottleneck attention (BAM) module, which adds spatial attention module after channel attention module to focus on channel and spatial important features simultaneously. In terms of operation units, ShuffleUnit operation is introduced to increase the diversity of operation selection, further improving the feature expression ability of the network. The experimental results on the CIFAR-10 dataset show that compared with the original AGNAS, our method can achieve lower classification error and higher stability; Compared with the other architecture search methods, our method has overall advantages in classification error, parameters, and search time.

Keywords

Neural Architecture Search, Image Classification, Bottleneck Attention Module, ShuffleUnit

1. 引言

近年来，深度神经网络在计算机视觉领域中的图像分类、目标检测、语义分割等任务上取得了显著成绩。然而，这些网络的结构和超参数往往需要人工选择，耗费了大量的人力。神经架构搜索(NAS)通过在预定义的搜索空间内自动搜索最佳网络架构，不仅有效减少了人工干预，还能够实现比人工设计网络更优异的性能。早期的 NAS 方法，如基于强化学习(RL) [1]和进化算法(EA) [2]的搜索方法，尽管能够找到高性能架构，但计算成本高，通常需要成千上万个 GPU 小时。高效神经架构搜索(ENAS) [3]采用在子架构之间共享权重的策略，将搜索时间降低到几个 GPU 天，极大地提高了 NAS 的实用性。之后，可微架构搜索(DARTS) [4]通过将离散搜索空间松弛为连续空间，并采用双层优化策略，实现了高效的端到端训练。为提升分类任务中的性能，基于梯度的可微架构搜索(GDAS) [5]方法通过可微分的 Gumbel-Softmax 来进行离散架构选择。稀疏梯度架构搜索(SGAS) [6]通过逐步修剪冗余操作来减少搜索空间。Shapley-NAS [7]通过评估候选操作的贡献来优化网络结构。基于可微分神经架构搜索(ADARTS) [8]提出了一种基于通道注意力的可微神经架构部分通道连接算法。基于爆炸引力场算法的神经架构搜索(EGFA-NAS) [9]通过爆炸引力场算法提升了全局搜索能力和搜索效率。

上述架构搜索方法中仍存在一些问题，如操作选择不合理导致整个网络性能下降[10]。注意力引导的架构搜索(AGNAS) [11]能够根据注意力机制对重要操作赋予高的权重，避免了操作被错误选择的问题。本文对 AGNAS 进行改进，将原有的通道注意力替换为瓶颈注意力(BAM)模块[12]，以在通道和空间通道上选择性地强调重要特征，并利用权重来量化每个操作对网络的贡献。此外，本文在操作选择中加入文献[13]中的 ShuffleUnit 操作，结合深度可分离卷积(DWConv)和通道特征组之间交换通道(Channel shuffle)等运算，增强通道组之间的信息交换能力，进而提升模型的特征表达能力。最后，将改进后的 AGNAS 应用于图像分类任务中，提升图像分类的正确率。

2. 改进的 AGNAS 架构介绍

改进的 AGNAS 整体网络架构如图 1 所示，由多个单元(Cell)堆叠而成。Cell 又分为 Reduction Cell 和

Normal Cell 两种类型。前者通过下采样操作减少特征图的空间分辨率并增加通道数，以此来压缩信息和减少计算开销；后者用于提取特征，并保持特征图的空间尺寸不变。每个 Cell 是一个有向无环图(DAG)，由 4 个中间节点(图 1 中用 0、1、2、3 示意)，及任意两个节点间 9 条边(图 1 中用 4 条边示意)组成。每个节点表示一组特征图，而每条边则表示从输入节点到输出节点的操作。每个 Cell 的输入为前两个 Cell 的输出特征，特别地，第一个 Cell 的输入为固定 Stem 层(由一个 3 × 3 卷积和批量归一化层构成)从原始输入图像中提取出特征。每个中间节点通过聚合来自其所有前节点的信息流来生成新的特征表示。输出节点是将中间节点的输出特征按通道维度进行拼接得到。

