Particle Swarm Optimization for Compact Neural Architecture Search for Image Classification

摘要

卷积神经网络（CNNs）在深度学习领域中具有卓越的表现，可以广泛应用于图像识别、语音识别等任务。神经架构搜索（NAS）方法可以自动设计网络架构，其中许多NAS方法设计的新型CNN架构优于人工设计的架构。然而，当前大多数NAS方法存在一些问题：生成架构的计算复杂性过高，会影响深度模型的部署；架构设计存在限制，会影响架构设计的灵活性。为解决这些问题，提出了一种基于进化计算（EC）的紧凑且灵活的NAS方法。首先，对搜索空间进行设计，将移动反转瓶颈卷积块（MBConv block）作为基本组件，以确保紧凑架构的初始质量。其次，设计了一个两级变长粒子群优化（PSO）方法，用于演化CNN的微观结构和宏观结构。此外，通过集成多种计算减少的方法，大大加速了评估过程。最后，在CIFAR-10和CIFAR-100数据集进行实验，评估了所提出的方法的性能。

关键词：神经架构搜索；粒子群优化；卷积神经网络

1 引言

神经架构搜索（NAS）是一种自动化机器学习算法（AutoML） [1]，可以在预定义的搜索空间中发现新颖且性能良好的网络架构。到目前为止，已经提出了许多有效的NAS算法，可以自动设计新颖的CNN架构 [2]。在一些视觉任务，特别是图像分类上，NAS已经取得了人类无法超越的非常高的性能 [3]、 [4]、 [5]，同时节省了大量的人力，进一步推动了深度学习技术在更多领域的推广与应用。

在不同的搜索方案中，进化计算（EC） [6]是一种灵活可行的替代方案，在NAS中具有巨大的应用潜力。EC算法是一种基于群体的全局优化方法，对局部极小值不敏感 [7]，因此，它可以处理深度模型的架构设计。而粒子群优化（PSO） [8]是处理具有挑战性的优化问题的一种很有前景的EC算法。与基于EC的NAS中广泛采用的遗传算法（GA）相比，PSO以其简单的实现和快速收敛而受到越来越多的关注 [9]、 [10]。此外，在NAS中，有一些架构参数（例如，卷积层中的滤波器数量）在搜索空间中具有更广泛的线性排序可选值（称为序数）。PSO基于动量的平滑更新方式使得能够直观地捕捉此类架构参数在一定范围内的大小关系，并有效地搜索合适的值 [7]、 [11]。

尽管NAS取得了巨大的成就，但仍然存在很多缺点。例如，目前提出的很多方法使用了很多人为的想法去预定义架构的整体主干，来提高搜索架构性能的下限，但这种设计会

导致不能充分探索CNN架构的关键参数（深度、宽度和分辨率变化 [12]），并且仍然严重依赖CNN中的人类专业知识。此外，NAS算法通常涉及对候选架构的大量性能评估。每次性能评估通常需要在训练数据集上进行数百个梯度训练周期，这将导致总体计算成本极高。因此，可以开发一种灵活的PSO算法，用于高效、紧凑的CNN架构设计，并采用多种计算减少的方法，加快架构搜索过程。

2 相关工作

2.1 基于进化计算的神经架构搜索算法

由于灵活的编码和强大的搜索能力，基于EC的NAS在CNN架构的自动化设计领域发挥着至关重要的作用 [2]。GeNet [13]是一种早期的基于EC的NAS方法。它将搜索的架构分为多个阶段，每个阶段都有不同数量的与下采样（池化）操作连接的节点。在每个阶段，所有卷积层都具有相同的预定义设置。采用二进制字符串编码策略来表示节点间连接。进化过程遵循遗传算法的典型操作，即初始化、评估、选择、交叉和变异。为了进一步减少搜索空间，AmoebaNet [5]提出基于“NASNet搜索空间” [4]来搜索最佳CNN架构，该空间预定义了双链式主干并设计了所谓的可转移单元。进化后的AmoebaNet-A的性能略好于NASNet-A基线 [4]。

当转移到ImageNet [14]时，AmoebaNet实现了新的最先进的性能，超越了手工制作的模型。但计算复杂度仍然很高，为3150个GPU天。最近，针对DenseNet风格的可转移块演化提出了一种基于PSO的方法 [10]。密集块的层数和增长率以固定长度的方式进行编码，并且演进的块被重复堆叠以构建最终的CNN架构。为了降低搜索成本，该方法名为EffPNet，使用基于SVM的二元分类器作为代理模型，根据结构参数、损失和精度来预测新粒子与其局部最佳解之间的性能比较结果。早期训练时期。与大多数NAS算法类似，固定的宏架构剥夺了最终架构的某些多样性。这些促使我们开发一种灵活的两级PSO算法，以同时高效地演化具有宏观和微观组件的CNN架构。

2.2 自动紧凑型CNN架构设计

Elsken等人 [15]提出了一种用于多目标NAS的Lamarckian进化算法，称为LEMONADE算法，该算法进行了预测性能和资源约束的联合优化。Lu等人 [16]提出了NSGA-Net，将NAS任务阐释为一个多目标问题，旨在最小化两个目标：（1）分类错误和（2）浮点运算（FLOPs）的数量。Mnasnet [17]将模型延迟并入NAS的目标函数，并探索由MBConv块构建的分层搜索空间，从而得到非常高效的MobileNet样式架构。吴等人 [18]使用相同的基于MobileNet模板的搜索空间，并将包含复杂性约束的目标函数松弛为可微以提高搜索效率。崔等人 [19]提出在计算复杂度小于300M FLOPs的约束下搜索高效网络。搜索空间由具有不同扩展比例和连接方式的MBConv块构成，并通过双层优化搜索网络架构。Louati等人 [20]通过架构中块的数量和每个块的节点数来衡量架构复杂性，还使用了双层进化算法来解决紧凑型CNN架构设计问题。

