AS-NAS:自适应可扩展神经架构 强化进化搜索 深度学习算法

Tong Zhang , IEEE 会员， Chunyu Lei,Zongyan Zhang,Xian-Bing Meng 和 CL Philip Chen , 院士,IEEE

摘要 神经架构搜索 (NAS) 由于其非凸性,是深度学习设计中的一个具有挑战性的问题。为了解决这个问题,提出了一种基于增强的易经占卜进化算法 (IDEA) 和可变架构编码策略的自适应可扩展 NAS 方法 (AS-NAS)。首先,与典型的基于强化学习（RL)和基于进化算法（EA)的 NAS 方法不同,开发了一种简化的 RL 算法并将其用作强化算子控制器,以自适应地选择 IDEA 的有效算子。在没有复杂的 actor-critic 部分的情况下,基于简化 RL 的强化 IDEA 可以以更低的计算成本提高原始 EA 的搜索效率。

其次,提出了一种可变架构编码策略,将神经架构编码为固定长度的二进制字符串。通过同时考虑可变层、通道和不同卷积层之间的连接,深度神经架构可以扩展。通过与增强型 IDEA 和可变架构编码策略的集成,深度神经架构的设计可以自适应扩展。最后,所提出的 AS-NAS 与L1/2正则化相结合,以增加优化神经架构的稀疏性。实验和比较证明了所提方法的有效性和优越性。

稿件于 2020 年 8 月 27 日收到； 2020年11月26日和2021年2月3日修订； 2021 年 2 月 17 日接受。出版日期 2021 年 2 月 23 日;当前版本的日期为 2021 年 10 月 1 日。这项工作得到了中国国家重点研发计划的部分支持,资助 2019YFB1703600 和资助 2019YFA0706200;部分由中国国家自然科学基金资助 62076102、U1813203、U1801262 和 62006081 资助;部分由国家自然科学基金广东省杰出青年基金资助 2020B1515020041;部分由广州市科技重大专项资助 202007030006;部分由广州市科学技术计划资助 202002030250;部分由粤港澳大湾区脑科学与类脑智能研究中心基金 2019016 资助;部分由中国博士后科学基金资助 2020M672630。（通讯作者:孟宪兵;CL 菲利普·陈。）

Tong Zhang 就职于华南理工大学计算机科学与工程学院,中国广州 510006,琶洲实验室,广州 510335,中国（电子邮件:tony@scut.edu.cn）。

Chunyu Lei、Zongyan Zhang 和 Xian-Bing Meng 来自华南理工大学计算机科学与工程学院,中国广州 510006（电子邮件:axbmeng@gmail.com）。

CL Philip Chen 是华南理工大学计算机科学与工程学院,广州 510006,琶洲实验室,广州 510335,科技学院计算机与信息科学系,澳门大学,中国澳门（电子邮件：philipchen@scut.edu.cn）。

本文中一个或多个图形的彩色版本可在 https://doi.org/10.1109/TEVC.2021.3061466 获取。

数字对象标识符 10.1109/TEVC.2021.3061466

索引词 深度学习、易经占卜进化算法 (IDEA)、神经结构搜索 (NAS)、强化算子控制器、可变结构编码。

一、引言

我N 最近几年,神经架构搜索（NAS）引起了极大的研究兴趣 [1]-[3]。许多深度学习方法都是由 NAS 方法设计的,并已成功证明了它们的优越性 [4]-[7]。 NAS方法的设计对于实现自动化机器学习（AutoML)[1]、[8]具有重要意义。正如 [9] 中所建议的,具有强归纳偏差的神经结构可以在不学习任何权重参数的情况下执行某些任务。

因此,对NAS方法的研究具有实际意义和理论意义[1],[2]。

现有的NAS优化方法主要分为六类,包括： 1)进化算法（EA）； 2)强化学习（RL）； 3)基于梯度的方法； 4) 代理模型； 5)网格和随机搜索； 6)混合方法[1]、[2]、[8]。对于基于 EA 的 NAS 方法,神经结构首先被编码为个体,然后可以根据与相应神经结构 [5]、[10] 的验证精度相关的适应度值对其进行初始化和演化。在典型的基于 RL 的 NAS 方法中,使用额外的控制器网络来确定一系列运算符和连接令牌,并按顺序构建神经架构 [11]、[12]。虽然网格和随机搜索是两种简单的算法,但它们也是有用的 NAS 方法 [1]。与上述三种方法不同,基于梯度的 NAS 方法通过将神经结构的离散搜索空间放宽为连续搜索空间来提供解决方案 [13]。

由于神经架构的搜索空间通常非常大,并且评估优化架构的有效性在计算上非常昂贵,因此提出了替代模型并将其用作替代 NAS 方法 [14]。

典型的代理模型通常基于贝叶斯优化和神经网络 [1]、[15]。其他 NAS 方法主要是以上五种方法的集成,如 RL 和 EA 的集成 [6] 和基于代理模型的 EA [16]。尽管所有这些 NAS 方法都可以成功应用于设计深度学习方法,但它们仍然存在改进 [1]、[6]。这是一项持续的工作

设计有效的 NAS 方法来平衡其收敛性能、泛化性能和计算复杂性。

本文的动机是试图综合 EA 和 RL 的优点并设计一种高效且可扩展的 NAS 方法。基于 EA 的 NAS 方法的性能取决于 EA [5]、[10] 的特定运算符。不同的运营商可能有不同的表现。如果不考虑这些差异，基于 EA 的方法可能不会有很高的搜索效率。至于基于 RL 的 NAS 方法，它应该学习额外的控制器来逐层搜索架构 [11]。控制器应尝试各种动作以获得奖励，然后可以将其用作学习控制器和评估相应架构的监督方式 [6]。显然，学习过程效率不高。

事实上，有各种将 RL 和 EA 相结合的成功案例 [17]-[19]。然而，将 EA 和 RL 集成到 NAS 方法中仍然是一项正在进行的工作。

此外，另一个目标是设计一种可变架构编码策略，以提高 NAS 方法的泛化性能和可扩展性。特定的神经架构因不同的任务而异。架构参数包括层数、不同层之间的关系（U-Net [20]、全连接层等），以及与结构相关的参数（内核大小、通道等）。很难设计一种基于固定架构编码策略的 NAS 方法，该方法可以应用于学习不同的架构以解决不同的任务。因此，有必要研究可变架构编码策略。

在本文中，提出了一种基于强化易经占卜 EA (IDEA) 和可变架构编码策略的自适应可扩展神经架构搜索方法 (AS-NAS)，用于设计深度学习模型。在没有复杂的 actor-critic 部分的情况下，开发了一种简化的 RL 算法并将其用作增强的算子控制器来自适应地选择 IDEA 的算子，从而以较低的计算成本提高了 IDEA 的搜索效率。此外，还开发了一种可变架构编码策略来对神经架构进行编码。除了对具有可变层和通道的神经体系结构进行编码外，编码策略还可以设计不同卷积层之间的可变连接。此外，应用L1/2正则化 [21] 进一步增加 AS-NAS 方法的稀疏性。最后，进行了实验以研究AS-NAS的有效性。

本文的贡献主要可以概括为三个方面。

1)提出AS-NAS来自适应设计神经架构，增强优化神经架构的泛化性能和可扩展性。

2) 设计了一种简化的强化算法作为强化算子控制器来自适应地选择IDEA的不同算子以提高其搜索效率。

3) 开发了一种可变架构编码策略，以对具有可变层、通道和不同层之间的连接的神经架构进行编码。

本文的其余部分组织如下。
第二部分简要回顾了相关工作。有关所提出方法的技术细节在第 III 节中进行了说明。
第四节介绍了实验和比较研究。
最后，讨论了几个结论和未来的工作。