与原 AGNAS 相比，本文的改进之处包括两个方面：(1) 每个操作后，采用 BAM 模块替换原通道注意力模块，以在通道和空间两个维度上关注重要特征；(2) 在原来的 8 种操作(3 × 3 和 5 × 5 可分离卷积 sep_conv、3 × 3 和 5 × 5 扩张可分离卷积 dil_conv、3 × 3 最大池化 max_pool、3 × 3 平均池化 avg_pool、跳跃连接 skip_connect、以及空操作 none)基础上，引入 ShuffleUnit 操作，增加操作选择的多样性。

接下来，先介绍改进的 AGNAS 的架构搜索原理，然后详细介绍 BAM 模块和 ShuffleUnit 操作。

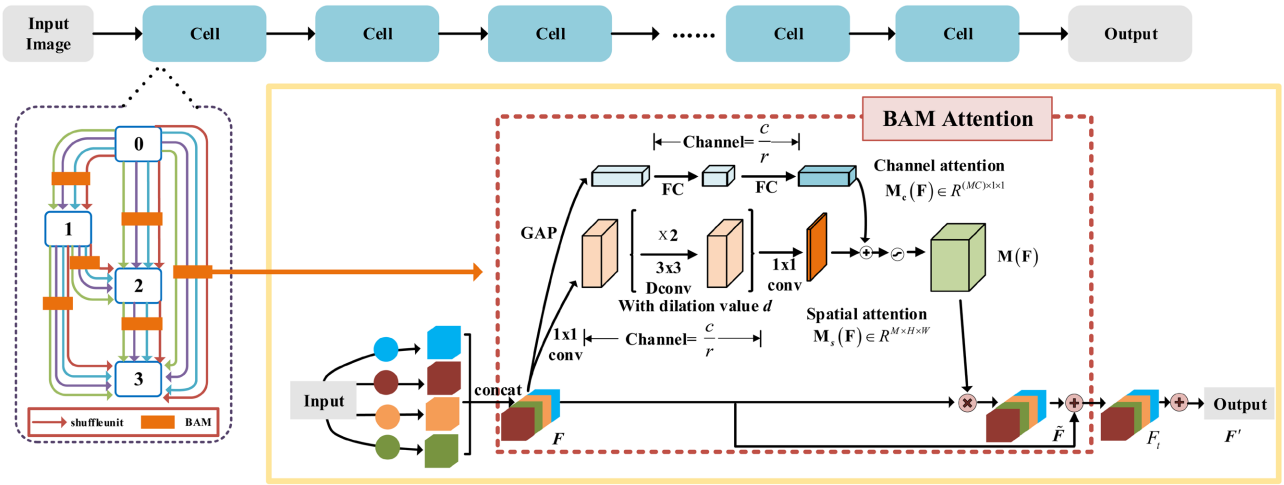


Figure 1. Architecture of the improved AGNAS
图 1. 改进的 AGNAS 的网络架构

2.1. 架构搜索原理

对于用于分类任务的整体架构，其搜索过程中采用的损失函数为，

$$\text{Loss} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{g=1}^{G} y_{n,g}\log\left(P_{n,g}\right) \tag{1}$$

其中，N 为样本数量，G 为类别数量，$y_{n,g}\left(n=1,2,\cdots,N;\ g=1,2,\cdots,G\right)$ 为第 n 个样本的真实标签，$P_{n,g}$ 为第 n 个样本被预测为类别 g 的概率值。

对于每个 Cell，其架构搜索过程如下。首先，采用所有 M = 9 种候选操作对输入特征进行运算，并将生成的特征在通道维度上拼接，得到中间特征图 $F \in \mathbb{R}^{(MC)\times H\times W}$，其中，$C$ 表示特征通道数，H 和 W 分别表示特征图的高和宽。

然后，对中间特征图 F，采用 BAM 模块关注通道和空间重要特征，得到注意力权重 $M\left(F\right) \in \mathbb{R}^{(MC)\times H\times W}$。进一步，定义任意的第 m ($m=1,2,\cdots,M$)种候选操作 A_m 的重要性 I_{A_m} 为，

$$I_{A_m} = \sum_{i=(m-1)C+1}^{mC}\sum_{j=1}^{H}\sum_{k=1}^{W} M\left(F\right)(i,j,k). \tag{2}$$