最近，提出了一种基于遗传算法的NAS方法 [21]，用于在预设参数数量的约束下设计CNN架构。修改的MBConv块被纳入搜索空间以促进轻量级CNN设计。从上述文献中不难看出，轻

量级的MBConv块广泛用于设计紧凑的架构，这也是本文的情况。然而，许多其他方法明确涉及强约束，可能会影响搜索算法的灵活性。在这项工作中，我们关注分类性能和参数数量的优化。不同之处在于，我们构建了一个更轻量级的架构搜索空间，包括具有任意稀疏连接和更少滤波器的修改参数高效的MBConv块。所提出的方法可以更灵活地导航这样的搜索空间，以在架构搜索中不明确地获得紧凑的架构，而无需专门对架构搜索施加复杂性约束。

3 本文方法

3.1 总体框架

此部分对本文将要复现的工作进行概述，其中，CNN架构的粒子由两部分组成：（1）网络配置粒子（称为sp-conf）和（2）网络连接粒子（称为sp-conn）。总体框架如图1所示，首先对原始训练集进行下采样来生成两个缩减的训练数据集，分别用于架构搜索Dtrain和评估Deval。然后，使用小批量随机梯度下降法，将每个候选CNN架构在Dtrain上进行训练，并在Deval上进行评估，得到适应度，根据适应度值，利用提前停止策略进行早停判断。然后对架构进行更新进化，在进化循环中，首先更新种群的sp-conf，然后更新sp-conn。最后，一旦满足停止标准，就会输出最佳CNN架构A*（也称为EPCNAS-A），并在A*的基础上，改变滤波器的数量（增加一倍和四倍），得到两个变体EPCNAS-B和EPCNAS-C，并分别在训练集和测试集上进行充分训练和测试，以获得其在任务中的最终性能。

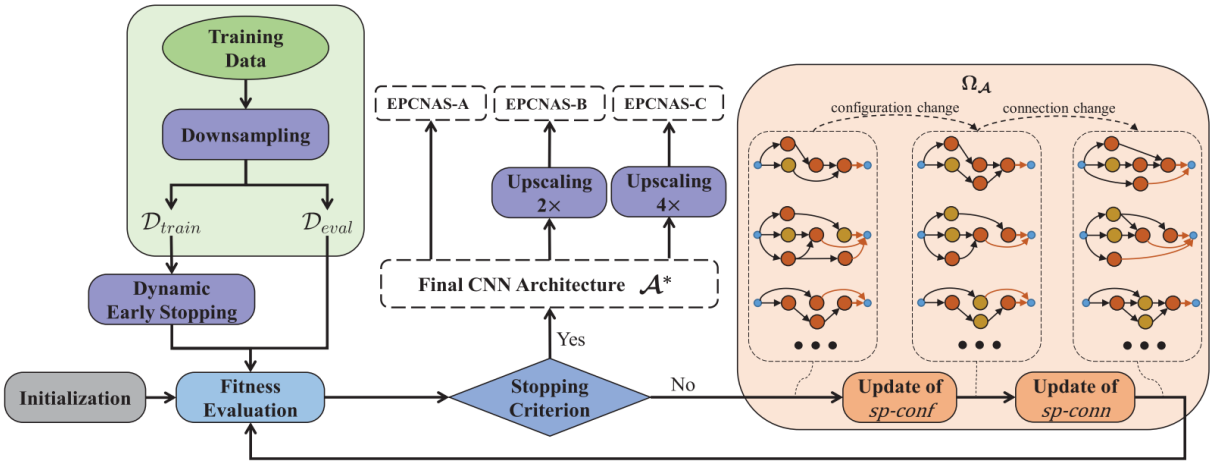

图 1. 总体架构

3.2 搜索空间

用于分类的典型CNN架构由三种类型的层组成：1）卷积层；2）池化层；3）全连接层 [22]。多个卷积层和池化层堆叠在一起以有效地从输入数据中提取特征，而全连接层通常放置在网络的尾部以实现分类。在这项工作中，我们将进化CNN中的整个卷积部分（即特征提取器网络）以进行图像分类。图 2描述了进化后的CNN架构的一个示例。目标CNN架构以标准卷积层开始，以全局平均池化(GAP)层加上两个全连接层结束。中间的灰色框是一个具有九个计算节点的可进化网络，其中每个节点代表一个具有不同配置的计算块/层，即节点5和7的池化层，以及其他节点的修改后的MBConv块。

与普通卷积模块相比，MBConv模块 [23]、 [24]是一种轻量级、高性能的计算范式。节点之间的连接结构为网状结构，编码灵活。理论上，这样的连接编码可以产生许多流行的网络架构，例如ResNet-like [25]和DenseNet-like [25]架构，并且搜索空间还包含更多未开发的新颖架构。每个节点的输入可以是任何先前节点的输出的串联，而那些没有输出连接的节点将被串联在一起作为进化特征提取器的最终输出（也是分类器的输入）。例如，在图 2中的CNN中，节点7-9的输出的串联将成为该网络分类部分的输入。