二。相关作品

这篇文章的主题是关于使用改进的IDEA设计NAS的方法。因此，我们主要从现有的 NAS 方法和 IDEA 的角度回顾相关工作。

A. NAS 方法

NAS方法的关键是设计搜索空间、架构优化器和模型估计方法[2]。现有的关于 NAS 的工作主要可以从这三个方面进行总结。在这三个主题中，模型估计方法强调了基于给定搜索空间评估优化神经架构性能的研究。由于直接使用优化架构训练模型然后在未见数据上估计其相应性能的计算量很大，因此提出了各种方法。模型估计方法主要包括五类[1]。第一类是指使用实际性能的较低保真度来加速模型评估过程的方法 [12]、[22]、[23]。第二种基于替代门模型，该模型学习曲线外推法来估计特定体系结构的性能，而无需直接评估体系结构 [15]。权重共享和提前停止标准也是有用的方法 [24]、[25]。另一类强调性能和相应资源预算之间的平衡 [2]，[26]。在本文中，我们关注搜索空间和架构优化器。因此，我们强调检讨这两个方面。

搜索空间定义了神经架构的表示格式。通过将神经体系结构视为有向无环图 (DAG)，可以将搜索空间的设计转化为设计 DAG。由于搜索空间决定了架构优化器可以搜索的候选神经架构，因此设计高效的搜索空间具有重要意义。现有关于搜索空间的工作主要分为四个方面[1]、[2]。直观的方法是直接定义搜索空间的整个架构[27]。

然而，直接搜索整个架构在计算上是昂贵的。此外，整个架构可能缺乏灵活性和可移植性。为特定任务学习的架构可能不会进一步用作另一个类似任务的先验 [1]，[11]。事实上，其他三种方法都是为了解决这些问题而提出的。基于单元的搜索空间侧重于学习高效单元和构建基于单元的体系结构，它可以进一步用于其他任务的先验或用于直接解决不同的任务 [7]、[12]、[22]。然而，基于小区的搜索空间也有局限性。学习到的单元将用于所有层，因此缺乏多样性。为了解决这个问题，设计了分层搜索空间。

例如，将进一步学习不同的细胞，并且

学习到不同的细胞将被用来构建不同的层[14],[26]。基于态射的搜索空间是另一种有前途的方法。通过从训练有素的网络中继承知识,它不仅可以处理任意非线性激活函数,还可以加速新任务的训练过程 [28]、[29]。

每个搜索空间都有自己的特点,并已证明其有效性。然而,可能仍存在进一步改进的空间。例如,遗传卷积神经网络 (CNN) 的显着特征是它可以在不同的基于卷积层的细胞之间建立可变的连接关系。然而,卷积核大小和通道数是固定的,无法优化[30]。此外,在设计的神经结构中,每个细胞存在的概率不等于 0.5,因此是否存在细胞不会因具体问题而异。受这项工作的启发,我们将研究如何解决上述问题并研究设计基于可变架构的搜索空间。

至于架构优化器,主要有六种
类别如前所述。

1)进化算法： EA 是随机算法,它模仿受自然启发的机制以迭代优化感兴趣的问题。设计 EA 的本质是平衡其探索和开发性能 [31]。有各种 EA [32],包括遗传算法 (GA) [10]、粒子群优化 (PSO) [5]、差分进化 [33]、IDEA [34]、鸡群优化 [35]、鸟群算法[36]等。对于基于 EA 的 NAS 方法,关键是设计高效的 EA 和适当的编码策略 [2]、[8]、[10]。NAS的本质是解决组合优化问题。所有 EA 变体都可用于解决组合问题。因此,理论上所有 EA 变体都可以用作 NAS 方法。

例如,提出了一种多目标遗传算法来解决 NAS,并已成功应用于解决 CIFAR-10 数据集 [10]。基于分解,PSO 也被应用于解决 NAS [5]。如何提高EA的搜索效率是基于EA的NAS方法研究的主要课题。在本文中,我们将研究如何使用 RL 来提高基于 EA 的 NAS 方法的搜索效率。

2)强化学习:基于 RL 的 NAS 方法的难点在于如何学习额外的控制器来逐层设计神经架构。已经设计了许多基于 RL 的 NAS 方法 [1]、[11]、[27]。一个典型的学习过程可以总结如下。首先,控制器用于生成特定的神经架构。其次,生成的基于架构的深度学习模型通过一定的参数优化器进行训练。然后,RL 可用于通过最大化学习模型在特定验证集上的预期准确性来学习和更新控制器 [27]。例如,Pham等人。 [11] 提出了一种基于长短期记忆的参数共享 NAS 方法。佐夫等人。 [12] 提出

一种基于递归神经网络的可迁移 NAS 方法。鉴于在学习神经架构之前应该学习额外的控制器这一事实,有必要研究如何降低基于 RL 的 NAS 方法的计算复杂度。

3)基于梯度的方法:通过将离散空间放宽为连续空间,基于梯度的方法可以用作NAS方法。这种方法的关键是如何制定搜索空间和相应的架构 [1]。开创性工作 DARTS 是通过使用 softmax 函数将离散空间松弛为连续空间 [7] 提出的。有许多工作进一步研究如何使用基于梯度的方法来解决 NAS 问题 [3]、[4]。例如,提出了一种修订的 DARTS,通过控制跳过连接操作的比例来规范搜索空间 [14]。可以考虑基于内存效率 [3] 来考虑另一种有用的方法来调整搜索空间。

4)代理模型:该方法的主要目标是学习有效的模型以加速寻找有前途的架构的搜索过程。传统的替代模型,包括神经网络、随机森林、贝叶斯优化方法和树结构的 parzen 估计器,在理论上都可以用作 NAS 方法 [1]、[2]。例如,Kandasamy等人。 [15] 使用贝叶斯优化方法作为替代模型来解决 NAS 问题。卡梅罗等人。 [37] 提出了一种基于随机森林的贝叶斯 NAS 方法。为了逐步预测架构,LSTM 也被用作替代模型 [14]。

5)网格和随机搜索:与基于EA的NAS方法不同,基于网格的NAS方法首先将搜索空间划分为规则的间隔,然后选择性能最佳的体系结构。至于基于随机搜索的 NAS 方法,他们随机探索给定的搜索空间以找到性能最佳的架构 [38]。如果没有明确的机制来学习和指导搜索过程,这两种方法的计算成本都会很高 [1]。

6)混合法:不同的方法各有千秋。综合不同的方法将是一个不错的选择。例如,随机森林被用作替代模型并集成到 EA 中作为 NAS 方法 [16]。基于 EA 和 RL 的 NAS 方法综合了这两种方法的优点 [1]、[6]。基于 RL 的突变控制器集成到 EA 中以自动优化神经架构。

在强化突变控制器的帮助下,EA 中的运算符可以比原始运算符更高效 [6]。虽然混合方法可以提高 EA 的搜索效率,但它也需要以昂贵的成本训练基于 RL 的变异控制器。为了进一步解决这个问题,本文设计了一种简化的强化算法并将其集成到 EA 中,作为一种高效的 NAS 方法。

B、理念

IDEA 的灵感来自易经转换。模仿《易经》卜卦中的繁、合、相变，设计了络、转、相三个算子。在IDEA中，搜索空间被称为卦空间H。卦空间中的个体被命名为卦hil， i = 1, 2, ..., N。 l是位串的长度， N表示十六进制语法空间的大小。 hil被编码为位串。六芒星状态ξ = (h1, h2,..., hn)T是状态大小为n的H的子集。在这里， n可以看作是人口规模 [34]，[39]。

对于M 元串， H中将存在Ml 个元素。
复杂运算符I可以表述如下：

$$我(hi) = \{我(hi1), 我(hi2), \ldots, 我(hij), \ldots, I(hil)\} \quad (1)$$

$$我（hij） = \begin{cases} \text{mod}M(hij + xij), & \text{如果rand} < pm \\ hij, & \text{否则} \end{cases} \quad (2)$$

其中j ∈ [1, l]，rand 表示范围为 0 到 1 的随机值。 xij(xij ∈ {0, 1, 2, ..., M − 1}) 是独立同分布的随机变量。 pm ∈ [0, 1] 是一个参数。

周转算子τ是易经中综合变换的推广，可以表述如下：

$$\tau\,(hijhij+1 \cdots hiq) = \begin{cases} hiq\ hiq-1 \cdots hij, & \text{rand} > 0.5 \\ hij\ hij+1 \cdots hiq, & \text{else} \end{cases} \quad (3)$$