最后，按照候选操作重要性的最大值选择每条边上的最佳操作 I_A^*，

$$I_A^* = \arg\max\left(I_{A_1}, I_{A_2}, \cdots, I_{A_m}\right). \tag{3}$$

重复同样操作，直到选出其余各个节点最佳操作为止。将所有节点最佳操作进行组合，即可得到该 Cell 的最优网络架构。

2.2. BAM 模块

BAM 模块的结构如图 1 中红色虚线框内所示，包括通道和空间注意力两条路径。通道注意力路径用于选择性地增强或抑制特征图中不同通道的响应，而空间注意力用于选择性地增强或抑制特征图中不同空间位置的响应。对特征图 $F \in \mathbb{R}^{(MC)\times H\times W}$，分别计算相应的通道注意力 $M_C(F)$ 和空间注意力 $M_S(F)$。

在通道注意力路径上，先将特征图 F 通过全局平均池化(GAP)，生成一个包含全局信息的通道向量 $F_c \in \mathbb{R}^{MC}$。然后，采用多层感知器(MLP) (用全连接运算实现)和批量归一化层(BN)来获取通道注意力 $M_c(F) \in \mathbb{R}^{(MC)\times 1\times 1}$，计算公式为，

$$\begin{aligned} M_c(F) &= BN\left(MLP\left(AvgPool(\boldsymbol{F})\right)\right) \\ &= BN\left(\mathrm{W}_1\left(\mathrm{W}_0 AvgPool(\boldsymbol{F}) + b_0\right) + b_1\right) \end{aligned} \tag{4}$$

其中，$\mathrm{W}_0 \in \mathbb{R}^{MC/r\times C}$ 和 $\mathrm{W}_1 \in \mathbb{R}^{MC\times C/r}$ 为两个全连接层的权重，$b_0 \in \mathbb{R}^{MC/r}$ 和 $b_1 \in \mathbb{R}^{MC}$ 为两个全连接层的偏置，r 为通道上的降维比例。

在空间注意力路径上，先采用 1 × 1 卷积来压缩特征图 F 的通道数，即将 F 降维为 $\mathbb{R}^{(MC)/r\times H\times W}$，然后利用两个 3 × 3 膨胀卷积(DConv)捕捉上下文信息。最后，用 1 × 1 卷积(Conv)将特征维度变为 $\mathbb{R}^{M\times H\times W}$。整个计算过程可表示为，

$$M_S(F) = BN\left(f^{1\times 1}\left(f^{3\times 3}\left(f^{3\times 3}\left(f^{1\times 1}(F)\right)\right)\right)\right) \tag{5}$$

其中，上标 1 × 1 和 3 × 3 分别表示卷积核的大小。

在获取 $M_C(F)$ 和 $M_S(F)$ 后，分别将两者的维度复制扩充至 $\mathbb{R}^{(MC)\times H\times W}$，并采用 Sigmoid 函数将其映射至 0 到 1 范围内，得到合并后的注意映射 $M(F)$ 为，

$$M(F) = \sigma\left(M_c(F) \oplus M_s(F)\right) \tag{6}$$

其中，σ 表示 Sigmoid 函数，$\oplus$ 表示逐元素求和。

进一步地，将 $M(F)$ 与 F 相乘，得到加权后的输出特征图 $\tilde{F} \in \mathbb{R}^{(MC)\times H\times W}$ 为，

$$\tilde{F} = F \otimes M(F) \tag{7}$$

其中，$\otimes$ 表示逐元素相乘。

又将 $\tilde{F}$ 和 F 进行逐元素求和得到 F_t，并对 F_t 进行通道维度上同操作数的逐元素求和，得到输出节点的特征图 $F' \in \mathbb{R}^{C\times H\times W}$，计算公式为，

$$\begin{cases} F_t = \tilde{F} \oplus F \\ F'(c) = \sum_{m=1}^{M} F_t\left(c + (m-1)C\right) \quad (c = 1, 2, \cdots, C) \end{cases}. \tag{8}$$