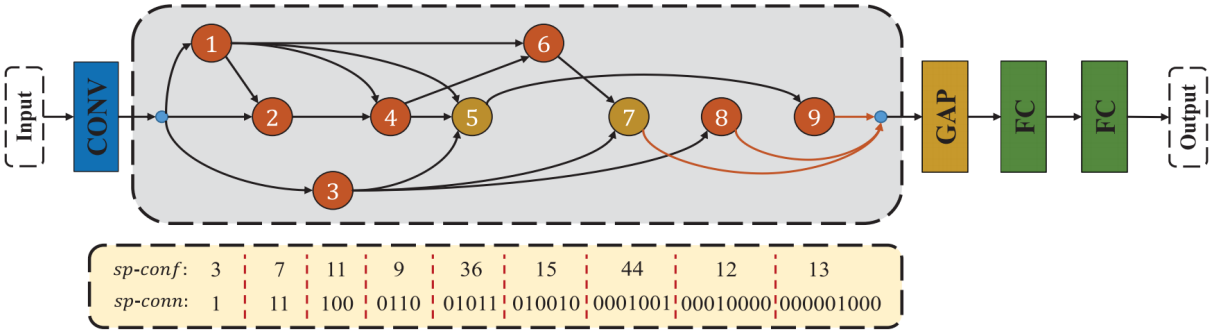

图 2. CNN架构示例图

目标CNN架构预计是紧凑的，即参数数量较少。为此，我们使用修改后的参数高效MBConv块作为构建块构建了一个有价值的搜索空间，使紧凑的架构搜索变得更加容易。修改后的MBConv块的架构如图 3所示。与MobileNetV3中的对应部分相比，删除了元素特征相加的快捷连接，这意味着修改后的MBConv块允许输入和输出特征图的数量不一致。

此外，SE块 [26]被高效通道注意（ECA）模块 [27]取代，该模块通过执行具有自适应内核大小的快速1D卷积，以更有效的方式捕获跨通道交互。该块采用通过1×1卷积层和d×d深度卷积层的输入通道来获得kb个特征图，然后将其输入ECA模块以学习通道注意力。最后，b个特征图将通过另一个1×1卷积层输出。一般来说，这样的MBConv块包含多种配置，即滤波器（或输出特征图）的数量b、DWConv层的内核大小d和扩展比率k。由于后两者的典型选择不多（d典型为3和5，k典型为3、4和6），而调整滤波器数量是探索灵活架构的更有效方法，因此可搜索架构这里修改后的MBConv块的参数仅为b，即特征图的数量。

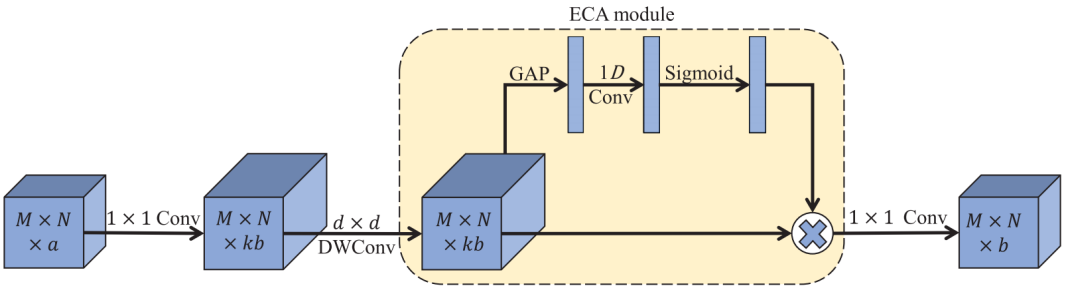

图 3. MBConv结构图

搜索空间设计具有以下特点。首先，搜索空间允许所提出的算法搜索具有不同数量节点的网络架构，这些节点指示最终模型的深度，以及每个MBConv块的滤波器数量反映网络宽度。此外，决定架构空间分辨率变化的池化层的数量、类型和位置被编码在搜索空间中。最后但并非最不重要的一点是，节点在目标架构中按顺序排列，每个节点都可以任意连接到其

先前的节点。这为灵活新颖的架构设计提供了更大的可能性。简而言之，所提出的搜索空间允许EPCNAS同时演化CNN架构的深度、宽度、空间分辨率变化和连接方式。

3.3 编码策略

如前所述，所有的块或层都被视为计算节点，这项工作涉及的可搜索架构参数包括：（1）网络深度（由进化节点的数量决定）；（2）节点配置（MBConv块的过滤器数量，池化层的池化类型）；（3）空间分辨率变化（池化层的数量和位置）；（4）节点之间的连接。显然，节点内配置信息与节点间连接信息不在同一级别。为了有效地搜索这些架构参数，本文针对架构演化过程提出了一种两级PSO方法。具体来说，代表完整CNN架构的粒子由两个子粒子组成，即用于网络配置的sp-conf和用于网络连接的sp-conn。此外，设计了一种微观更新方法来演化由sp-conf编码的（1）–（3）中的架构参数，而另一种宏观更新方法则负责演化由sp-conn编码的架构参数（4）。两种更新方法之间的分工和合作极大地降低了编码复杂性，而不会产生太多额外的计算开销。下面分别描述两个子粒子的粒子编码策略。

（1）sp-conf的编码：sp-conf的编码需要包含网络的深度信息和网络中各层的配置信息，因此它应该是一个变长的表示，并且粒子的每个维度都编码网络相应层的相关配置。在这里，我们采用我们之前的工作FPSO [28]中提出的灵活的变长粒子编码策略来进行sp-conf的编码，在相同的编码参数下完美地满足了上述的编码要求。有两个主要差异需要注意。

首先，FPSO中粒子的每个维度代表一个精确的标准网络层，而sp-conf的每个维度代表一个计算节点，要么是MBConv块，要么是池化层。第二个区别在于MBConv块的输出特征图数量范围设置为[1,16]，比FPSO中的卷积层小得多。这是因为这项工作中的最终架构包含许多用于特征图串联的快捷连接，并且在中间架构搜索阶段应用了缩减操作以降低计算复杂度。计算节点可以用二进制形式的六位表示，其中第一位用于区分节点类型，即0表示MBConv块，1表示池化层。有目的地添加第二位，以扩展搜索空间，以实现后续对不同粒子长度（即网络架构的深度）的探索。我们也称该位为操作位，因为一旦在标准PSO更新过程后变为1，就会通过分割操作进一步调整维度的长度，如图 4所示。