其中j < q，1 ⩽ j ⩽ n，1 < q ⩽ n。 j和q是随机选择的。

易经中位的影响用互算子M简单表述，其数学表达式可描述如下：

$$M(hi) = \{M(hi1), M(hi2), \ldots, M(hij), \ldots, M(hil)\} \quad (4)$$

$$M(hij) = \begin{cases} hij+r, & 1 \leqslant j \leqslant \frac{l}{2} \\ hij-r, & \frac{l}{2} \leqslant j \leqslant l \end{cases} \quad (5)$$

其中r = round((1/6) × l)。

与其他 EA 一样，IDEA 的一个典型流程可以描述如下。首先，从卦空间中随机生成一个初始卦状态。其次，每个卦的适应度值由与特定问题相关的适应度函数来评估。然后使用精英主义策略选择父代来生成下一个六芒星状态，这是基于上述三个算子。

上述步骤在满足某个准则之前迭代执行。

三、提议的方法

自动设计高效深度学习模型的关键在于搜索空间和架构优化器。

因此，我们尝试从这两个方面来设计 NAS 方法。从搜索空间的角度，提出了一种基于可变架构的编码策略，以同时学习可变层、通道和不同层之间的连接。然后可以通过基于梯度的优化器对学习到的基于神经架构的模型进行训练。通过这种可变架构编码策略，学习到的神经架构可以很容易地扩展。从架构优化器的角度出发，开发了一种简化的基于 RL 的 EA，增强型 IDEA，以根据学习模型的性能反馈自适应地选择 IDEA 的不同算子，从而提高 IDEA 的搜索效率。借助这个强化的操作员控制器，学习到的神经架构也可以根据奖励反馈进行自适应优化。通过从搜索空间和架构优化器的角度整合这两种策略，神经架构的设计可以自适应扩展。

此外， L1/2正则化将进一步集成到 AS-NAS 中以增加其稀疏性。

A. 基于可变架构的搜索空间

神经结构可以表述为 DAG。因此，NAS的关键是设计层数、不同层之间的连接关系以及其他与结构相关的参数，如内核大小和通道等。为了解决这些问题，基于[30]中的编码策略提出了基于可变架构的编码策略框架。在所提出的编码策略中，设计的神经结构由一系列基于一些有序块的单元组成，并被编码成由三部分组成的固定长度的二进制字符串，而相应的二进制字符串在[30] 中只有一个部分。这里，第一部分表示特定小区是否存在。如前所述，在设计的神经结构 [30] 中，每个细胞存在的概率不等于 0.5。然而，这个新的附加部分可以有效地解决这个问题，使学习到的神经结构具有可扩展性，并随着特定问题而变化。第二个表示特定卷积层的输出通道。选择四个特定通道并分别编码为 00、01、10 和 11。最后一部分表示每个单元格中不同块之间的连接关系。这部分与[30]中的编码策略相同。因此，目标是解决组合优化问题，以找到由这三个部分制定的最佳神经架构。

为简单起见，卷积层和池化层用作基本块。选择卷积层和池化层作为块的另一个原因是它们被广泛使用并已成功证明其广泛的实用性 [30]、[40]、[41]。应该注意的是，可以选择不同的内核大小和池化策略。然而，在这篇文章中，每个卷积层的内核大小被简单地设置为 3×3。对于池化层，使用具有 2×2 窗口的最大池化。

所提出的编码策略的关键是对二进制字符串的第三部分中不同单元和块之间的连接关系进行编码。在提出的编码策略中，存在两种连接关系。第一个是不同cell之间的连接关系。 1表示对应的单元格存在，0表示对应的单元格不存在。如果两个特定的单元格都存在，则它们之间存在连接。为了确保特定的二进制有效，三个默认块被集成到这个神经架构中。

一方面，放置一个预定义的卷积层

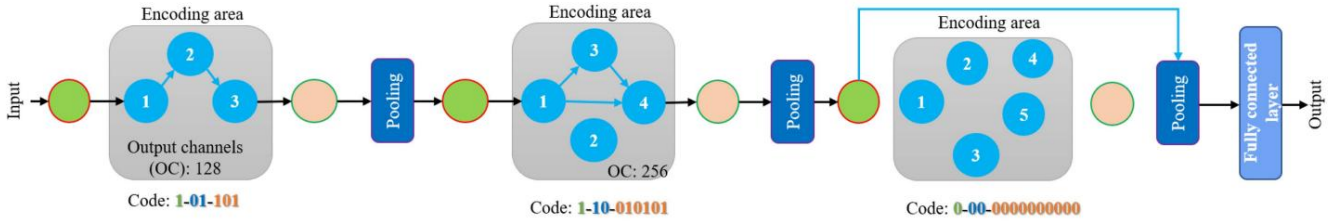

图 1. 编码策略的示例。

在每个单元格之前。另一方面,预定义的卷积层和池化层依次放置在单元之后。
如果不存在特定的单元格,则保留单元格之前的预定义卷积层和单元格之后的池化层。但是,单元格之后的卷积层将与单元格一起删除。此外,在每个卷积层中,卷积运算后还存在 ReLU。

预定义的全连接层放置在设计的神经架构的末尾。此外,优化后的神经结构的有效性也可以从优化策略中得到保证,这将在后面说明。

对于每个cell中不同block之间的第二种连接关系,可以归纳如下。考虑一个有m 个单元格的候选网络,第 i 个单元格中的块数为ni, i ∈ [1, m]。在每个单元中,存在几个有序块,它们是卷积层。只有编号较低的块才能连接到编号较高的块。

在第i个cell中,需要用0.5× ni(ni -1)个比特来编码block之间的连接关系,按照block的顺序依次编码。对于编码的二进制字符串,有ni -1 个部分。对于第一部分,该位表示第 i 个单元格中Gi,1和Gi,2之间的连接。至于第二部分,有两位表示Gi,1、 Gi,2、 Gi,3等之间的联系。对于最后一部分,存在ni − 1位表示Gi,1, Gi,2, ..., Gi,ni−1和Gi,ni之间的联系。给定第 i 个单元格中的两个块Gi,j1和Gi,j2 （j1 < j2）,如果对应的位等于 1,则Gi,j1连接到Gi,j2 ,即这两个块之间存在边。此外,块Gi,j2将Gi,j1的输出作为元素求和的一部分。

综上所述,整个编码后的二进制字符串由所有单元格对应的字符串组成,这些字符串按照每个单元格的序号依次串联起来。在每个单元格中,二进制字符串由固定长度二进制字符串的三部分组成。整个神经结构可以编码成长度为L = 0.5 (ni(ni − 1)) 的二进制字符串。需要注意的是,小区存在的概率为0.5,因此是否存在特定小区因具体问题而异。因此,神经架构的设计是可扩展的。

$$\text{我}=1$$

图 1 给出了所提出的编码策略的示例。存在三个单元将用于构建候选神经架构。每个单元格的代码由三部分组成。第一部分表示对应的单元格是否存在。可以明显看出,只有前两个单元格存在。第二部分表示输出通道。因为“01”表示128,所以第一个单元格的数字等于128。最后一部分编码每个单元格中不同卷积层之间的连接。这里以第一个小区为例。第一层和第二层之间有一条边,因此第一位等于1。因为第一层和第三层之间没有连接,所以第二位设置为0。因为第二层和第三层之间存在边层,第三位赋值为1。

因此,特定模型的相应神经结构可以说明如下。感兴趣的数据首先作为卷积层（绿色块)输入,然后将其输出输入到具有三个有序卷积层的第一个单元格中。第一个单元格的输出,即第一个单元格中第三个卷积层的输出,输入到后面的卷积层（浅橙色块)和轮询层,其输出再输入到后面的卷积层（绿色块） ,第二个单元格。因为第三个单元格不存在,所以第三个卷积层（绿色块)的输出将直接输入到第三个轮询层,其输出最终输入到一个全连接层。

最终的结果可以从全连接层得到。

B. 基于简化 RL 和L1/2正则化的强化 IDEA

在这里,IDEA 被用作架构优化器来优化神经架构。鉴于有2L个候选神经结构,搜索空间非常大,因此很难解决组合优化问题。为了提高 IDEA 的搜索效率和泛化性能,基于简化的 RL 和L1/2正则化设计了改进的 IDEA。