2.3. ShuffleUnit 操作

ShuffleUnit 是 ShuffleNet V2 的一个组成单元，其结构如图 2 所示。首先，用两个分支对特征进行卷

积运算。第一个分支主要的运算依次为 1x1 Conv、步长为 2 的 3 × 3 DWConv 和 1 × 1 Conv；第二个分支主要的运算依次为步长为 2 的 3 × 3 DWConv 和 1 × 1 Conv。此外，两个分支中，1 × 1 Conv 后面需要 BN 和 ReLU 激活运算，步长为 2 的 3 × 3 DWConv 后面需要 BN 运算。然后，将两个分支得到的特征图在通道维度上拼接。最后，将拼接的特征图划分为多个通道组，并利用 Channel shuffle 实现通道组之间的信息交换。

显然，DWConv 和 1 × 1 Conv 的组合能有效提高计算效率，步长为 2 的 DWConv 能实现特征在空间上的降采样，两条分支可视为特征的重复使用。因此，ShuffleUnit 是一个能高效完成具有更多特征通道和更大网络容量的信息提取单元，而且具有通道组之间信息交换能力。

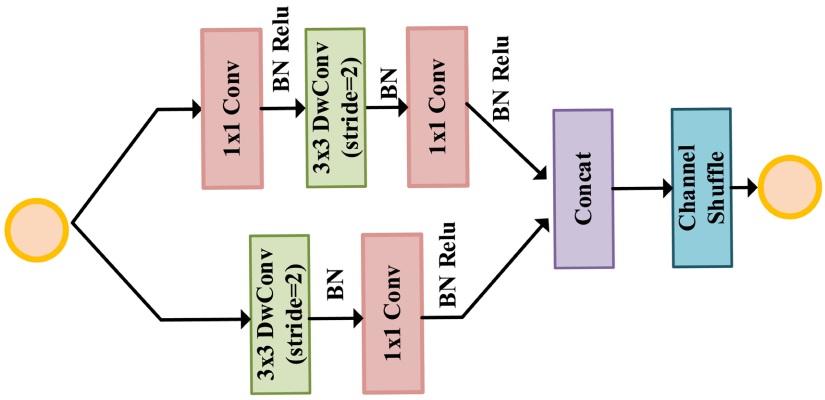


Figure 2. ShuffleUnit operation
图 2. ShuffleUnit 操作

3. 实验结果

本文采用的数据集为 CIFAR-10，包含 60,000 幅 32 × 32 的彩色图像。总共有 10 个类别目标(飞机、汽车、鸟、猫、鹿、狗、青蛙、马、船和卡车)，每类包含 6000 幅图像。由于本文的架构搜索需要验证集参与，因此，将数据集划分为训练、验证和测试集，比例为 2.5:2.5:1。