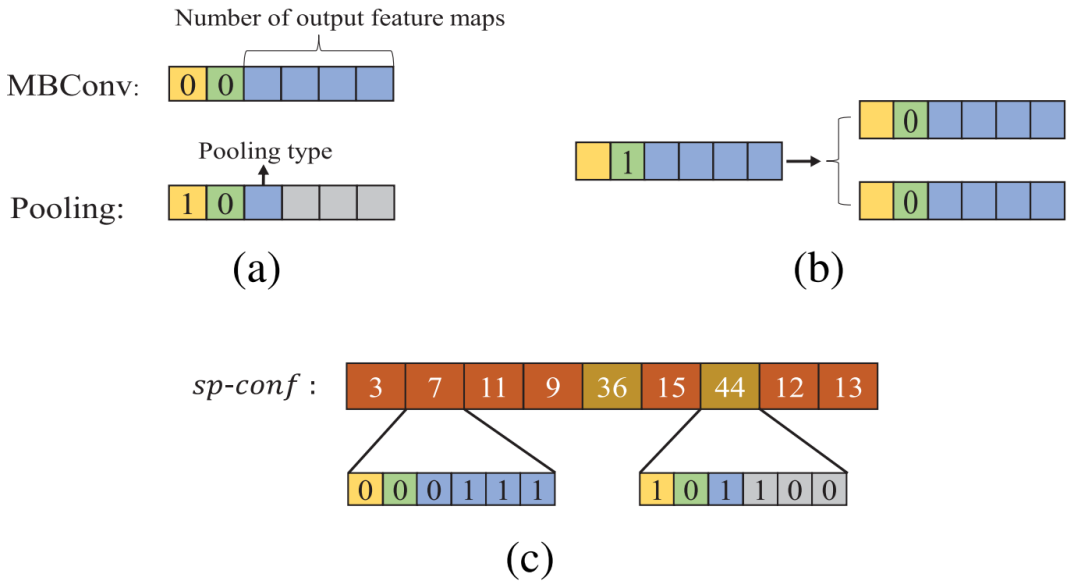

图 4. sp-conf编码策略

图 4(a)以二进制形式说明了正常MBConv块和池化层的6位表示，图 4(b)显示了分割操作。对于普通的MBConv块，其二进制表示的最后4位携带该块的输出特征图编号信息，而普通池化层的池化类型只有两个选项——1）最大和2）平均——可以用一位来表示。基于此编码，以十进制形式表示，MBConv块和池化层的编码范围分别为[0,15]和[32,47]。请注意，值15表示16个输出特征图，因为二进制字符串从0开始。图 2给出了sp-conf的示例，图 4(c)显示了其二维的相应二进制形式。

忽略长度因素，sp-conf的更新本质上是一种标准的连续PSO更新方法。根据所提出的编码策略，进化的sp-conf的所有维度都代表正常的MBConv块或池化层。标准PSO更新时，更新后的维度如果小于0，将被丢弃。否则，如果其二进制表示串的操作位变为1，则该维度将进一步分为二维（如图 4(b)所示），也相当于将对应的节点分割成代表两个正常块/层的两个节点。给定标准粒子更新的维度值s，不同节点类型和操作对应的编码范围以及调整后的参数请参见表1。特别是，如果维度值落在[16,31]或[48,+∞)范围内，则该节点将被分裂为两个相同类型的节点。注意，两次“分割池化层”操作在参数调整上是不同的。前一个操作产生的两个池化层具有不同的池化类型，而后者是相同的。sp-conf的编码策略为整个值空间中的每个维度值提供了合理的含义，这极大地方便了sp-conf的灵活更新。

（2）sp-conn的编码：关于承载节点间连接信息的spconn的编码，受到GeNet [13]的启发，并在GeNet的基础上进行了一个改进，使编码灵活性大大增强。我们进化了整个特征提取网络，其中涉及池化层，这更加通用和灵活，而GeNet仅搜索与池化层连接的不同阶段内的卷积网络，连接方式的示例如图 2所示。

第一个标准卷积层的输出是演进架构的输入，称为输入节点。计算节点按顺序排列，对于第i个节点，用i位二进制串编码其与输入节点和前i-1个计算节点的连接关系。例如，我们使用4位来表示输入节点（节点1-4）之间的连接。从图 2中可以看出，节点4与节点1和2连接，与输入节点和节点3断开，因此节点4的连接代码为0110。将连接组合起来得到最终的sp-conn演进网络中所有计算节点的代码。sp-conn是一个相当长的二进制字符串，其长度取决于演进的网络架构的深度。假设网络架构的计算节点数为num，则对应的sp-conn的长度为$num(num+1)/2$。显然，这也是一种变长表示，并且将利用所提出的基于BPSO的方法来有效地处理变长sp-conn的更新。

3.4 适应度评估

在迭代网络架构搜索阶段，需要评估每个新更新的粒子以获得其适应度值。典型的适应度评估涉及在训练集Dtrain（例如，梯度训练的e epoch）上充分训练解码后的CNN架构，然后在适应度评估集Deval上检查其性能。不难看出，这个过程占据了大部分计算量NAS算法的复杂度。为了减轻架构搜索和评估的计算负担，我们通过集成多种计算减少的方法，提出了一种有效的加速方案，这些方法是提前停止策略、图像下采样和架构缩小。

标准的早期停止策略使用少量的epoch而不是e个epoch来训练候选架构，并使用其评估的性能作为适应度值。然而，所搜索的架构的复杂性是多种多样的。我们希望为潜在的好的架构训练更多的epoch，并尽早放弃不好的架构。为此，通过将早期停止训练中的时期数更改为动态数字，开发一种改进的早期停止策略。

给定一个较小的$\varepsilon(\varepsilon \ll e)$，学习率计划在SGD的个体架构训练过程中在$\varepsilon$ epoch内衰减。训练期间误差曲线的小波动是允许的。若当前epoch距迄今为止最好的评估精度对应的epoch多