1)Reinforced IDEA Based on Simplified RL： NAS的关键是设计一个高效的架构优化器。然而,对于 EA,它的演化过程在很大程度上依赖于随机算子 [6]。在IDEA中,其三个算子是顺序实现的,没有有效的反馈来通过自适应选择高效的算子来指导演化过程。

而且,互通运营商也不适合设计的NAS。原因可归纳如下。一方面,编码后的比特串具有明确的含义。因此,互算子不适合仅使用部分比特来生成有效的神经架构。另一方面,至少有 4 位用于使用互操作符。然而,在编码的位串的几个部分中有2位和3位。为了解决这个问题,使用交叉算子来代替相互算子。就像 GA 中的交叉算子一样,有一个概率pc决定

一个个体将被另一个具有更好适应度值的个体的相应位所取代。如果没有比特定个体具有更好适应度值的个体,则对应的个体可以随机选择一个位进行改变。

为了进一步提高IDEA的搜索效率,提出了一种强化算子控制器并将其集成到IDEA中。和原来的IDEA一样,翻转和交叉算子也是在复杂算子的基础上实现的。但是,这两个运营商不会同时实施。他们的搜索性能将用作强化操作员控制器的反馈,以自适应地选择将实施的操作员。算子的优化性能越好,被选择的可能性就越大。如果周转算子的表现优于交叉算子,则周转算子将获得正强化奖励,从而增加选择周转算子的概率,反之亦然。为了设计这种增强型操作员控制器,增强型算法 [42]、[43] 的简化版本开发如下。

不失一般性,令yi = 1 表示第i 个个体选择翻转算子， yi = 0 表示第i 个个体选择交叉算子。这里， yi是一个伯努利随机变量,即yi只选择 0 和 1。

给定一个特定的yi， gi(yi, pi)是与pi相关的概率质量函数,可以表述如下：

$$gi(yi, pi) = \begin{matrix} pi, yi = 1 \\ 1 - pi, yi = 0。 \end{matrix} \quad (6)$$

在这里,开发了一种简化的强化算法来学习概率gi。考虑关于参数w 的关联即时强化学习任务。 w在每次试验收到强化值后进行调整。强化算法就是这样一种描述如何更新w的算法。

典型的强化算法是“Reward increment = Nonnegative factor * Offset reinforcement * char acteristic eligibility”的首字母缩写[42],可以用（7）式表示。在这篇文章中,我们简单地制定增强算法如下：

$$gi\ wi = \eta i \cdot (ri(t) - bi) \frac{\partial \ln}{\partial wi(t)} wi(t)$$
$$= wi(t - 1) + wi \quad (7)$$

$$pi(t) = f(wi(t)) = 1 + \exp(-wi(t)) \ pi(t) = pi(t - 1) + pi(t) \quad (8)$$

其中强化基线b设置为 0， ηi > 0 表示学习习率。 ri(t)是第 i 个个体在时间步t获得的奖励值。

鉴于一开始没有算子偏好,概率pi初始设置为0.5， wi设置为0。随着迭代过程的进行,这两个算子的偏好将逐渐形成。显然,奖励值越大,选择相应算子的概率越大,反之亦然。基于由 (6)、(7) 和 (8) 制定的建议的强化操作员控制器,选择特定操作员的概率是成正比的到相应的奖励值。这可以通过以下定理证明。

定理:考虑第i个人， ηi > 0,由8计算出的pi与ri成正比。

证明:根据（7）、(8)和链式法则,我们有 $\partial \ln gi \ \partial pi \ \partial pi \ \partial wi = yi$

$$\frac{\partial \ln gi}{\partial wi} = \frac{-pi。}{} = \frac{yi - pi}{-pi)} \times pi(1 - pi) \ pi(1 \quad (9)$$

将(9)式代入(7)式， wi(t)为

$$wi = \eta i \cdot ri(yi - pi)$$
$$wi(t) = wi(t - 1) + wi。 \quad (10)$$

对于特定的yi,相应的pi是确定的,因此yi − pi将是常数。结合ηi > 0 和(10),我们可以得出结论， wi与ri成正比。根据（8）， wi也与pi 成正比。

因此， pi与 ri 成正比。

如何确定奖励值ri(t)取决于问题。一般来说,开始阶段对适应度值的提升会大于后期的提升。为了公平对待这两个阶段， ri(t)被简单地表述为与当前和先前适应度值之间的差异以及当前和先前最佳适应度值之间的差异相关的函数。正如[43]一样,使用0、0.5、0.75和1四个不同的值来区分两个操作员的绩效反馈的不同效果。概率将每k 次迭代更新一次。

上述运算符将在整个编码二进制字符串上执行。同时,每个运算符将分别在编码的二进制字符串的三个部分上实现。因此,单独的优化器不能破坏由二进制字符串编码的神经结构的特定部分。因此,优化的神经架构是有效的。

2） L1/2正则化:虽然学习到的神经结构可以基于增强的IDEA和可变编码策略进行自适应扩展,但所提出方法的这种优越性是以大量参数为代价获得的。为了解决这个问题,将L1/2正则化与增强的 IDEA 相结合,以增加学习神经网络的稀疏性。一般来说， L1/2正则化比L0正则化更容易解决。而且,它比L1正则化更稳定,也可以生成更稀疏的结果 [21]。然而， L1/2正则化项在原点不可微,可能导致误差振荡。此处,平滑函数用于逼近非平滑函数|w|。 [44] 它可以表述如下：

$$f(w) = \begin{matrix} |w|, |w| \geqslant \varepsilon > 0 \\ -\frac{w4}{8\varepsilon 3} + \frac{4\varepsilon w2}{8} + \frac{1}{8\varepsilon}, |w| < \varepsilon \end{matrix} \quad (11)$$

其中 ε 是接近于 0 的正常数。

因此,损失函数可以表述如下：

$$= -\frac{1}{N} \sum_{i=1}^{否} \sum_{j=1}^{C} ci,j \log zi,j + \lambda f(w) \quad 1/2 \quad (12)$$

其中,此损失函数的第一项是交叉熵误差。 N表示一批样品的数量， C表示

算法一： AS-NAS

输入:卦象状态大小n,最大迭代次数Nmax,复杂概率pm,交叉概率pc，
η， k,λ,训练集和验证集Dtrain， Dvalid， batch size Bs,训练
epoch数Ep

输出:与其相应参数相关的优化神经结构

1初始化卦象和概率p
选择不同的运营商。
2对于每个epoch,对学习到的神经结构进行解码,以六边形表示,使用|Dtrain|/
Bs批样本作为输入,使用Adam基于损失函数优化学习到的神经结构的权重和
偏差。

3执行Ep epochs 后,使用|Dvalid|/Bs批样本作为输入,计算优化后的神经架
构的验证误差。验证错误被用作六边形的适应值。

4对每一个卦象,使用卦象周转
当p大于 0 和 1 之间的随机值时,算子生成新的卦。否则,使用基于复杂的交叉算
子生成新的卦。

5执行步骤2和步骤3计算所有卦的适应值,更新卦。

6每隔k更新每个卦的reward和p
迭代。
7执行步骤 4-6 直到达到Nmax次迭代。

类数， ci,j是one-hot fashion中的标签， zi,j表示特定网络经过softmax后的输出，λ(λ > 0)是参数。

鉴于 NAS 的优化是一个双层优化问题,开发了两阶段学习方案来优化神经结构和相应的参数。在训练阶段,损失函数用于评估优化编码神经结构的有效性。至于验证阶段,测试错误被用作标准来学习具有最佳验证性能的所选神经架构的参数。在这里，Adam [45] 用于训练优化神经架构的参数。在 Adam 中,beta1 = 0.9,beta2 = 0.999，epsilon = 1e-8。基于两阶段学习,最终可以优化神经结构和相应的参数。算法 1 总结了所提出方法的整个过程。

四、实验和比较验证

为了研究所提出的方法 AS NAS 的有效性,将从两个方面进行实验和比较。一方面,将进行消融实验以验证所提出方法的主要部分的必要性和有效性。另一方面,几种基于 NAS 方法的深度学习方法和设计良好的机器学习方法被用作比较进一步研究AS-NAS优越性的方法。
为了比较可信,比较方法的实验结果要么直接使用相应参考文献中的结果,要么使用从相应参考文献中获得的代码进行计算。

