·热点与综述· 文章编号: 1000-3428(2020)09-0016-11 文献标志码: A 中图分类号: TP391

基于无梯度进化的神经架构搜索算法研究综述

尚迪雅[1],孙 华[1],洪振厚[2],曾庆亮[1]

(1.新疆大学 软件学院 软件工程技术重点实验室,乌鲁木齐 830002;
2.深圳大学 光电工程学院 光电子器件与系统教育部/广东省重点实验室,广东 深圳 518060)

摘 要: 自动化深度学习是目前深度学习领域的研究热点,神经架构搜索算法是实现自动化深度学习的主要方法之一,该类算法可以通过对搜索空间、搜索策略或优化策略进行不同定义来自动设计神经网络结构。阐述进化算法和进化神经网络的发展历程,分类介绍以进化算法为搜索策略实现神经架构搜索的方法和过程,并比较基于进化算法的不同神经架构搜索算法的特点和现状,在此基础上,对神经架构搜索算法的搜索空间、搜索策略以及算法的未来发展方向进行探讨和展望。

关键词: 神经架构搜索; 自动化深度学习; 进化算法; 搜索策略; 进化神经网络

开放科学(资源服务)标志码(OSID):

中文引用格式: 尚迪雅,孙华,洪振厚,等. 基于无梯度进化的神经架构搜索算法研究综述[J]. 计算机工程,2020,46(9):16-26.

英文引用格式: SHANG Diya, SUN Hua, HONG Zhenhou, et al. Review of research on neural architecture search algorithms based on non-gradient evolution[J]. Computer Engineering, 2020, 46(9): 16-26.

Review of Research on Neural Architecture Search Algorithms Based on Non-Gradient Evolution

SHANG Diya[1], SUN Hua[1], HONG Zhenhou[2], ZENG Qingliang[1]

(1. Key Laboratory of Software Engineering Technology, School of Software, Xinjiang University, Urumqi 830002, China;
2. Key Laboratory of Optoelectronic Devices and Systems of Ministry of Education and Guangdong Province, College of Optoelectronic Engineering, Shenzhen University, Shenzhen, Guangdong 518060, China)

【Abstract】 Automated deep learning is one of the new research hotspots in the field of deep learning. Neural architecture search algorithms are frequently used for the implementation of automated deep learning, as they can automatically design neural network structure by defining different search space, search strategy or optimization strategy. This paper introduces the development history of evolutionary algorithms and evolutionary neural networks. Then it introduces different methods and processes of using evolutionary algorithms as the search strategy to implement neural architecture search, and compares the features and development status of these neural architecture search algorithms. On this basis, this paper discusses the search space, search strategy and future development direction of neural architecture search algorithms.

【Key words】 neural architecture search; automated deep learning; evolutionary algorithm; search strategy; evolutionary neural network

DOI: 10.19678/j.issn.1000-3428.0057520

0 概述

近年来,随着人工智能技术的快速发展,自动化机器学习(Automated Machine Learning, AutoML)[1-2]已成为机器学习领域[3-5]的一个热点研究方向,并广泛应用于计算机视觉[6]、数据挖掘[7]、图像分割[8]和自然语言处理[9]等领域。

机器学习包括统计机器学习和深度学习2个方面[10]。统计机器学习在处理问题时一般包括数据预处理[11]、特征工程[12]和模型选择[13]等内容,这些操

基金项目: 新疆维吾尔自治区自然科学基金(2015211C263)。

作者简介: 尚迪雅(1995—),女,硕士研究生,主研方向为图像处理、自动化机器学习; 孙 华(通信作者),副教授、博士; 洪振厚,硕士; 曾庆亮,硕士研究生。

收稿日期: 2020-02-27 修回日期: 2020-04-23 E-mail: xj_sh@163.com

作均需要手动完成。在深度学习领域，神经网络结构在通常情况下也需人工设计，为了得到一个效果较好的神经网络结构，往往需要耗费很多时间和精力。为解决上述问题，AutoML 应运而生。AutoML 包含针对统计机器学习的自动化算法和针对深度学习的自动化算法 2 个部分。针对统计机器学习的自动化算法包括自动特征工程[14]和自动模型选择[15-16]等内容。本文的研究对象是针对深度学习的自动化算法，即自动化深度学习。自动化深度学习算法是一种特殊的机器学习算法，其可以获得高性能且十分灵活的网络结构。自动化深度学习算法用概念组成的网状层级结构来表示网络世界，每一个概念由更简单抽象的概念相连而成，且前者可通过后者计算而得。自动化深度学习根据当前问题的特点，通过自动搜索网络结构得到更多样且性能更好的神经网络结构，由此减少人工设计网络所花费的时间和人力并提高搜索效率和神经网络的性能。

网络架构自动搜索方法包括多种搜索策略，以进化算法为搜索策略的神经网络架构搜索(Neural network Architecture Search，NAS)算法，按照复杂程度可分为基于神经元的神经架构搜索算法、基于 CNN 的神经架构搜索算法和基于 DNN 的神经架构搜索算法。本文分析并归纳这 3 类算法的特点和应用现状，并对神经架构搜索算法的未来发展方向进行展望。