于$\lambda(\lambda < \varepsilon)$epoch，或者如果评估的精度当前epoch获得的架构的比迄今为止最好的架构低超过一定程度ψ，例如3%，因为该架构的训练被认为是不稳定的，并且在竞争最佳架构中的潜力较低建筑学。否则，该架构将继续训练，直到达到预定义的最大历元。在这种方法下，大多数较差的架构通常会在少于ε epoch后停止训练，而更好的架构的训练通常会超过ε epoch。作为粗略估计，所提出的动态提前停止方法使得所提出算法的计算成本为原始成本的ε/e，同时与标准提前停止方法相比大大提高了架构训练的效率。

图像下采样和架构缩减这两种低保真方法通过减少每个训练周期的计算量来加速架构评估过程。在架构搜索过程中将原始数据集下采样到较小的尺寸，例如将CIFAR-10 [29]的32×32图像调整为16×16的尺寸，以便训练和评估所需的内存和计算量在这个缩小的数据集上，候选网络可以大大减少。同样，我们缩小了搜索阶段的网络架构，即将MBConv块的滤波器数量的搜索范围压缩到25%，介于[1,16]之间。这样，搜索到的架构的尺寸就会非常小，这也可以显着提高适应度评估的效率。

假设下采样比和降尺度比分别为γ和η，那么通过结合这三个计算效率高的设置，搜索过程的总计算成本可以减少到原始的$\gamma^2\eta^2\varepsilon/e$。图 1描述了这三种技术在所提出的算法中如何工作。请注意，在最后的训练阶段，生成的架构将通过将每层的宽度分别乘以2和4进行两级缩放，最终输出总共三个架构。

3.5 粒子更新

一个完整的CNN架构由一个包含sp-conf和sp-conn的粒子共同表示。因此，网络架构的演化涉及到这两个子粒子的联合更新，这是由两级PSO算法完成的。如前所述，两级PSO方法包括演化配置的更新（即sp-conf）和演化网络拓扑的更新（即sp-conn），这两个过程在一次迭代中依次进行，然后进行适应度评估。

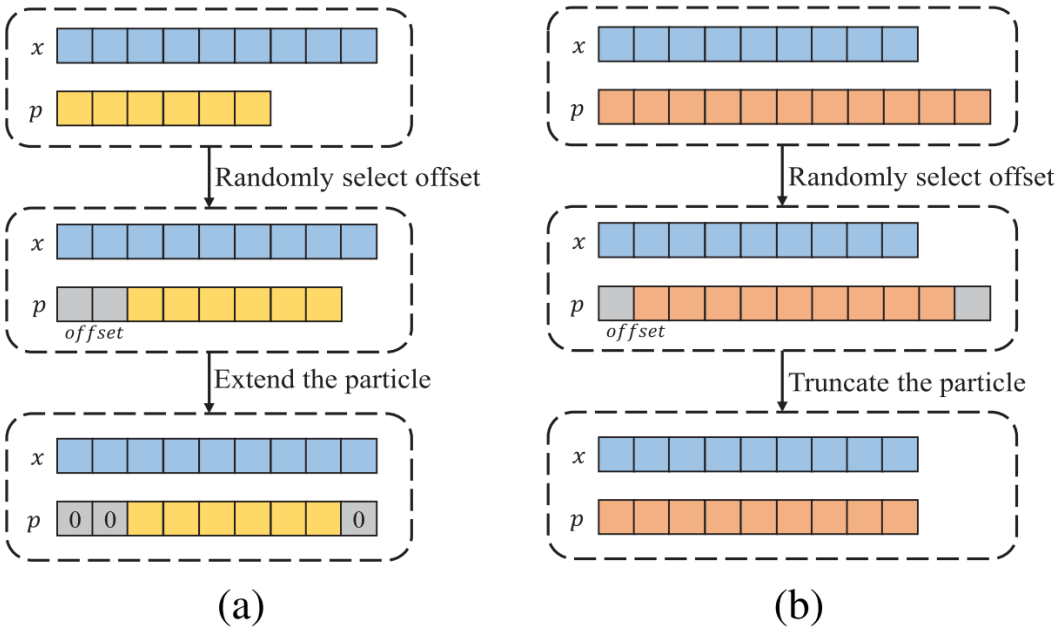

图 5. 粒子对齐操作

在演化过程的每次迭代中，首先进行sp-conf的更新，然后基于演化后的配置信息，更新连接拓扑sp-conn。sp-conf的每个维度都是十进制整数，因此我们可以使用标准连续PSO算

法，然后进行裁剪（向下舍入）操作来计算子粒子的速度和位置。sp-conn与sp-conf不同，sp-conn是一个二进制字符串。因此，在获得连续速度矢量后，子粒子sp-conn将由BPSO算法进一步更新。但由于两个子粒子的长度可变，无法直接计算标准形式的两个子粒子的速度矢量。因此，在计算速度矢量之前利用FPSO [28]中的粒子对齐操作来调整个人最佳解和全局最佳解的长度。

粒子对齐操作如图 5所示，主要包括两个步骤：偏移量选择和粒子长度调整。为每次计算设置不同的偏移量可以增加粒子更新的随机性和后续种群的多样性。具体操作为：给定当前粒子x和参考粒子p，从$[0, abs(len(x)-len(p))]$中随机选择一个偏移量，其中$abs(\cdot)$表示取绝对值，$len(\cdot)$是长度获取操作。然后，根据选定的偏移量，扩展p（如果短于x）或截断（如果长于x）以匹配x的长度。值得注意的是，sp-conf的更新不仅会演化基本块的配置，还可能带来网络长度的变化。在sp-conf更新之后、sp-conn更新之前，如果sp-conf维数有减少或增加，则需要对sp-conn进行相应调整，如图 6所示。