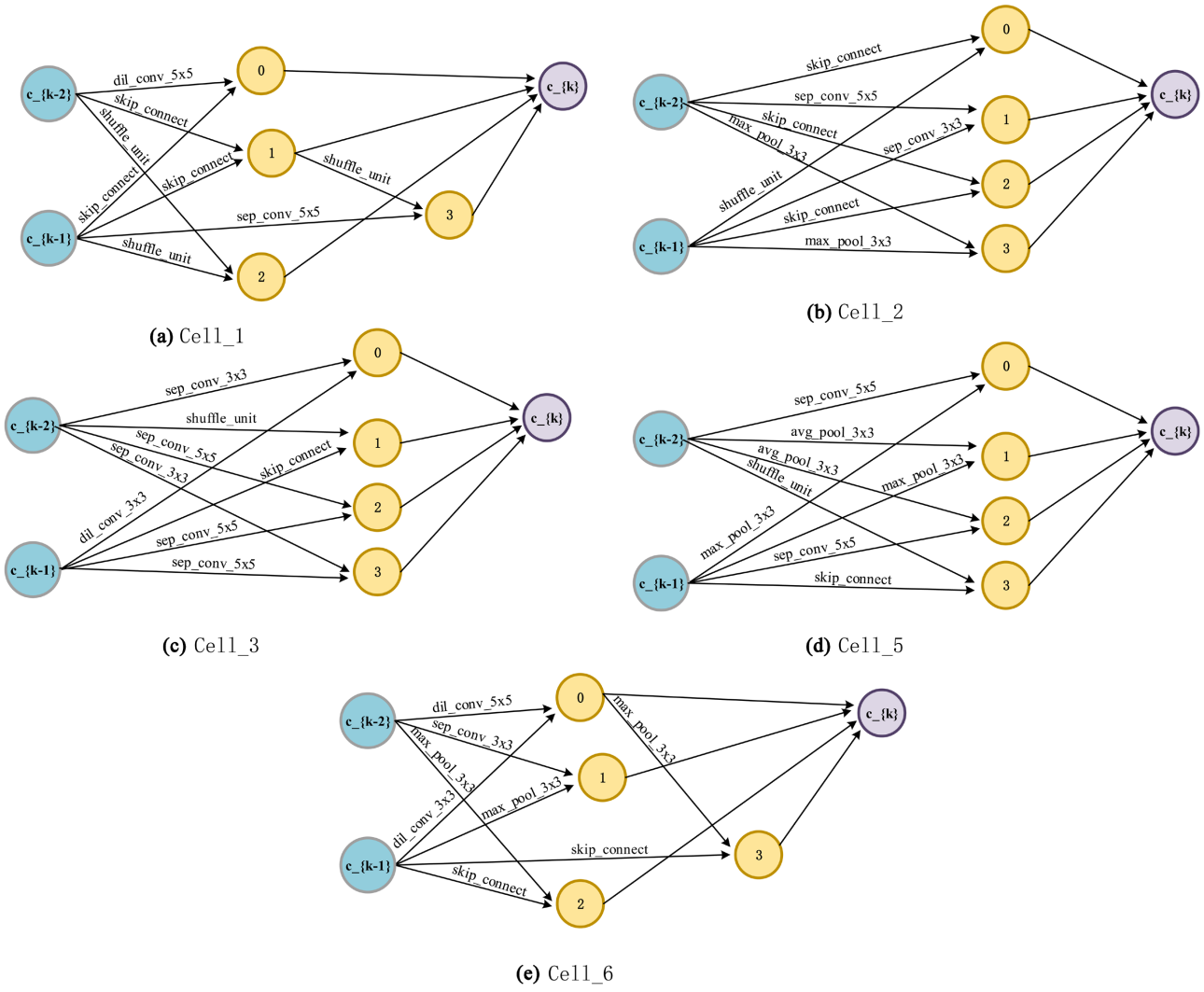
实验软件环境为 Ubuntu20.04 系统，深度学习框架为 pytorch1.12.1，硬件配置 CPU 为 Intel® Xeon(R) W-2235 CPU @ 3.80 GHz × 12，内存 64 G，GPU 为 NVIDA GeForce RTX 3090，显存 24G。实验使用 SGD 优化器来更新网络权重，初始学习率为 0.025，动量为 0.9，权重衰减为 3×10^{-4}。

3.1. 架构搜索结果

为了在一个大的空间中搜索，先构建一个包含 8 个 Cell (2 个 Reduction Cell 和 6 个 Normal Cell)的超网。其中，2 个 Reduction Cell 分别放置在第 2 个(超网的 1/3 长度处)和第 5 个(超网的 2/3 长度处)位置处。采用的候选操作选择包括：3 × 3 和 5 × 5 sep_conv、3 × 3 和 5 × 5 dil_conv、shuffle_unit、max_pool_3 × 3、avg_pool_3 × 3、skip_connect、以及 none 这 9 种操作。搜索过程中，计算 Cell 中各节点每个候选操作对应的注意力权重，并选择具有最高注意力权重的操作。将所有被选中操作进行组合，即可得到该 Cell 的最优网络架构。每一轮搜索中，8 个 Cell 的架构都会更新。

搜索结束后，为了平衡计算效率和性能，仅选择了其中 5 个 Cell 进行堆叠。其中，2 个 Reduction Cell 被全部保留，6 个 Normal Cel 中随机选出 3 个。选出的 5 个 Cell 的结构如图 3(a)~(e)所示。其中图 3(b)和(d)是 Reduction Cell (Cell_2 和 Cell_5)，其余为 Normal Cell (Cell_1、Cell_3 和 Cell_6)。图 3(a)中 Cell_1 采用了 5 × 5 dil_conv 和多个 skip_connect，而且在节点 1 到节点 3 和两个输入节点分别到节点 2 的过程