1　神经网络架构搜索

随着深度学习技术的发展，网络结构由最初的 LeNet[17](7 层结构)发展到后来的 AlexNet[18]、VGGNet[19]、GoogLeNet[20-21](22 层结构)以及 ResNet[22](152 层结构)。网络结构越来越深，模型也变得越来越复杂，虽然这些模型足够灵活，但人工设计网络结构仍然需要大量的专业知识以及充足的时间，而且针对深度学习模型的调参优化也非常繁琐。众多的超参数和网络结构参数会产生爆炸性的组合，常规的随机搜索和网格搜索效率较低，因此，效率较高的 NAS 和模型优化引起了学者们的广泛关注。

NAS[23]可以针对特定数据集从初始时刻设计性能良好的模型。一般 NAS 问题可以被定义为搜索空间和搜索策略 2 个子问题。搜索空间定义了被优化问题的变量和参数等，其代表一组可供搜索的神经网络架构。搜索策略定义了使用什么算法可以快速、准确找到最优的网络结构参数配置。常见的搜索方法包括随机搜索、基于贝叶斯优化、基于进化算法、基于强化学习[24-25]和基于可微分神经架构搜索[26-27]等。本文以基于进化算法的 NAS 为研究对象，阐述进化算法、进化神经网络的发展历程，分析近年来较受关注的若干基于进化算法的神经架构搜索算法的特点和应用现状。

2　进化算法和进化神经网络

2.1　进化算法的发展

进化算法是一类算法的统称，为一个“算法簇”。进化算法借鉴与模拟生物界的多种生物现象来实现搜索过程，一般包括基因编码、种群初始化、交叉变异算子、经营保留机制等基本操作。进化算法簇主要包含遗传算法(Genetic Algorithm，GA)、进化策略(Evolutionary Strategies，ES)和进化规划(Evolutionary Programming，EP)，后期又提出了遗传规划(Genetic Programming，GP)。

遗传算法[28]是模拟达尔文生物进化论中自然选择和遗传学机理的生物进化过程的计算模型，是一种通过模拟自然进化过程搜索最优解的方法。

进化策略[29]是一种模仿生物进化的求解参数优化问题的方法。与遗传算法不同的是，进化策略采用实数值作为基因，总是遵循均值为零、方差为某值的高斯分布变化规律来产生新的个体，然后保留最优的个体。

进化规划是在人工智能技术研究过程中提出的一种有限状态机进化模型，在此模型中机器状态基于分布的规律进行编译。文献[30]对进化规划进行改进，使其可处理实数空间的优化问题，并在变异运算中引入正态分布变异算子，使得进化规划成为一种优化搜索工具并广泛应用于很多实际问题。进化规划模拟生物种群层次上的进化，因此，在进化过程中主要强调生物种群行为上的联系，即依据种群层次上的行为进化而建立父、子代间的行为链，优秀的子代才有资格生存。

遗传规划[31]是一种自动搜索程序的算法，该算法是一种新的全局优化搜索算法，简单通用、鲁棒性强且对非线性复杂问题展现出很强的求解能力，因此被广泛应用于许多不同的领域。

从适应度的角度而言，遗传算法可用于选择优秀的父代(优秀的父代产生优秀的子代)，而进化规划和进化策略则用于选择子代(优秀的子代才能存在)。遗传算法与遗传规划强调的是基于编码基因的链式优化与筛选，而进化规划和进化策略更着重于结果选择。

近年来，对于遗传算法、进化策略、进化规划和遗传规划的定义一直存在较多争议，且遗传算法的应用较为广泛，因此，在很多情况下，对于进化算法而言并没有进行明确地区分，通常将遗传算法称为进化算法。

进化算法是一种无梯度的优化算法，先随机生成一个种群(N 组解)，然后循环执行选择、交叉、变异，直到满足最终条件后停止循环。进化算法尽管有多个重要分支，但它们却有着共同的进化框架。假设 P 为种群，t 为进化代数，$P(t)$ 为第 t 代种群，则进化计算的基本结构可以描述如下:

```
{
确定编码形式和搜索空间;
初始化各进化参数,并设置进化代数 t = 0;
初始化种群 P(0);
计算初始化种群的适应度;
While(不满足终止条件) do
{
t = t + 1;
利用选择操作从 P(t - 1) 代中选出 P(t) 代群体;
对 P(t) 代种群执行进化操作;
计算 P(t) 代种群的适应度值;
}
}
```

2.2 进化神经网络

1989 年 MILLER 等人提出神经进化的概念,利用遗传算法进行神经网络的进化,通过搜索行为空间可以在特定任务中取得更好的结果[32]。

神经网络是一种对人类智能结构的模拟方法,其通过对大量人工神经元的广泛并行互联,构造人工神经网络系统从而模拟生物神经系统的智能机理。进化计算是一种对人类智能的演化模拟方法,其通过对生物遗传和演化过程的认知,用进化算法去模拟人类智能的进化规律。

进化神经网络将进化算法的进化自适应机制与神经网络的学习计算机机制有机结合在一起,实现对动态环境的自适应。进化神经网络包括链接权值、网络结构和学习规则的进化,图 1 所示为神经网络的进化过程。