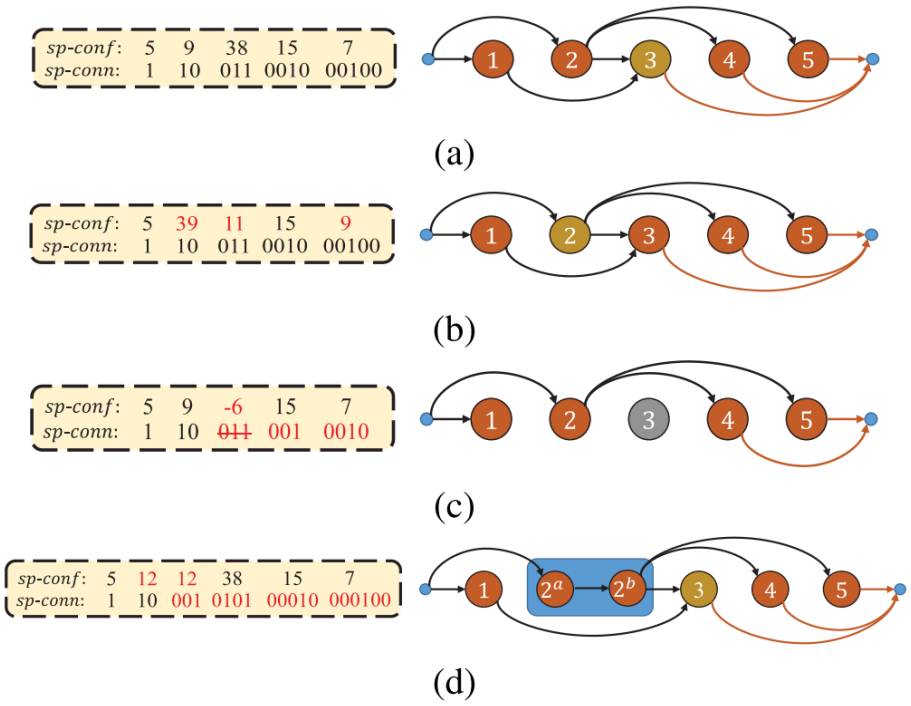

图 6. sp-conf更新示例

图 6(a)给出了粒子及其相应架构的示例，图 6(b)显示了不改变长度的粒子更新。如果更新过程中删除了某个节点，则需要从对应的sp-conn中删除与该节点相关的编码信息，如图 6(c)所示。如果一个节点被分成两个节点，那么这两个节点将被连接，并且原节点的输入和输出连接被分配给两个新节点，如图 6(d)所示。这两种更新方法将共同互补地为整个架构搜索做出贡献。更新过程之后，在解码和评估新架构时，如果节点接收到不同空间分辨率的特征图作为输入，我们将添加标准的跨步卷积操作来对具有更大分辨率的特征图进行下采样，以确保它们的串联尺寸为完全相同的。这可能是两次更新造成的，从中可以看出，架构参数——空间分辨率的变化，实际上是受到了两次更新过程的影响。

4 复现细节

4.1 与已有开源代码对比

复现原文代码已开源https://github.com/HuangJunh/EPCNAS。本次复现工作针对原文使用的CIFAR-10和CIFAR-100两个数据集进行复现。同时改进模型架构，引用了FR卷积结构，并在FR卷积结构的基础上进行了改进，使网络输入输出层可以不一致，符合本文算法的思路。通过引入FR卷积结构，提高了架构搜索的速率，降低了搜索时间。

4.2 实验环境搭建

所有复现和改进实验均在Windows系统中的Pycharm软件上执行，本文的实验使用了pytorch 11.1版本和python 3.8的环境，并分别在两个数据集：CIFAR-10、CIFAR-100上进行复现和改进实验。

4.3 复现细节

首先在减小的训练数据集Dtrain上进行训练，初始化种群粒子，然后使用两级粒子群对种群进行评估、更新，根据评估结果利用早停策略进行判断。继续进行迭代进化，直至满足停止准则（达到设定的迭代次数），输出最佳的CNN架构EPCNAS-A，并通过改变滤波器数量（加倍和四倍），得到两个扩展变体EPCNAS-B和EPCNAS-C，最后将得到的三个架构在训练集和测试集上进行完全训练和测试，得到最终结果。

4.4 改进思路

原论文架构中使用的基本块是MBConv3的结构，通过1×1卷积层和d×d深度卷积层的输入通道来获得kb个特征图，然后将其输入ECA模块以学习通道注意力。最后，通过另一个1×1卷积层输出b个特征图，如图 3所示。这个架构采用了ECA模块，可以很好的学习通道注意力，但它结构还是有点复杂，导致搜索效率较低。我对原文架构的基本块进行了改进，采用了FR卷积运算的结构 [30]，如图 7所示。

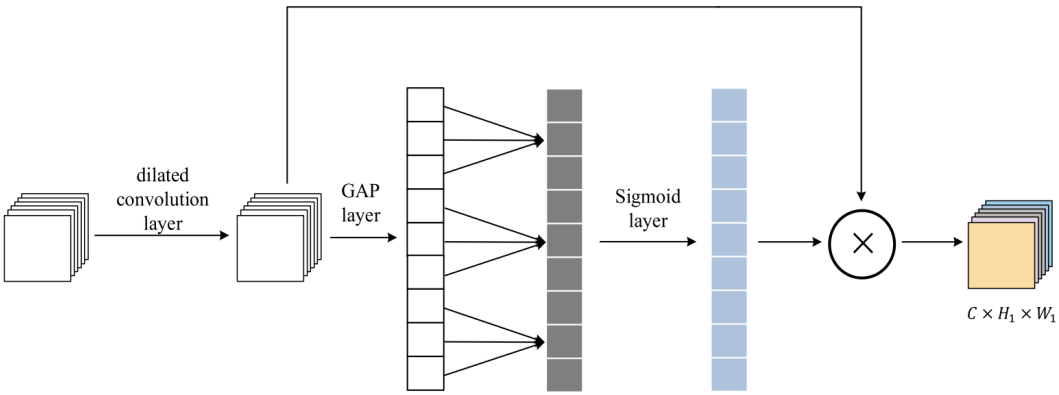

图 7. FR卷积结构

正方形块的数组表示输入特征，扩展的卷积层将输入特征图从$x \in \mathbb{R}^{C\times H\times W}$扩展为$T(x) \in \mathbb{R}^{C\times H_1\times W_1}$。GAP层之后是通道注意模块，然后是一个sigmoid函数层，该函数层使用大小为θ的