在实验中,基准和数据集包括 NAS-Bench-201 [46]、纽约大学对象识别基准(NORB) [47] 和 CIFAR-10。 NORB 由 50 种不同的 3-D 通用玩具的 48 600 张图像组成,属于五类:1)人类； 2) 动物； 3)汽车； 4) 飞机； 5) 卡车。每个图像有 2*32*32 像素。在这里,25 个对象的 24 300 张图像用于训练,其他图像用于测试。 CIFAR-10 由属于十个类别的 60 000 个 32×32 RGB 图像组成 [48]。 50 000 张图像用于训练,另外 10 000 张图像用于测试。 NAS-Bench-201 作为新的 NAS 基准测试,是 NAS-Bench-101 的扩展。它的搜索空间由四个节点和五个操作（归零、跳过连接、1×1 卷积、3×3 卷积和 3×3 平均池化层)组成,总共有 15625 个架构。为三个数据集提供了使用相同设置的训练日志和每个候选架构的性能,包括 CIFAR-10、CIFAR-100 和 ImageNet16-120 [49]。所有实验均在配备 Intel Xeon Silver 4214 CPU、8 GeForce RTX 2080 Ti 和 11-GB 主内存的服务器上进行记忆。

A. 消融实验

为了研究可变架构编码策略、强化 IDEA 和L1/2正则化的必要性和有效性,消融实验将分为两部分。第一个是使用 NAS-Bench-201 来验证增强后的 IDEA 的有效性。由于 NAS 基准具有固定的搜索空间并且只关注搜索算法本身,我们选择另一个数据集 NORB 进行第二个实验,以研究所提出方法的其他部分的有效性。

在消融实验的第一部分中,使用了包括几种 EA 和基本 IDEA 在内的七种方法作为比较方法。对于 IDEA 和强化 IDEA,状态空间大小pc、 pm、 k 和 η 分别为 20、0.4、0.6、4 和 0.1。进行了 500 次独立运行。然后,我们报告了 CIFAR-10、CIFAR-100 和 ImageNet16-120 数据集上即时验证和测试遗憾的平均性能。

如表 I 所示,可以清楚地看出,增强型 IDEA 在所有三个数据集上都优于基本 IDEA。与基本 IDEA 相比,增强型 IDEA 在 CIFAR-10 和 CIFAR-100 数据集上的准确率提高了 1%–2%。因此,我们可以安全地得出结论,增强的操作员控制器确实可以通过自适应选择有效的操作员来提供有效的反馈来指导进化过程。凭借强化操作员控制器的优点,强化IDEA也可以显示出其优于其他六种比较方法的优势。

为了进一步研究可变架构编码策略和L1/2正则化的有效性,进行了消融实验的第二部分。因为

表一

NAS-BENCH-201消融实验对比结果

Method	CIFAR-10		CIFAR-100		ImageNet16-120	
	valid	test	valid	test	valid	test
ENAS [11]	39.77±0.00	54.30±0.00	15.03±0.00	15.61±0.00	16.43±0.00	16.32±0.00
REA [22]	91.19±0.31	93.92±0.30	71.81±1.12	71.84±0.99	45.15±0.89	45.54±1.03
REINFORCE [42]	91.09±0.37	93.85±0.37	71.61±1.12	71.71±1.09	45.05±1.02	45.24±1.18
RS [50]	90.93±0.36	93.70±0.36	70.93±1.09	71.04±1.07	44.45±1.10	44.57±1.25
RSPS [51]	80.42±3.58	84.07±3.61	52.12±5.55	52.31±5.77	27.22±3.24	26.28±3.09
DE [52]	91.11±0.05	-	-	73.13±0.13	-	46.38±0.35
IDEA	91.22±0.16	93.84±0.21	71.64±0.46	71.92±0.33	45.71±0.18	44.76±0.59
Reinforced IDEA	91.71±0.01	94.01±0.26	73.61±0.07	73.99±0.29	46.99±0.37	46.90±0.38

表二

NORB消融实验的比较结果

Method	Parameters(M)	Test Error(%)
NAES[30] + reinforced IDEA	1.75±0.0180	7.05±0.6475
NAES[30] + reinforced IDEA + $L_{1/2}$	1.75±0.0109	10.87±0.5994
Ours without $L_{1/2}$	7.62±1.6831	4.93±0.3801
Ours	11.52±0.5279	6.14±0.3837

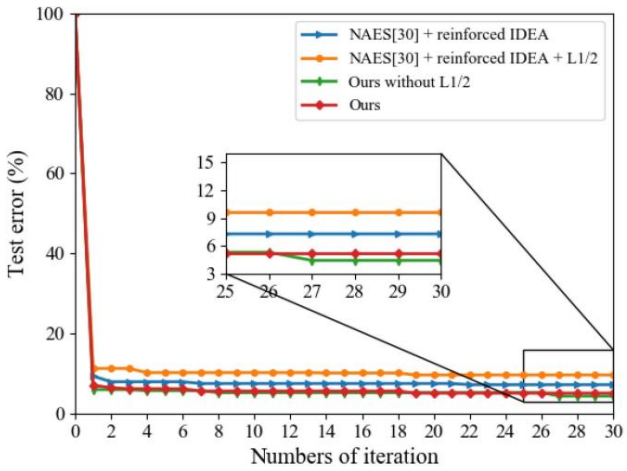

图 2. NORB 数据集四种比较方法的收敛曲线。

提出的编码策略基于[30]中的神经架构编码策略（NAES），选择NAES作为比较方法。基于增强的 IDEA，进行了比较实验，以研究是否存在可变架构编码策略和L1/2正则化对所提出算法的性能的影响。基于十次独立试验的统计结果用于分析比较结果。为了公平比较，所有比较方法对应的参数都是相同的。这里，候选神经结构由两个单元组成，其中的块数分别为 3 和 5。四个输出通道分别为 32、64、128 和 256。

表二给出了在 NORB 数据集上的比较结果。可以清楚地看到，我们的 AS-NAS 方法有和没有L1/2正则化都可以优于基于 NAES 的方法。如图 2 中一次试验的收敛曲线所示，我们的方法也可以比基于 NAES 的方法收敛得更快。这些结果清楚地表明所提出的可变架构编码策略的优越性。这些优势的原因总结如下。所提出的编码策略中候选神经结构的数量是219，而对应的NAES是213 。从理论上讲，从更大的搜索空间比从更小的搜索空间更有可能找到最优的神经架构。此外，NAES 只专注于优化每个单元格中不同块之间的连接。

对于所提出的编码策略，每个块中的输出通道也可以优化，而它们在 NAES 中是固定的。有了这些优点，我们的方法可以获得比基于 NAES 的方法更好的精度。至于我们的方法收敛性能的优越性，可以说明如下。一旦预定义了候选神经结构的细胞及其相应块，NAES 中特定细胞存在的概率就接近于 1。然而，在所提出的编码策略中，相应的概率等于 0.5。因此，所提出的编码策略中候选神经结构的多样性优于 NAES。此外，如果单元和块设计不当，NAES 生成的初始化神经架构很可能比所提出的编码策略表现更差，并且 NAES 生成的候选神经架构可能更可能落入局部最优。然而，对于所提出的编码策略，情况并非如此。更重要的是，增强的 IDEA 可以使用生成的神经架构的性能作为反馈来自适应地指导搜索过程。

因此，我们的方法比基于 NAES 的方法更有可能收敛得更快。

至于L1/2正则化，在表 II 中清楚地表明， L1/2正则化会导致我们的方法性能下降。此外，我们的具有L1/2正则化的 AS-NAS 方法学习的神经结构比没有L1/2正则化的具有更多的参数。原因是正则化方法不仅侧重于最小化训练误差，还强调提高稀疏性。然而，L1/2正则化可以有效地增加学习神经结构的稀疏性。如表 III 所示，采用L1/2正则化的 AS-NAS 学习到的神经结构的平均参数个数为 11.52M，其中只有 6.30M参数的绝对值大于 0.001，只有 4.31M参数的绝对值大于 0.001。绝对值大于 0.01。然而，对于没有L1/2正则化的AS-NAS ，相应的值分别为 7.62M、 7.56M和 6.99M 。调查