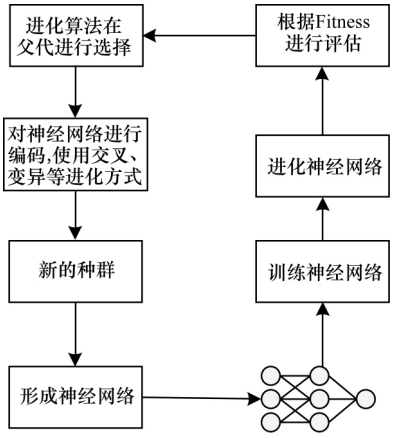

图 1 神经网络的基本进化过程

Fig. 1 Basic evolutionary process of neural network

3 基于无梯度进化的神经架构搜索算法

基于无梯度进化的 NAS 按照复杂程度可以分为基于神经元的神经架构搜索、基于 DNN 的神经架构搜索和基于 CNN 的神经架构搜索。

基于神经元的神经架构搜索是指神经网络结构是以神经元为基本单元,一个节点就是一个神经元,然后由神经元的堆叠来构造神经网络结构。其进化过程则是通过改变神经元之间的链接权重或拓扑结构,从最简单的单个神经元的结构开始搜索,直至得到符合当前问题的最优结构为止。

基于 DNN 的神经架构搜索是指当神经网络结构越来越复杂时,为了提高搜索效率,基本单元由固定的 DNN 层组成,一个节点就是一个 DNN 层。其进化过程则是通过一种共同进化的方式,增加神经网络的可复用性和可迁移性。

基于 CNN 的神经架构搜索是指随着 CNN 的广泛应用,为了能够自动构建 CNN 架构,而将 CNN 中的卷积层、池化层等操作作为基本单元,节点表示卷积操作、池化操作等操作类型。其进化过程主要采用经典的遗传算法,主要步骤包括初始化、选择、变异、交叉和适应度评估等。

3.1 基于神经元的神经架构搜索算法

经典的基于神经元的神经进化方式可以分为 2 种,一种是固定神经网络的拓扑结构,改变权重值,另一种是神经网络的拓扑结构和权重值都改变,从而实现神经进化。

3.1.1 神经网络拓扑结构固定的方式

在神经网络拓扑结构固定的方式中,通过改变权重值来进化神经网络结构。如图 2 所示,在一个全链接的神经网络结构中,网络拓扑结构是指一个隐藏的神经元层,每个隐藏的神经元分别链接到每个网络的输入和每个网络的输出,进化可以通过不断尝试变异的方式来修改链接中间的权重值和偏置值,从而改变神经网络的预测结果。

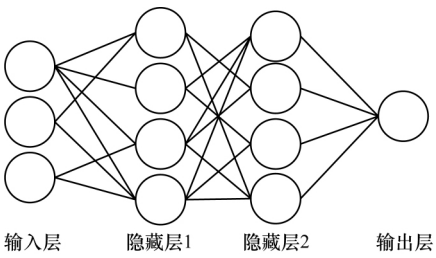

图 2 全链接神经网络结构

Fig. 2 The architecture of fully connected neural network

3.1.2 权重及其形态改变的方式

链接权重并不是影响神经进化的唯一参数,神经网络的拓扑结构也会影响其效果。将遗传算法与神经网络相结合的经典算法之一是增强拓扑的神经进化网络(Neuro Evolution of Augmenting Topologies, NEAT)算法[33]。当用一个很复杂的神经网络来解决一个简单的问题时会造成层结构的浪费,因此,NEAT 分析需要使用多少链接,忽略其中无用的链接从而构成更小的神经网络结构并提高运行效率。NEAT 包括修改权重、添加节点和链接 3 种突变方式,在现有链接中插入节点。NEAT 算法的基因编码、神经网络变异和交叉的具体过程如下:

1) NEAT 的基因编码方式

NEAT 的基因编码方式如图 3 所示,其中,Node

Genes 是指节点类型，用于存储网络中节点的信息，包括输入节点、隐藏节点和输出节点; Connect. Genes 是指链接，用于存储节点间的信息，包括输入节点、输出节点、权重值(Weight) 、连接方式(直接连接(Enabled) 和间接连接(DISABLED)) 以及创新号(Innovation ID) ，其中，创新号在交叉过程中作为识别标准。

2) NEAT 的神经网络变异方式

NEAT 中的神经网络变异包括链接变异和节点变异 2 种类型。链接变异为添加链接，既链接 2 个以前未链接的节点，如图 4(a) 所示，新增加节点 3 和节点 5 之间的链接。在节点变异中，现有的链接被拆分，新节点被放置在旧链接之间的位置，旧链接被禁用，既旧链接变为 Disable，2 个新的链接被添加到基因组中，如图 4(b) 所示，在节点 3 和节点 4 之间增加了节点 6，因此，节点 3 到节点 4 的链接变为 Disable，新增节点 3 到节点 6 和节点 6 到节点 4 的链接。

3) NEAT 的神经网络交叉方式

NEAT 的神经网络交叉主要通过 Innovation Number“对齐”父母的基因序列，若某链接父母都存在，则随机选择一个传给子代，该基因称为匹配基因; 若某链接只存在父母一方，则将存在的那个基因传给子代，该基因称为不匹配基因。不匹配基因又分为过剩基因和不相交基因，均表示基因组中不存在的结构。孩子的基因序列应是父母基因的并集。图 5 所示为 NEAT 基因的交叉操作，通过 Innovation ID 匹配不同网络拓扑的基因组，Parent1 和 Parent2 在节点和链接上具有相似的结构，但是它们也有区别，Innovation ID(显示在每个基因的顶部) 能够显现哪些基因相匹配，对于相匹配的基因则选择随机遗传，如果有不匹配的基因，则继承具有更好适应度的亲本。这样即使没有任何拓扑分析，也可以创建一个新的结构，将双亲的重叠部分以及它们不同的部分进行组合。