中均选择了 shuffle_unit 操作。Cell_2 采用 3×3 和 5×5 的 sep_conv、avg_pool_3×3 和多个 skip_connect，且在输入节点(前一个 Cell 的输出)到节点 0 的过程中选择了 shuffle_unit 操作。Cell_3 采用 3 × 3 和 5 × 5 的 sep_conv 和 3 × 3 的 dil_conv，并通过 skip_connect 将不同层的特征图融合，且也在输入节点(前两个 Cell 的输出)到节点 1 的过程中选择了 shuffle_unit 操作。Cell_5 采用 5 × 5 sep_conv、avg_pool_3 × 3 和 max_pool_3×3，且在输入节点(前两个 Cell 的输出)到节点 3 的过程中选择了 shuffle_unit 操作。Cell_6 主要使用 3×3 和 5×5 的 dil_conv、max_pool_3×3 和 3×3 sep_conv，并通过多个 skip_connect 融合特征。在这些选中的操作中，shuffle_unit 操作被选择到的概率达到了 15%，因此，shuffle_unit 操作的引入不仅增加了操作的多样性，而且在架构用于分类的性能提高方面发挥着重要作用。



(a) Cell_1 **(b)** Cell_2

(c) Cell_3 **(d)** Cell_5

(e) Cell_6

Figure 3. Detailed architecture of 5 cells searched on the CIFAR-10 dataset (Light blue represents input features; light purple represents output features; light yellow represents intermediate nodes)

图 3. CIFAR-10 数据集上搜索到的 5 个 Cell 的详细架构(淡蓝色表示输入特征；淡紫色表示输出特征；黄色表示中间节点)

受文献[11]的启发，将选出的 5 个 Cell 进行堆叠，形成一个新的包含 20 个 Cell 的网络，以用于最终的分类任务。堆叠过程中，将 2 个 Reduction Cell (Cell_2 和 Cell_5)分别放置在网络的第 7 个和第 14 个位置；其余 3 个 Normal Cell (Cell_1、Cell_3 和 Cell_6)各堆叠 6 次，结果如图 4 所示。采用 CIFAR-10 的训

练集和验证集对该网络的参数进行训练，后进行分类测试，得到的混淆矩阵如表 1 所示。显然，每类目标的正确率均高于 95%，10 类目标的平均分类正确率为 97.54%。

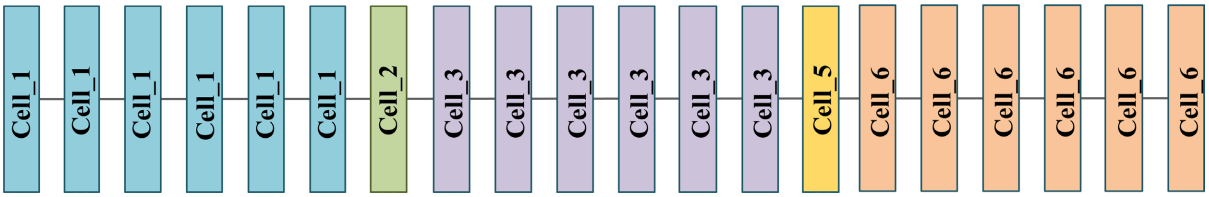


Figure 4. Classification network
图 4. 分类网络

3.2. 消融实验

以原始的 AGNAS 模型作为基线，分别以及同时加入 BAM 模块和 ShuffleUnit 操作，得到测试集上的平均分类错误率如表 2 所示。基线模型得到的平均测试错误率为 2.53%。当引入 BAM 模块后，测试错误率降低了 0.03%；当引入 ShuffleUnit 操作时，测试错误率降低了 0.04%；当同时引入 BAM 模块和 ShuffleUnit 操作时，测试错误率降低了 0.07%。结果表明，引入 BAM 模块或 ShuffleUnit 操作均能够降低测试错误率；当同时引入两者时，分类性能进一步得到提升。