表三
NORB上稀疏神经网络的比较结果

Method	Parameters(M)	Test Error(%)	Parameters(>0.001)	Test Error(parameters>0.001)	Parameters(>0.01)	Test Error(parameters>0.01)
Ours without $L_{1/2}$	7.62±1.6831	4.93±0.3801	7.56±1.6691	4.93±0.3849	6.99±1.5437	4.97±0.3622
Ours	11.52±0.5279	6.14±0.3837	6.30±2.6354	6.14±0.3854	4.31±2.4314	6.35±0.4486

表IV
NORB的比较结果

Method	Parameters(M)	Test error (%)
DARTS [7]	3.34	7.8
NSGA [10]	0.08	7.4
Genetic CNN [30]	1.76	11.9
Stacked auto-encoders [53]	-	13.7
Deep belief nets [53]	-	11.5
Deep boltzmann machines [53]	-	10.3
Hierarchical ELM [53]	-	8.7
Broad learning [53]	-	10.7
Ours	3.75	5.97

为了考虑稀疏性对学习神经结构准确性的影响,小于 0.001 和 0.01 的相应参数被完全删除。如表 III 所示,参数大于 0.01 的压缩神经网络的准确率下降,而参数大于 0.001 的相应压缩神经网络的准确率几乎没有下降。因此,本文只考虑大于 0.001 的参数。可以清楚地看到,基于L1/2正则化的压缩神经网络比没有L1/2正则化的压缩神经网络具有更少的参数。

总之,可变架构编码策略和强化 IDEA 在优化神经架构方面发挥着重要作用。虽然L1/2正则化可能会导致所提出的方法性能下降,但它确实可以增加学习神经网络的稀疏性。如果我们关注神经网络的稀疏性,并且压缩神经网络的准确性略有下降,那么基于L1/2正则化的方法将是一种很好的替代方法,而不是没有L1/2正则化的方法。

B. 比较结果几种基于 NAS

的方法和其他精心设计的机器学习方法被用来进一步研究我们的 AS-NAS 的优越性。这里的超参数如下所示。对于 Adam,其 epsilon 参数为 10−8， beta 参数为 0.9 和 0.999,初始学习率为10−4。 batch size 和 epoch 分别是 50 和 30。 ε、λ 和 η 分别为 0.05、0.08 和 0.1。状态空间大小、 pc、 pm、 k 和最大迭代次数分别为 20、0.4、0.6、4 和 30。

如表 IV 所示,所提出的 AS-NAS 可以胜过解决 NORB 的所有比较方法。 AS-NAS 找到的最优神经结构如图 3 所示。

为了进一步研究 AS-NAS 的有效性和优越性,使用了广泛使用的数据集 CIFAR-10。超参数如下图所示。 epoch 和 λ 分别为 240 和 0.001。初始学习率为0.001，

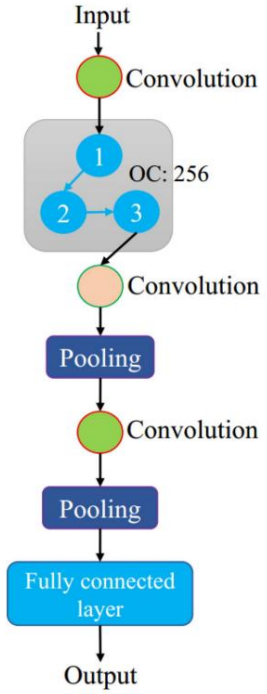

图 3. AS-NAS 获得的用于求解 NORB 数据集的最优神经结构。

在前 200 个纪元之前,它将减少 0.8 倍。其他超参数与上述实验中的相应超参数相同。候选神经结构由三个单元组成,其中的块数分别为 3、4 和 5。四个输出通道分别为64、128、256和512。候选神经结构的数量为228。然而,在[30]中对应的值为219 。理论上,所提出的方法可能比 [30] 中的方法更有可能找到最优的神经结构。如表V所示,也可以清楚地看到AS-NAS可以获得比其他方法最好的结果。 AS-NAS 实现的最佳神经架构如图 4 所示。

因为 CIFAR-10 的总数比 NORB 大,也比 NORB 复杂,所以在 CIFAR-10 上学习的神经结构的参数总数也比 NORB 大。 L1 /2正则化也可以有效地增加学习神经结构的稀疏性。如图 5 所示,我们的具有L1/2正则化的 AS-NAS 中大于 0.001 的参数仅占参数总数的 4.8%。

对于我们没有L1/2正则化的方法,相应的值为 98.5%。尽管L1/2正则化会导致我们的方法性能下降,但我们的具有L1/2正则化的方法仍然可以获得 2.23% 的测试误差并且表现优于其他比较方法。

表五
CIFAR-10的比较结果

Method	Test error (%)	Parameters (M)	Search method
DenseNet-BC [54]	3.46	25.6	manual
ENAS + cutout [11]	2.89	4.6	RL
NASNet-A+cutout [12]	2.65	3.3	RL
NAS + more filters [27]	3.65	37.4	RL
NSGA-Net [10]	2.50	26.8	evolution
AmoebaNet-B+cutout [22]	2.55	2.8	evolution
Genetic CNN [30]	7.10	-	evolution
Hierarchical evolution [55]	3.75	15.7	evolution
PNAS [14]	3.41	3.2	surrogate model
Sparse CNN [41]	7.47	-	gradient-based+regularization
PC-DARTS [3]	2.57	3.6	gradient-based
RDARTS [4]	2.95	-	gradient-based
DARTS(second order)+cutout [7]	2.76	3.3	gradient-based
diffGrad [13]	5.63	-	gradient-based
ResNet50 (LReLU) [13]	5.69	-	gradient-based
SNAS (single-level)+moderate constraint+cutout [56]	2.85	2.8	gradient-based
Ours	2.23	16.6	evolution+RL

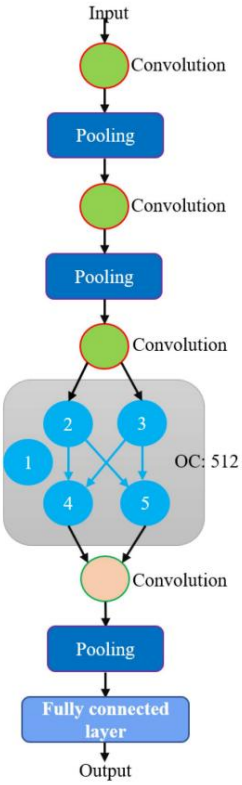

图 4. AS-NAS 获得的用于求解 CIFAR-10 数据集的最优神经结构。

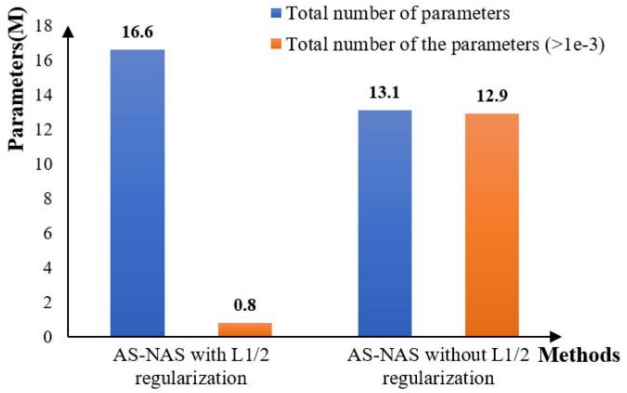

图 5.在 CIFAR-10 数据集上使用和不使用L1/2正则化的 AS-NAS 中的参数总数。

根据这些实验和广泛的比较,可以有把握地得出结论,AS-NAS 可以成功地用于设计高效的深度学习方法。

五、结论

在本文中,提出了一种基于增强 EA 和可变架构编码策略的自适应可扩展 NAS 方法。为了设计一种高效的NAS方法并提高其泛化性能和可扩展性,我们从搜索空间的角度提供了解决方案和架构优化器同时进行。一方面,开发了一种可变架构编码策略,以使用可变层、可变输出通道和不同卷积层之间的可变连接来模拟搜索空间。神经架构的设计可以是可扩展的。另一方面,开发了一个增强的算子控制器,并将其集成到IDEA中,以提高IDEA的搜索效率。通过与强化 IDEA、 L1/2正则化和可变架构编码策略的集成,所设计的 NAS 方法可以有效地平衡其效率和泛化性能,并具有良好的稀疏性。在 NORB 和 CIFAR-10 数据集上的实验清楚地证明了 AS-NAS 的有效性和优越性。