一维卷积来计算权重。这些矩形表示通道的特征映射，灰色矩形表示一维卷积的输出，蓝色的矩形表示通道的权重，彩色方块的数组表示FR操作的输出特征映射。我在FR卷积结构的基础上进行了改进，增加了1×1卷积层，使得输入和输出的通道可以不同，如图 8所示。

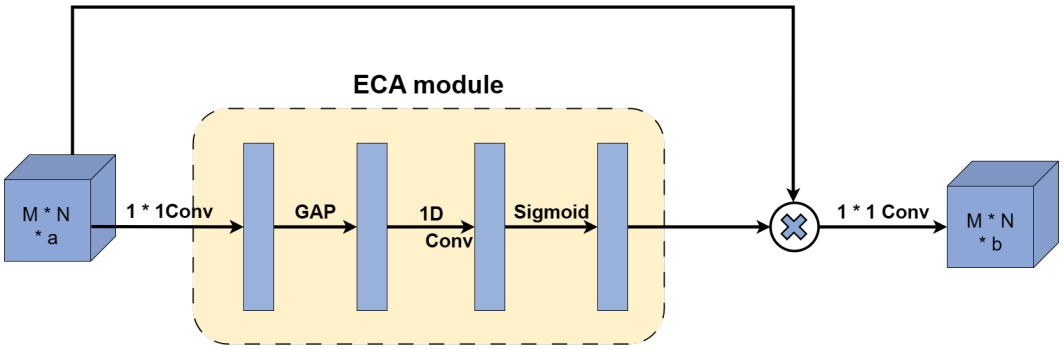

图 8. 改进的基本块结构

5 实验结果分析

5.1 复现结果

本文使用了两级PSO方法，用于演化CNN的微观结构和宏观结构。其中，在搜索阶段，对应的PSO算法的参数设置如表1所示：

表 1. PSO算法参数设置

参数	值
种群大小 α	20
进化代数 β	20
加速度常数 $c1$	1.49618
加速度常数 $c2$	1.49618
惯性权重 ω	0.7298
sp-conn的最大尺寸值	63

其中，在评估阶段，具体的实验参数设置如下表2所示：

表 2. 实验超参数设置

超参数	值
完全训练的轮次 e	450
早停训练轮次 ε	20
下采样比率 γ	50%
缩小比例 η	25%
与最优轮次的最大距离 λ	3
精度降低阈值 ξ	0.03

将EPCNAS搜索到的最佳的CNN架构及其两个变体EPCNAS-A，EPCNAS-B和EPCNAS-C，在训练集和测试集上进行完全训练和测试，得到最终结果。其中，在CIFAR-10数据集上的实验结果如下表3所示：

表 3. CIFAR-10数据集复现结果

方法	参数（M）	精度（%）	评估时间（GPU天数）
原EPCNAS-A	0.12	95.05	1.17
原EPCNAS-B	0.31	96.27	1.17
原EPCNAS-C	1.16	96.93	1.10
EPCNAS-A	0.12	94.79	0.98
EPCNAS-B	0.41	96.13	1.08
EPCNAS-C	1.52	96.69	1.54

由于实验设备、环境等方面的差异，在CIFAR10数据集上，复现的效果相对于原论文的效果，EPCNAS-A的精度相差了0.26%，但评估时间减少了0.19；EPCNAS-B的精度相差了0.14%，但评估时间减少了0.09；但在EPCNAS-C上，精度相差了0.24%，时间增加了0.44。但总体精度和时间相差不大。

在CIFAR100数据集上的实验结果如表4所示：

表 4. CIFAR-100数据集复现结果

方法	参数（M）	精度（%）	评估时间（GPU天数）
原EPCNAS-A	0.15	74.63	1.25
原EPCNAS-B	0.35	79.44	1.13
原EPCNAS-C	1.29	81.67	1.10
EPCNAS-A	0.12	73.28	1.01
EPCNAS-B	0.42	77.50	0.95
EPCNAS-C	1.51	79.92	1.25

由于实验设备、环境等方面的差异，在CIFAR-100数据集上，复现的效果相对于原论文的效果，EPCNAS-A的精度相差了1.35%，但评估时间减少了0.24；EPCNAS-B的精度相差了1.94%，但评估时间减少了0.18；但在EPCNAS-C上，精度相差了1.75%，时间增加了0.15。但总体精度相差1-2%左右，时间相差不大。

5.2 改进结果

使用图 7中改进的基本块结构进行实验，在CIFAR-10数据集上进行实验，实验结果如表5所示：

表 5. CIFAR-10数据集改进结果

方法	参数（M）	精度（%）	搜索时间（GPU天数）	评估时间（GPU天数）
EPCNAS-A	0.12	94.79	3.13	0.98
EPCNAS-B	0.41	96.13	3.13	1.08
EPCNAS-C	1.52	96.69	3.13	1.54
新EPCNAS-A	0.06	93.27	2.15	0.95
新EPCNAS-B	0.17	94.68	2.15	1.08
新EPCNAS-C	0.53	95.44	2.15	1.50