Table 1. Confusion matrix
表 1. 混淆矩阵

Class	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck	Accuracy (%)
airplane	975	0	3	2	2	0	0	0	15	3	97.5
automobile	0	987	0	0	1	0	0	0	1	11	98.7
bird	2	0	972	6	4	5	10	0	1	0	97.2
cat	1	1	2	947	5	35	6	1	2	0	94.7
deer	0	0	3	8	980	4	2	3	0	0	98.0
dog	1	1	4	31	6	954	0	3	0	0	95.4
frog	0	0	3	1	1	1	994	0	0	0	99.4
horse	0	0	1	4	2	6	0	987	0	0	98.7
ship	8	7	2	0	0	0	1	0	982	0	98.2
truck	1	17	0	0	0	0	1	0	5	976	97.6
Total											97.54

Table 2. Results of the ablation experiment
表 2. 消融实验结果

Baseline	BAM	ShuffleUnit	Test error (%)↓
√			2.53
√		√	2.49
√	√		2.50
√	√	√	**2.46**

3.3. 与其他方法的对比

将本文方法与其他方法进行对比，各方法的测试错误率、模型参数量、搜索时间和搜索算法如表 3 所示。在基于非梯度搜索算法的架构搜索方法(NASNet-A [14]、AmoebaNet-B [2]、PNAS [15]和 ENAS [3])中，NASNet-A 和 AmoebaNet-B 两种方法的平均测试错误率分别达到了 2.65%和 2.55%，但它们的搜索时间却非常长，分别需要 1800 和 3150 GPU 天。这些方法依赖于计算密集型的 RL 学习和 EA 搜索算法，资源消耗巨大。而基于梯度(Gradient)算法的架构搜索方法(DARTS [4]、GDAS [5]、PC-DARTS [16]、SGAS [6]、DARTS + PT [10]和 AGNAS [11])，虽在平均测试错误率上略高，但它们的参数量少、搜索时间短，优势更为明显。其中，PC-DARTS 仅需 0.1 GPU 天。

本文改进模型获取的平均测试错误率达到了 2.46%，错误率的波动范围为±0.001%，参数量为 3.8 M，搜索时间仅为 0.39 GPU 天。与原始的 AGNAS 相比，不仅平均测试错误率降低，而且分类性能稳定。此外，在参数量上和计算效率上与原始的 AGNAS 相当。与其他梯度方法相比，平均测试错误率得到明显降低，分类性能更为稳定，而耗时在同一数量级上。因此，综合考虑分类错误率、性能稳定性、效率和资源消耗，本文方法比其他方法更具有优势。

Table 3. Comparison with other methods
表 3. 与其他方法的比较

Methods	Test error (%)	Parameters (M)	Search cost (GPU-days)	Search algorithm
NASNet-A [14]	2.65	3.3	1800	RL
AmoebaNet-B [2]	2.55 ± 0.05	2.8	3150	EA
PNAS [15]	3.41 ± 0.09	3.2	225	SMBO
ENAS [3]	2.89	4.6	0.5	RL
DARTS (1st order) [4]	3.00 ± 0.14	3.3	1.5	Gradient
DARTS (2st order) [4]	2.76 ± 0.09	3.3	4	Gradient
GDAS [5]	2.93	3.4	0.21	Gradient
PC-DARTS [16]	2.57 ± 0.07	3.6	0.1	Gradient
SGAS [6]	2.66 ± 0.24	3.7	0.25	Gradient
DARTS+PT [10]	2.61 ± 0.08	3.0	0.8	Gradient
AGNAS [11]	2.53 ± 0.03	3.6	0.4	Gradient
Ours	**2.46** ± 0.001	3.8	0.39	Gradient

4. 结束语

本文对原始 AGNAS 进行改进，每个操作后引入 BAM 注意力机制，并增加一个新的 ShuffleUnit 操作。首先，分析了网络的架构搜索原理，并详细阐述了 BAM 注意力机制和 ShuffleUnit 操作的工作原理。然后，在 CIFAR-10 数据集上开展了图像分类任务实验，获取了适用于该数据集的架构。消融实验和对比实验结果表明，本文提出的改进模型比原 AGNAS 模型具有更低的分类错误率和更稳定的分类性能。此外，综合考虑测试错误率、参数量和搜索时间多方面因素，本文方法比其他方法优势更为明显。

基金项目

在此特别感谢江西省自然科学基金对本文的支持：20224BAB202002。

参考文献

[1] Bello, I., Zoph, B., Vasudevan, V., *et al*. (2017) Neural Optimizer Search with Reinforcement Learning. *Proceedings of Machine Learning Research*, Sydney, 6-11 July 2017, 459-468.