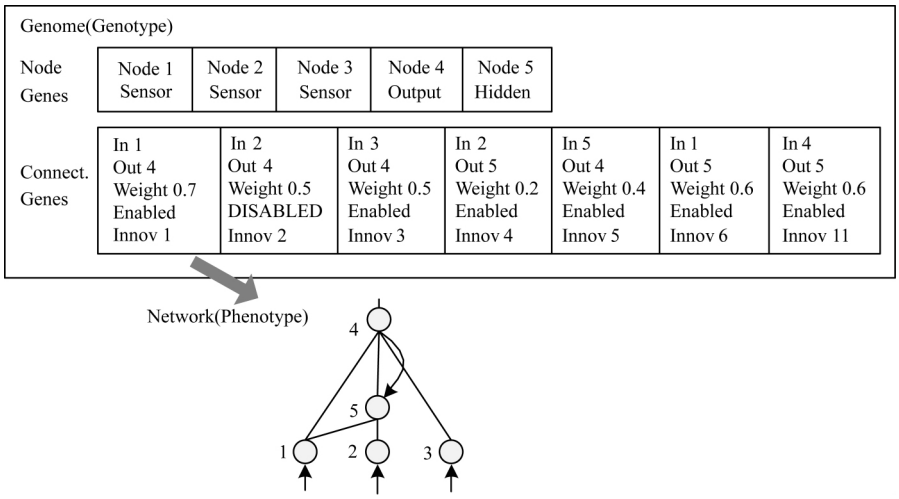

图 3　神经网络在 NEAT 中的表现形式

Fig. 3　Representation of neural network in NEAT

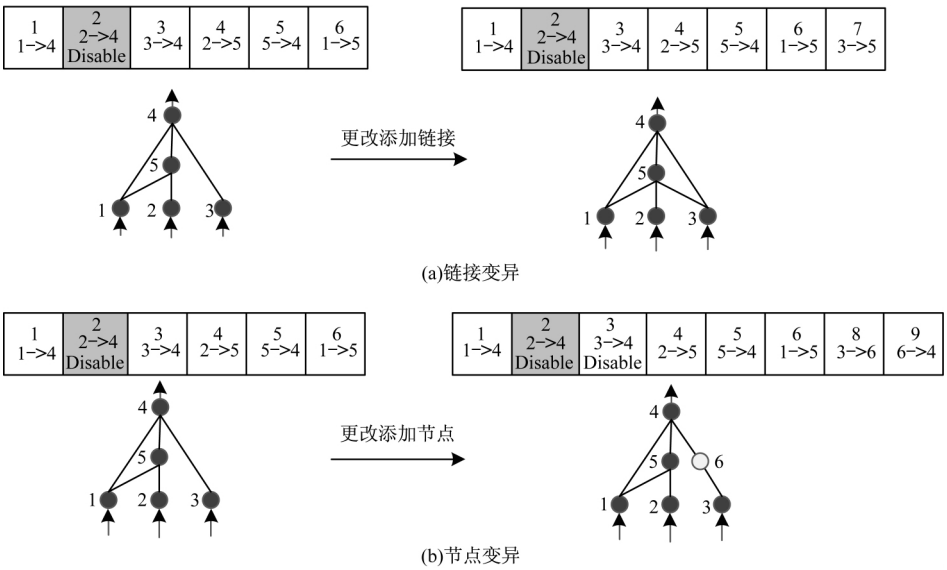

图 4　NEAT 的 2 种神经网络变异方式

Fig. 4　Two neural network mutation modes of NEAT

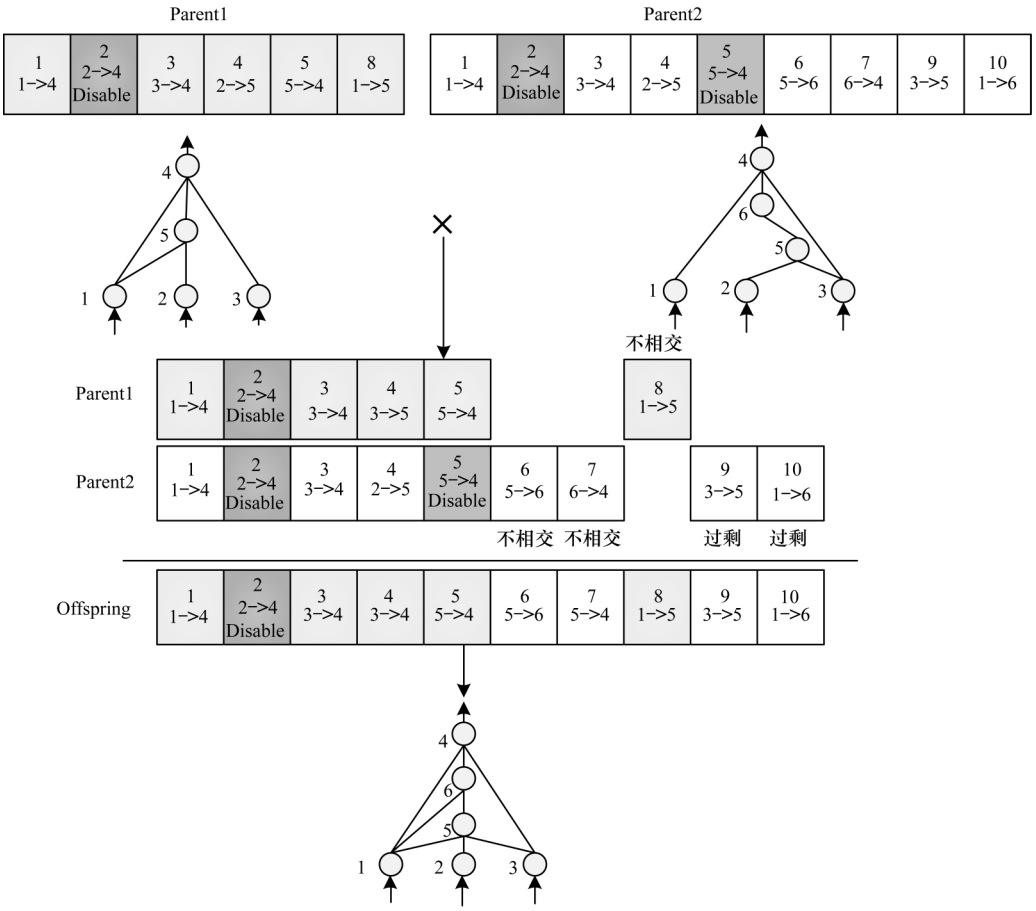

图 5 NEAT 的神经网络交叉方式

Fig. 5 Neural network cross mode of NEAT

NEAT 算法能够将复杂的问题分割成较小且可以被优化的问题(子任务)来进行解决。

3.2 基于 DNN 的神经架构搜索算法

随着深度神经网络的发展,网络的深度和复杂程度不断增加,网络的拓扑结构也变得十分复杂,从而产生了数百万的超参数。研究人员在优化深度神经网络时不可能考虑到所有的参数,只能凭借经验或者实验优化小部分的参数,而那些没有被选中的参数也有可能具有重要的地位。为解决该问题,基于 DNN 的神经架构搜索应运而生,其通过自动搜索 DNN 的结构来找到更为合适的神经网络架构。

为了更好地优化 DNN 的架构设计,文献[34]提出经典 CoDeepNEAT(Coevolution DeepNEAT)算法,该算法完全继承于 NEAT 算法。NEAT 算法主要是通过优化拓扑结构和权重的方式,而 CoDeepNEAT 算法则采用一种结合拓扑结构和超参数实现共同进化的方式。

3.2.1 DeepNEAT 算法

DeepNEAT 算法与 NEAT 算法具有相同的进化方式,首先创建一个具有最小复杂度的群体,然后通过变异的方式给网络结构不断添加节点或者边,实现进化的过程;在交叉操作时,使用 Innovation ID 的形式,可以准确地表示在拓扑结构多样化的种群中基因之间的匹配关系;最后基于相似性度量将群体划分为物种,物种再不断进化形成新的物种,从而保证结构的创新性。