在CIFAR-10数据集上，改进后的架构减少了参数量和评估时间，大大减少了搜索时间（减少了将近一个GPU天），但精度减少了1-2%左右。

在CIFAR-100数据集上改进后的结果，如表6所示：

表 6. CIFAR-100数据集改进结果

方法	参数（M）	精度（%）	搜索时间（GPU天数）	评估时间（GPU天数）
EPCNAS-A	0.12	73.28	3.12	1.01
EPCNAS-B	0.42	77.50	3.12	0.95
EPCNAS-C	1.51	79.92	3.12	1.25
新EPCNAS-A	0.08	65.99	1.99	0.45
新EPCNAS-B	0.25	70.25	1.99	0.45
新EPCNAS-C	0.89	73.60	1.99	0.58

在CIFAR-100数据集上，虽然改进后的架构减少了参数量和评估时间，大大减少了搜索时间（减少了一个多GPU天），但精度减少了7-8%左右。可能改进后的架构对大数据集CIFAR-100不适用。

综上所述，改进后的架构可能更适用于不是很大的数据集，如CIFAR-10，但改进的架构可以应用在实验设置较差的实验环境，即对实验环境要求不高，可以大大减少实验时间，便于进行下一步实验。

6 总结与展望

EPCNAS可以用于演化紧凑型CNN架构，其中两级PSO算法以可变长度的方式演化CNN的配置和节点之间的连接，通过结合早停策略、下采样和架构缩减方法，显著降低了计算复杂度，该算法能够同时自动搜索网络架构的深度、宽度、空间分辨率变化和连接方式。在参数规模和搜索成本方面，EPCNAS优于许多手工设计的架构和自动NAS算法，并在“自动+手动”同行竞争者中取得了非常有竞争力的性能。复现实验中，CIFAR-10复现实验中精度和时间都相差不大，但由于设备的原因，在CIFAR-100复现实验中，精度相差1-2%左右，在时间上相差不大，因此复现工作基本实现了论文中的实验效果。

在改进实验中，改进后的架构可能更适用于不是很大的数据集，如CIFAR-10，但改进的架构可以应用在实验设置较差的实验环境，即对实验环境要求不高，可以大大减少实验时间，便于进行下一步实验。

未来还可以对EPCNAS进行进一步研究和改进,例如，可以使用不同的紧凑型神经网络（RNN、基于注意力的网络等）进行设计算法框架；该算法应用于多目标优化问题；可以使用更多的数据集进行实验，进一步验证算法的功能，如ImageNet。

参考文献

[1] Xin He, Kaiyong Zhao, and Xiaowen Chu. Automl: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212:106622, 2021.

[2] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G Yen, and Kay Chen Tan. A survey on evolutionary neural architecture search. *IEEE transactions on neural networks and learning systems*, 2021.

[3] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In *International conference on machine learning*, pages 2902–2911. PMLR, 2017.

[4] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

[5] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789, 2019.

[6] Thomas Bäck, David B Fogel, and Zbigniew Michalewicz. Handbook of evolutionary computation. *Release*, 97(1):B1, 1997.

[7] Ashraf Darwish, Aboul Ella Hassanien, and Swagatam Das. A survey of swarm and evolutionary computing approaches for deep learning. *Artificial intelligence review*, 53:1767–1812, 2020.

[8] Russell Eberhart and James Kennedy. Particle swarm optimization. In *Proceedings of the IEEE international conference on neural networks*, volume 4, pages 1942–1948. Citeseer, 1995.

[9] Yanan Sun, Bing Xue, Mengjie Zhang, and Gary G Yen. A particle swarm optimization-based flexible convolutional autoencoder for image classification. *IEEE transactions on neural networks and learning systems*, 30(8):2295–2309, 2018.

[10] Bin Wang, Bing Xue, and Mengjie Zhang. Surrogate-assisted particle swarm optimization for evolving variable-length transferable blocks for image classification. *IEEE transactions on neural networks and learning systems*, 33(8):3727–3740, 2021.

[11] Bin Wang, Yanan Sun, Bing Xue, and Mengjie Zhang. Evolving deep convolutional neural networks by variable-length particle swarm optimization for image classification. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2018.

[12] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[13] Lingxi Xie and Alan Yuille. Genetic cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1379–1388, 2017.

[14] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[15] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Efficient multi-objective neural architecture search via lamarckian evolution. *arXiv preprint arXiv:1804.09081*, 2018.

[16] Zhichao Lu, Ian Whalen, Vishnu Boddeti, Yashesh Dhebar, Kalyanmoy Deb, Erik Goodman, and Wolfgang Banzhaf. Nsga-net: neural architecture search using multi-objective genetic algorithm. In *Proceedings of the genetic and evolutionary computation conference*, pages 419–427, 2019.

[17] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2820–2828, 2019.

[18] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10734–10742, 2019.

[19] Jiequan Cui, Pengguang Chen, Ruiyu Li, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast and practical neural architecture search. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6509–6518, 2019.

[20] Hassen Louati, Slim Bechikh, Ali Louati, Chih-Cheng Hung, and Lamjed Ben Said. Deep convolutional neural network architecture design as a bi-level optimization problem. *Neurocomputing*, 439:44–62, 2021.

[21] Siyi Li, Yanan Sun, Gary G Yen, and Mengjie Zhang. Automatic design of convolutional neural network architectures under resource constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[23] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[24] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.

[25] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[26] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[27] Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Eca-net: Efficient channel attention for deep convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11534–11542, 2020.

[28] Junhao Huang, Bing Xue, Yanan Sun, and Mengjie Zhang. A flexible variable-length particle swarm optimization approach to convolutional neural network architecture design. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 934–941. IEEE, 2021.

[29] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[30] Haoyu Zhang, Yaochu Jin, Ran Cheng, and Kuangrong Hao. Efficient evolutionary search of attention convolutional networks via sampled training and node inheritance. *IEEE Transactions on Evolutionary Computation*, 25(2):371–385, 2020.