尽管所提出的工作取得了良好的效果并显示出良好的可扩展性,但仍有一些工作值得深入研究。

首先,可以进一步研究其他 EA 以用作 NAS 方法。此外,还可以进一步使用其他类型的深度学习方法,由AS-NAS自动设计。

事实上,所提出的强化 EA 是一个关于使用强化操作员控制器自适应地选择有效操作员的框架。增强的 IDEA 只是用作

插图。同样,除了卷积层,传统的前馈网络和LSTM也可以作为块。然后,所提出的方法可以进一步用于深度神经网络和深度 LSTM 的自动化设计。

其次,有必要进一步研究如何调整超参数和其他基于复杂架构的参数。在 AS-NAS 中,卷积层的内核大小、不同的激活函数和强化算子控制器中的超参数会影响所提出方法的性能。但是,它们都没有优化。优化这些参数的关键是如何对它们进行编码,提高算子的搜索效率。

最后但同样重要的是,可以进一步研究 AS-NAS 的可转移性。通过可变架构和L1/2正则化,AS-NAS 获得的优化神经架构可以进一步应用于求解各种数据集,并验证 AS-NAS 的泛化性和可迁移性。

参考

[1] X. He、K. Zhao 和 X. Chu,“AutoML:最先进的调查”,
2019. [在线]。可用:arXiv:1908.00709。

[2] T. Elsken、JH Metzen 和 F. Hutter,“神经架构搜索:一项调查”,2018 年。[在线]。可用:
arXiv:1808.05377。

[3] Y. Xu等人, “PC-DARTS:内存高效架构搜索的部分通道连接”, Proc。诠释。会议。学习。代表, 2020 年,第 1-13 页。

[4] A. Zela、T. Elsken、T. Saikia、Y. Marrakchi、T. Brox 和 F. Hutter,“理解和强化可微架构搜索”,Proc 。诠释。会议。学习。代表, 2020 年,第 1-28 页。

[5] J. Jiang、F. Han、Q. Ling、J. Wang、T. Li 和 H. Han,“基于分解的多目标粒子群优化的高效网络架构搜索”,神经网络,卷。 123,第 305-316 页,2020 年 3 月。

[6] Y. Chen等人, “RENAS:强化进化神经架构搜索”, Proc。 IEEE 会议。电脑。可见。模式识别,美国加利福尼亚州长滩,2019 年,第 4787–4796 页。

[7] H. Liu、K. Simonyan 和 Y. Yang,“DARTS:可微架构搜索”,2018 年。[在线]。可用:
arXiv:1806.09055。

[8] F. Hutter、L. Kotthoff 和 J. Vanschoren,自动化机器学习:方法、系统、挑战。瑞士 Cham:
Springer,2019 年。

[9] A. Gaier 和 D. Ha,“权重不可知神经网络”,神经信息处理进展。 Syst., 2019, pp. 5364–
5378, 温哥华, BC, 加拿大: Curran Associates, Inc.

[10] Z. Lu等人, “NSGA-Net:使用多目标遗传算法进行神经结构搜索”, Proc。热内特。进化。电脑。会议, 2019 年,第 419–427 页。

[11] H. Pham、MY Guan、B. Zoph、QV Le 和 J. Dean,“通过参数共享进行高效的神经架构搜索”,2018 年。[在线]。可用:arXiv:1802.03268。

[12] B. Zoph、V. Vasudevan、J. Shlens 和 QV Le,“学习用于可扩展图像识别的可迁移架构”,Proc 。 IEEE 会议。电脑。可见。模式识别,美国犹他州盐湖城,2018 年,第 8697–
8710 页。

[13] SR Dubey、S. Chakraborty、SK Roy、S. Mukherjee、SK Singh 和 BB Chaudhuri,
“diffGrad:卷积神经网络的优化方法”, IEEE Trans。神经网络。学习。系统,卷。 31,没有。 11,第 4500–4511 页,2020 年 11 月。

[14] C. Liu等人, “Progressive neural architecture search”, Proc。欧元。会议。电脑。可见。 (ECCV), 2018 年,第 19-34 页。

[15] K. Kandasamy、W. Neiswanger、J. Schneider、B. Poczos 和 EP Xing,“使用贝叶斯优化和最优传输的神经架构搜索”,神经信息处理系统进展。

美国纽约州红钩市:Curran,2018 年,2016-2025 页。

[16] Y. Sun、H. Wang、B. Xue、Y. Jin、GG Yen 和 M. Zhang,“使用基于端到端随机森林的性能预测器的代理辅助进化深度学习”,IEEE Trans 。进化。计算机,卷。 24,没有。 2,第
350-364 页,2020 年 4 月。

[17] MM Drugan,“强化学习与进化计算:混合算法调查”, Swarm Evol。计算机,卷。 44,
第 228-246 页,2019 年 2 月。

[18] Y. Huang、W. Li、F. Tian 和 X. Meng,“具有强化学习策略的适应度景观崎岖多目标差分进化算法”, Appl。软计算,卷。 96,第 1-13 页,2020 年 11 月。

[19] F. Zou、GG Yen、L. Tang 和 C. Wang,“动态多目标优化的强化学习方法”, Inf。科学,卷。 546,第 815-834 页,2021 年 2 月。

[20] T. Falk等人, “U-Net:细胞计数、检测和形态计量学的深度学习”, Nat。方法,卷。 16,
没有。 1,第 67–70 页,2019 年。

[21] Z. Xu、H. Zhang、Y. Wang、X. Chang 和 Y. Liang, “L1/2正则化”, Sci。中国信息网科学,卷。 53,没有。 6,第 1159–1169 页,2010 年。

[22] E. Real、A. Aggarwal、Y. Huang 和 QV Le,“图像分类器架构搜索的正则化演化”,
Proc 。 AAAI 会议。神器。
情报,卷。 33, 2019, 第 4780–4789 页。

[23] Y.-Q。 Hu, Y. Yu, W.-W. Tu、Q. Yang、Y. Chen 和 W. Dai,“通过传递级数扩展实现多保真度自动超参数调整”,Proc 。 AAAI 会议。神器。情报,卷。 33, 2019, 第 3846–3853
页。

[24] C. Wong、N. Houlsby、Y. Lu 和 A. Gesmundo,“使用神经 AutoML 进行迁移学习”,神经信息处理系统进展。美国纽约州红钩市:Curran,2018 年,第 8356–8365 页。

[25] E. Real、A. Aggarwal、Y. Huang 和 QV Le,“图像分类器架构搜索的老化演化”, Proc。
AAAI 会议。神器。 Intell., 2019,第 5048–5056 页。

[26] B. Wu等人, “FBNet:通过微分神经架构搜索进行硬件感知的高效卷积网络设计”,
Proc 。 IEEE 会议。电脑。可见。
模式识别,美国加利福尼亚州长滩,2019 年,第 10734–10742 页。

[27] B. Zoph 和 QV Le,“使用强化学习进行神经架构搜索”,2016 年。[在线]。可用:
arXiv:1611.01578。

[28] T. Wei、C. Wang、Y. Rui 和 CW Chen,“网络态射”, Proc。诠释。会议。马赫。学习,
2016 年,第 564–572 页。

[29] H. Jin、Q. Song 和 X. Hu,“Auto-keras:一种高效的神经架构搜索系统”,Proc 。第 25 届 ACM SIGKDD Int。会议。知识发现。
Data Min., 2019, pp. 1946–1956.