DeepNEAT 算法与 NEAT 算法的主要区别在于,NEAT 算法的最小单元是基于神经元构成的,而 DeepNEAT 算法中的节点则是一个 DNN 层,每一个节点包含 2 种编码方式:实数编码(表示 Channel 数量、Kernel Size 等数值参数)和二进制编码(表示节点类型和类别参数)。2 种算法的另一个区别在于链接,在 NEAT 算法中链接表示权重,而在 DeepNEAT 算法中则代表链接关系与链接方向。

3.2.2 CoDeepNEAT 算法

DeepNEAT 算法在构造深度神经网络结构时,会导致网络结构复杂,而且结构没有规整性。因此,文献[34]采用一种共同进化的方式,使得网络结构简单化并增加其复用性。

在 CoDeepNEAT 算法中,涉及模块和蓝图 2 个新的概念。模块表示一个小的 DNN 结构,蓝图表示一个图,其中,每个节点包含指向特定模块的指针。

CoDeepNEAT 算法和 DeepNEAT 算法采用相同的进化方法,但是 CoDeepNEAT 算法同时进化 2 组

蓝图和模块，并且采用共同进化的方式，通过用相应的模块替换蓝图节点，将模块和蓝图组装成网络，从而得到一个大型的网络结构，称为集合网络。

在计算适应度时，根据蓝图中每个节点的指向，从相应的模块中随机选择一个个体加入到集合网络中，如果多个蓝图节点指向相同的模块，则在所有蓝图中使用相同的模块。该集合网络的适应度包括其中所有的蓝图和模块，计算包含该蓝图或模块的所有集合网络的平均适应度作为整个集合网络的适应度。CoDeepNEAT 算法的进化过程如图 6 所示，由 2 组蓝图和模块同时进化得到最右边的集合网络。

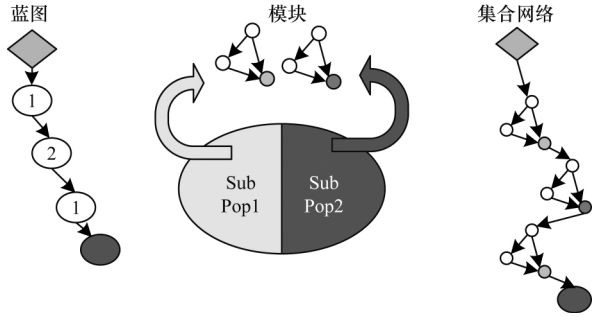

图 6　CoDeepNEAT 算法的进化过程

Fig. 6　Evolutionary process of CoDeepNEAT algorithm

3.3　基于 CNN 的神经架构搜索算法

在图像分类、目标检测等图像处理领域，为了提高深度学习的准确率，研究人员提出了 CNN，但是，人工设计的 CNN 结构存在局限性。因此，自动构造 CNN 的结构显得十分重要。