[2] Real, E., Aggarwal, A., Huang, Y. and Le, Q.V. (2019) Regularized Evolution for Image Classifier Architecture Search. *Proceedings of the AAAI Conference on Artificial Intelligence*, **33**, 4780-4789. https://doi.org/10.1609/aaai.v33i01.33014780

[3] Pham, H., Guan, M., Zoph, B., *et al*. (2018) Efficient Neural Architecture Search Via Parameters Sharing. *Proceedings of Machine Learning Research*, Stockholm, 10-15 July 2018, 4095-4104.

[4] Liu, H., Simonyan, K. and Yang, Y. (2019) DARTS: Differentiable Architecture Search. *ICLR International Conference on Learning Representations*, New Orleans, 6-9 May 2019, 1-13.

[5] Dong, X. and Yang, Y. (2019) Searching for a Robust Neural Architecture in Four GPU Hours. 2019 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Long Beach, 15-20 June 2019, 1761-1770. https://doi.org/10.1109/cvpr.2019.00186

[6] Li, G., Qian, G., Delgadillo, I.C., Muller, M., Thabet, A. and Ghanem, B. (2020) SGAS: Sequential Greedy Architecture Search. 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 13-19 June 2020, 1617-1627. https://doi.org/10.1109/cvpr42600.2020.00169

[7] Xiao, H., Wang, Z., Zhu, Z., Zhou, J. and Lu, J. (2022) Shapley-NAS: Discovering Operation Contribution for Neural Architecture Search. 2022 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), New Orleans, 18-24 June 2022, 11882-11891. https://doi.org/10.1109/cvpr52688.2022.01159

[8] Xue, Y. and Qin, J. (2023) Partial Connection Based on Channel Attention for Differentiable Neural Architecture Search. *IEEE Transactions on Industrial Informatics*, **19**, 6804-6813. https://doi.org/10.1109/tii.2022.3184700

[9] Hu, X., Huang, L., Zeng, J., Wang, K. and Wang, Y. (2023) EGFA-NAS: A Neural Architecture Search Method Based on Explosion Gravitation Field Algorithm. *Complex & Intelligent Systems*, **10**, 1667-1687. https://doi.org/10.1007/s40747-023-01230-0

[10] Wang, R., Cheng, M., Chen, X., *et al*. (2021) Rethinking Architecture Selection in Differentiable NAS. *International Conference on Learning Representations*, Vienna, 4-8 May 2021, 1-18. https://doi.org/10.48550/arXiv.2108.04392

[11] Sun, Z., Hu, Y., Lu, S., *et al*. (2022) AGNAS: Attention-Guided Micro and Macro-Architecture Search. *Proceedings of Machine Learning Research*, Baltimore, 17-23 July 2022, 20777-20789.

[12] Park, J., Woo, S., Lee, J., *et al*. (2018) BAM: Bottleneck Attention Module. *British Machine Vision Conference*, Newcastle, 17-18 July 2018, 1-14. https://doi.org/10.48550/arXiv.1807.06514

[13] Ma, N., Zhang, X., Zheng, H. and Sun, J. (2018) ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. *Computer Vision—ECCV* 2018, Munich, 8-14 September 2018, 122-138. https://doi.org/10.1007/978-3-030-01264-9_8

[14] Zoph, B., Vasudevan, V., Shlens, J. and Le, Q.V. (2018) Learning Transferable Architectures for Scalable Image Recognition. 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, 18-23 June 2018, 8697-8710. https://doi.org/10.1109/cvpr.2018.00907

[15] Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L., *et al*. (2018) Progressive Neural Architecture Search. *Computer Vision—ECCV* 2018, Munich, 8-14 September 2018, 19-35. https://doi.org/10.1007/978-3-030-01246-5_2

[16] Xu, Y., Xie, L., Zhang, X., *et al*. (2020) PC-DARTS: Partial Channel Connections for Memory-Efficient Architecture Search. *International Conference on Learning Representations*, Addis Ababa, 26-30 April 2020, 1-13. https://doi.org/10.48550/arXiv.1907.05737