[30] L. Xie 和 A. Yuille,“Genetic CNN”, Proc。 IEEE 诠释。会议。电脑。
Vis.,意大利威尼斯,2017 年,第 1379–1388 页。

[31] X.-B。 Meng、XZ Gao、Y. Liu 和 H. Zhang,“一种具有栖息地选择和回声多普勒效应优化的新型蝙蝠算法”,
专家系统申请,卷。 42号17–18,第 6350–6364 页,2015 年。

[32] AN Sloss 和 S. Gustafson,“2019 年进化算法回顾”,载于遗传编程理论与实践 XVII。瑞士 Cham:Springer,2020 年,第 307–344 页。

[33] M. Kaur 和 D. Singh,“使用基于深度神经网络的多目标差分进化的多模态医学图像融合技术”, J. Ambient Intell。人性化计算,第 1-11 页,2020 年 8 月。doi: 10.1007/
s12652-020-02386-0。

[34] CLP Chen、T. Zhang、L. Chen 和 SC Tam,“易经占卜进化算法及其收敛分析”, IEEE Trans。
赛伯恩,卷。 47,没有。 1,第 2-13 页,2017 年 1 月。

[35] X. Meng、Y. Liu、X. Gao 和 H. Zhang,“一种新的仿生算法:鸡群优化”,Proc 。诠释。会议。 Swarm Intell., 2014 年,第 86-94 页。

[36] X.-B。 Meng、XZ Gao、L. Lu、Y. Liu 和 H. Zhang,“一种新的仿生优化算法:鸟群算法”, J. Exp。理论。
神器。情报,卷。 28,没有。 4,第 673–687 页,2016 年。

[37] A. Camero、H. Wang、E. Alba 和 T. Bäck,“使用无训练性能指标的贝叶斯神经架构搜索”,2020 年。[在线]。
可用:arXiv:2001.10726。

[38] P. Liashchynskyi 和 P. Liashchynskyi,“网格搜索、随机搜索、遗传算法:NAS 的大比较”,2019 年。[在线]。
可用:arXiv:1912.06059。

[39] T. Zhang、CLP Chen、L. Chen、X. Xu 和 B. Hu,“基于易经算子的高度非线性替换框的设计”, IEEE Trans。
赛伯恩,卷。 48,没有。 12,第 3349–3358 页,2018 年 12 月。

[40] T. Zhang、X. Wang、X. Xu 和 CLP Chen,“GCB-Net:图卷积广泛网络及其在情绪识别中的应用”, IEEE Trans。影响。计算机,抢先体验,2019 年 8 月 27 日,doi: 10.1109/
TAFFC.2019.2937768。

[41] Q. Xu、M. Zhang、Z. Gu 和 G. Pan,“通过对 CNN 的全连接层进行稀疏化正则化来补救过度拟合”,神经计算,卷。 328,第 69–74 页,2019 年 2 月。

[42] RJ Williams,“用于联结强化学习的简单统计梯度跟踪算法”, Mach。学习,卷。 8,没有。 3,第 229–256 页,1992 年。

[43] X.-B。孟,H.-X。李和 X.-Z。高,“用于结构设计问题的基于自适应强化学习的蝙蝠算法”,
Int。 J.
仿生计算机,卷。 14,没有。 2,第 114–124 页,2019 年。

[44] Q. Fan、JM Zurada 和 W. Wu，“具有平滑L1/2正则化惩罚的前馈神经网络在线梯度方法的收敛”， Neurocomputing,卷。 131,第 208-216 页,2014 年 5 月。

[45] DP Kingma 和 J. Ba，“Adam:一种随机优化方法”，
2014. [在线]。可用:arXiv:1412.6980。
[46] X. Dong 和 Y. Yang，“NAS-Bench-201:扩展可重复神经架构搜索的范围”,2020 年。[在线]。可用:arXiv:2001.00326。

[47] Y. Lecun、FJ Huang 和 L. Bottou，“姿势和光照不变的通用对象识别学习方法”,Proc 。 IEEE 计算。社会。会议。电脑。可见。模式识别,卷。 2, 2004, 第 97–104 页。

[48] A. Krizhevsky 和 G. Hinton，“从微小图像中学习多层特征”,Dept. Comput。科学,大学。多伦多,加拿大安大略省多伦多市,众议员 TR-2009,2009 年。

[49] P. Chrabaszcz、I. Loshchilov 和 F. Hutter，“图像网的下采样变体作为 cifar 数据集的替代方案”,2017 年。[在线]。
可用:arXiv:1707.08819。
[50] J. Bergstra 和 Y. Bengio，“超参数优化的随机搜索”， J. Mach。学习。水库,卷。 13，没有。 2,第 281–305 页,2012 年。
[51] L. Li 和 A. Talwalkar，“神经结构搜索的随机搜索和再现性”， Proc。第 35 个不确定性 Artif。智能。会议， 2020 年,第 367–377 页。

[52] N. Awad、N. Mallik 和 F. Hutter，“神经架构搜索的差异进化”,2020 年。[在线]。可用:
arXiv:2012.06400。
[53] CLP Chen 和 Z. Liu，“广泛的学习系统:无需深层架构的有效且高效的增量学习系统”， IEEE Trans。神经网络。学习。系统,卷。 29,没有。 1,第 10-24 页,2018 年 1 月。

[54] G. Huang、Z. Liu、L. Van Der Maaten 和 KQ Weinberger，“密集连接的卷积网络”，
Proc 。 IEEE 会议。电脑。可见。
模式识别,檀香山,夏威夷,美国,2017 年,第 4700–4708 页。
[55] H. Liu、K. Simonyan、O. Vinyals、C. Fernando 和 K. Kavukcuoglu，“高效架构搜索的层次表示”,2017 年。
[在线的]。可用:arXiv:1711.00436。
[56] S. Xie、H. Zheng、C. Liu 和 L. Lin，“SNAS:随机神经架构搜索”,2018 年。[在线]。可用:
arXiv:1812.09926。

张彤（IEEE 会员)于 2009 年获得中国广州中山大学软件工程学士学位,以及应用数学硕士学位和博士学位。分别于 2011 年和 2016 年获得中国澳门大学软件工程学士学位。

他目前是广州华南理工大学计算机科学与工程学院副教授,也是中国广州琶洲实验室的副教授。他一直致力于许多 IEEE 会议的出版事务。他的研究兴趣包括情感计算、进化计算、神经网络和其他机器学习技术及其应用。

Chunyu Lei于 2020 年获得中国郑州郑州大学计算机科学与技术学士学位。他目前正在中国广州华南理工大学攻读计算机科学与技术硕士学位。

他的研究兴趣主要包括神经结构搜索、计算智能和迁移学习。

Zongyan Zhang于 2020 年获得中国广州华南理工大学计算机科学与技术学士学位,目前正在攻读博士学位。计算机科学与技术学位。

他的研究兴趣主要包括多模态优化、神经结构搜索和计算智能。

Xian-Bing Meng获得博士学位。 2019年毕业于中南大学,中国长沙。

现为华南理工大学计算机科学与工程学院博士后研究员。

他的研究兴趣包括计算智能、机器学习和智能建模。

CL Philip Chen (Fellow, IEEE) 于 1985 年在美国密歇根州安阿伯市的密歇根大学获得电气和计算机科学硕士学位,并于 1985 年获得博士学位。 1988 年在美国印第安纳州西拉斐特的普渡大学获得电子和计算机科学学位。

他是讲座教授和院长
华南理工大学计算机科学与工程学院,广州,琶洲实验室,中国广州。作为美国工程技术教育认证委员会的计算机工程、电气工程和软件工程专业的项目评估员,他成功设计了澳门大学的工程和计算机科学专业,获得了华盛顿/首尔协议的认证通过香港工程师学会,其中被认为是他作为前科学技术学院院长对澳门工程/计算机科学教育的最大贡献。他目前的研究兴趣包括控制论、系统和计算智能。

陈博士于 1988 年获得母校普渡大学颁发的 2016 年杰出电气和计算机工程师奖。他于 2018 年因其在系统和控制论以及机器学习方面的贡献而获得 IEEE Norbert Wiener 奖。他还是 2018 年、2019 年和 2020 年 Clarivate Analytics 的高被引研究员。他目前是 IEEE TRANSACTIONS ON CYBERNETICS 的主编， IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE和 IEEE TRANSACTIONS ON FUZZY SYSTEMS 的副主编。他于 2012 年至 2013 年担任 IEEE Systems, Man, and Cybernetics Society 主席,并于 2014 年至 2019 年担任 IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS的主编。他是 TC 9.1 Economic 的主席2015年至2017年,国际自动控制联合会和商业系统。他是 AAAS、IAPR、CAA和HKIE的院士,欧洲科学院院士和欧洲科学与艺术学院院士。