3.3.1　Genetic CNN 算法

Genetic CNN 算法于 2017 年被提出[53]，其是一种通过自动学习来获得最优 CNN 结构的方法，该算法将遗传算法用于 CNN 架构搜索中。在自动搜索网络结构的过程中，需要设置一些约束条件使搜索过程不会无限发展，其中一个约束条件就是 CNN 以池化层为界划分为不同的层，每个层中包含一系列预定义的构建块(由卷积层和池化层组成)，然后利用一种新的编码方式将网络结构通过固定长度的二进制编码进行表示，最后利用遗传算法在一个大的搜索空间内实现优化，从而提高搜索空间的遍历效率。Genetic CNN 算法的编码方式与进化方式具体如下:

1) Genetic CNN 的编码方式

对于 Genetic CNN 算法而言，编码方式是其重要的组成部分，该算法主要使用二进制编码方式，如图 7 所示。

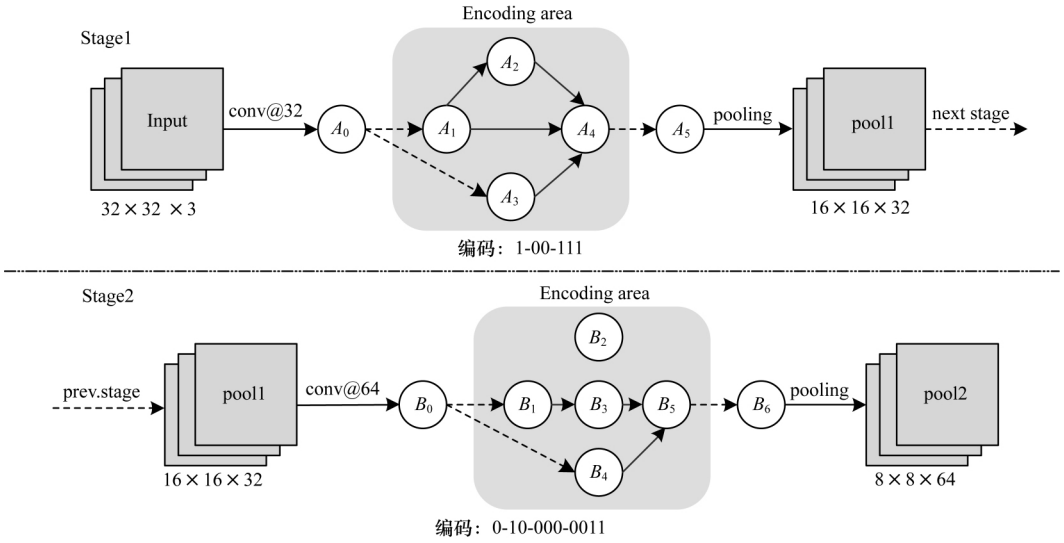

图 7　Genetic CNN 的编码方式

Fig. 7　Coding method of Genetic CNN

(1) 默认节点。图 7 中包含 2 个层(Stage)，每个 Stage 包含 2 个默认节点，一个是默认输入节点，另一个是默认输出节点。输入节点的数据来自前一个 Stage 的数据，然后执行卷积操作; 输出节点的数据为经过前面卷积操作之后的结果，然后进行求和再执行卷积操作，最后将结果输出到池化层。在 Stage1 中，默认输入节点为 A_0，默认输出节点为 A_5; 在 Stage2 中，默认输入节点为 B_0，默认输出节点为 B_6。

(2) 普通节点。除默认节点外的节点称为普通节点，即图 7 中编码区域中的节点，它们的节点序号唯一且有序。同一 Stage 中的所有卷积操作具有相同的卷积核和通道数量。

(3) 每一个节点均代表一个卷积操作，编码只针对普通节点进行。

(4) 孤立节点。孤立节点是为了保证具有更多节点的 Stage 可以模拟具有更少节点的 Stage 表示的所有结构,如图 7 中 Stage2 的 B_2 就称为孤立节点。

在图 7 中,普通节点的编码方式为:

(1) Stage 的个数为 S,Stage 中的节点个数为 K,则 $S=2$,$K_1=4$,$K_2=5$。通过简单的排列组合计算二进制点数,Stage1 所需要的二进制点数为 6,Stage2 所需要的二进制点数为 10。

(2) 对节点之间的链接逻辑进行编码,有链接表示为 1,无链接表示为 0。

Stage1 中的编码顺序为:1 和 2;1 和 3、2 和 3;1 和 4、2 和 4、3 和 4,即 1-00-111。

Stage2 中的编码顺序为:1 和 2;1 和 3、2 和 3;1 和 4、2 和 4、3 和 4;1 和 5、2 和 5、3 和 5、4 和 5,即 0-10-000-0011。

2) Genetic CNN 的进化方式

Genetic CNN 算法使用遗传算法进行 CNN 结构的搜索,主要分为初始化、选择、变异、交叉以及适应度评估等步骤。

(1) 初始化。初始化操作主要用于网络中个体的初始化,即创建初始种群。

(2) 选择。选择操作在每一代开始时执行,采用随机选择的方式,每个个体的适应度是其被选中的概率,适应度值越高,被选中的概率越大。

(3) 变异。每一个个体根据一定的概率改变自身结构,变异操作能够避免架构陷入局部最优解。

(4) 交叉。交叉操作能够使得相邻的个体以一定的概率互换 Stage 结构。

(5) 适应度评估。在个体的适应度评估中,主要以个体的适应度函数值为标准,减少由于随机性造成的不稳定性,假如一个个体之前被评估过,则将其适应度函数值的平均值作为该个体当前的适应度函数值。

Genetic CNN 算法因其约束条件而在很大程度上减少了网络结构的搜索范围,但是这些约束条件也导致很多的局限性。例如,每一层的卷积核大小和通道数量相同,这就导致了 CNN 的多样性。

3.3.2 CGP-CNN 算法

CGP-CNN 算法[36-37]于 2017 年被提出,该算法的思想与 Genetic CNN 算法类似,都是通过遗传算法进行架构的迭代选择与升级,两者的主要区别在于编码方式的不同。CGP-CNN 算法主要基于有向无环图,其基本组成单元为 Datanode 和 Graphnode,即在 CGP-CNN 算法中,将多个简单的神经元构成的有向无环图作为一个进化单元进行优化。CGP-CNN 算法的节点类型、编码方式及进化过程具体如下:

1) CGP-CNN 算法的节点类型

CGP-CNN 算法中包括 6 种节点类型,分别是 ConvBlock、ResBlock、Max Pooling、Average Pooling、Concatenation 和 Summation。这些节点操作由行、列和通道的大小定义成三维(3D)张量。

(1) ConvBlock 由标准卷积处理组成,步长为 1,然后是批量归一化和 ReLU 激活函数,其中,卷积块不进行降维,因此,采用零值填充来保持前后特征图维度不变,并且具有不同大小的 Kernel Size 和不同的 Channel 数。

(2) ResBlock 由卷积处理、批量归一化、ReLU 函数和张量求和组成。ResBlock 架构如图 8 所示,输入的行和列大小与卷积后的 ConvBlock 保持一致。在 ResBlock 中,$M\times N\times C$ 的输入特征映射被变换为 $M\times N\times C'$ 的输出映射,并且具有不同大小的 Kernel Size 和不同的输出 Channel 数。

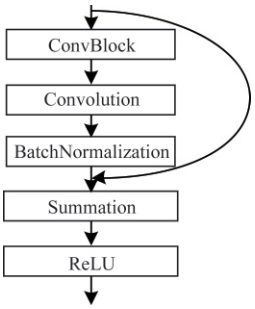

图 8 ResBlock 的结构

Fig. 8 Structure of ResBlock

(3) Pooling 池化层执行 Max Pooling 和 Average Pooling 的操作,使用 2×2 的 Kernel,步长 Stride = 2。

(4) Concatenation 将不同特征图进行合并,如果特征图大小不同,则对更大的特征图进行 Max Pooling 完成下采样,从而实现特征图大小一致。

(5) Summation 主要用于 skip-connection 的特征图元素两两相加,当特征图大小不同时,对更大的特征图进行最大池化完成下采样,从而实现特征图大小一致。

2) CGP-CNN 的编码方式

基于进化算法的神经网络架构在应用时,通常有直接编码和间接编码 2 种编码形式。直接编码主要将神经元的数量和链接情况作为基因类型,而间接编码主要表达网络架构的生成规则。CGP-CNN 算法采用笛卡尔基因编程这种直接编码的方式,从而表示 CNN 的结构和连通性。CGP-CNN 算法具有灵活性,可以表示可变长度的网络结构和跳远链接。

图 9 所示为一种 CGP-CNN 的编码方式,其中左边是 Genotype,右边是 CNN 架构图,节点 5 是一个未激活的节点。

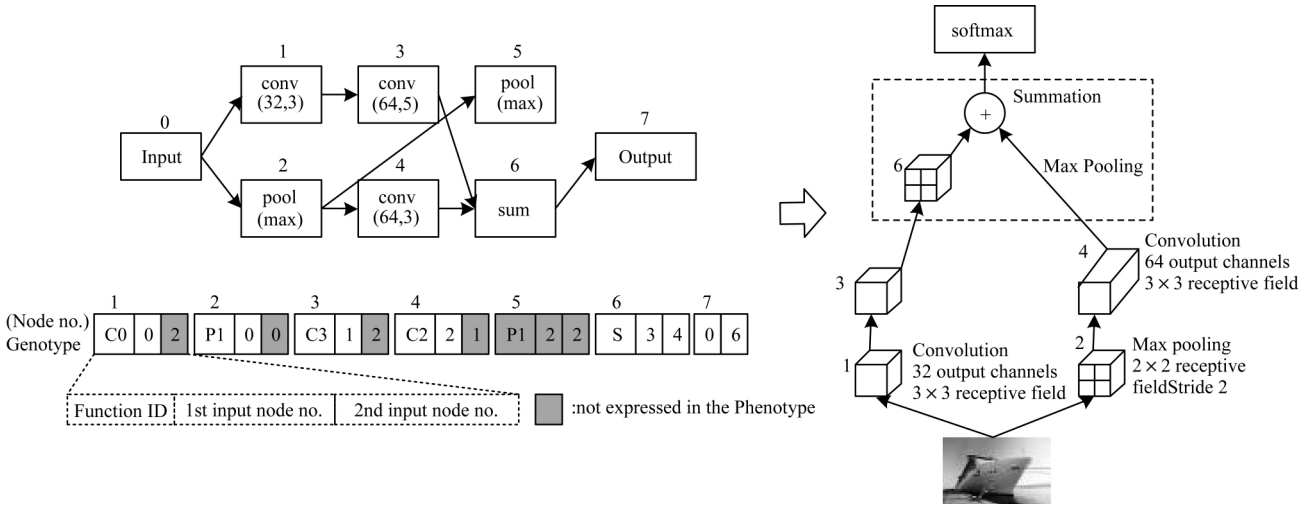

图 9　CGP-CNN 的编码方式

Fig. 9　Coding method of CGP-CNN

3) CGP-CNN 算法的进化过程

CGP-CNN 算法的变异方式有 2 种: 第 1 种是强制突变，为了有效地使用计算资源，需要在每一代并行地评估一些候选解决方案，因此，在使用变异操作时，要保证至少有一个活动节点改变之后出现后选解，此时称为强制突变; 第 2 种是中性突变，为了维持对 CGP 进化的有效性，如果后代的适应度值没有得到改善，可以通过使用中性突变的方式重新对父代执行变异操作，在中性突变过程中，只改变非活动节点的基因而不改变 Phenotype。

CGP-CNN 算法验证了在自动构建 CNN 架构时，可以使用笛卡尔基因编程的方式，并且基本单元可以采用如 ConvBlock、ResBlock 这样的构建块。但是，为了得到效果更好的网络结构，在进化过程中，CGP-CNN 算法需要消耗很多资源，因此，后续将重点研究降低计算资源消耗的方法，或者在进化过程中通过减少多余的 CNN 结构来实现进化算法的优化。

3.4　基于进化算法的 NAS 改进

CoDeepNEAT 算法、Genetic CNN 算法和 CGP-CNN 算法在基于进化算法进行神经网络架构搜索时，都很依赖先验知识并基于已有的模块进行叠加组合。此外，这些算法要求每一个个体均从初始时刻开始训练，这样就需要很多的计算资源才能满足架构搜索任务的需求。为解决上述问题，研究人员提出 Large-Scale 算法[38]、Better Topologies 算法[39]。为了提高搜索效率并降低搜索空间，文献[40]提出了 Hierarchical Representation 算法。

Large-Scale 算法和 Better Topologies 算法的设计思想类似，只是在具体的参数设置上存在区别，两者都采用了非常简单的神经网络设定方法，使得算法进化出网络结构。初始化阶段从一个包含 3 个节点(输入节点、中间节点和输出节点) 的简单架构开始。进化过程则是对 internal node 的连接方式进行改变，其中，在 Large-Scale 算法中，对 internal node 采用卷积、池化、批量归一化等操作中的某一个或者多个相组合; 在 Better Topologies 算法中，对 internal node 执行卷积和池化操作。这 2 种算法与 NEAT 算法不同的是只采用了变异操作，而没有使用交叉操作。Better Topologies 算法有 5 个变异算子，Large-Scale 算法有 11 个变异算子。

Hierarchical Representation 算法是一种结合模型的结构分层表示和进化策略的高效架构搜索方法。为了降低网络架构搜索的复杂性，提高搜索效率，Hierarchical Representation 算法对搜索空间进行约束，通过增加一个分层的网络结构来约束搜索空间。Hierarchical Representation 算法的基本单元为基元(motif) ，较低层次的 motif 由卷积、池化等操作构成，然后通过将这些低层次 motif 多次叠加的方式最终形成神经网络结构，其中，堆叠的方式由进化策略或者随机搜索决定。这种通过堆叠的方式构建网络结构，类似于 VGGNet、ResNet 和 Inception 结构，使用模块化进行设计。

3.5　算法对比与分析

前文提到的 7 种算法对比如表 1 所示。NEAT 算法的基本单元为神经元(Neuron) ，采用链表的方式进行编码，有增加连接、增加节点和修改权重 3 种变异方式，在进行交叉操作时，根据 Innovation ID 决定采用交叉操作的 2 个基因序列。CoDeepNEAT 算法的基本单元为 DNN Layer，采用实数编码和二进

制编码 2 种编码方式，并且有改变连接和改变节点 2 种变异方式，在交叉操作阶段，根据历史标记确定 2 个基因排列方式。Genetic CNN 算法的基本单元是通过 Block 方式构造的 CNN 结构，采用二进制编码，根据每个个体的概率进行变异和交叉。CGP-CNN 算法的基本单元与 Genetic CNN 算法相同，都是通过 Block 方式构造的 CNN 结构，采用的变异方式为强制变异和中性变异，没有交叉操作。Large-Scale 算法和 Better Topologies 算法的基本单元都为 Node，编码方式分别采用 DNA Graph 和整数编码，分别为 11 种变异方式和 5 种变异方式，都没有交叉操作。Hierarchical Representation 算法的基本单元是由一个三级结构表示的有向无环图，这个有向无环图包括 3 个节点(特征图)，每2 个节点的边有 6 种基本操作可供选择，在进化过程中包括增加连接、修改连接和删除连接 3 种变异方式，没有交叉操作。

表 1 7 种基于进化算法的神经架构搜索算法对比

Table 1 Comparison of seven neural architecture search algorithms based on evolutionary algorithm

算法	基本单元	编码方式	变异操作	交叉操作
NEAT 算法	Neuron	数组或者链表	3 种变异方式	根据 Innovation ID 进行交叉操作
CoDeepNEAT 算法	DNN Layer	实数编码/二进制编码	2 种变异方式	根据历史标记进行交叉操作
Genetic CNN 算法	CNN(Block)	二进制编码	1 种变异方式	根据概率交叉 Stage
CGP-CNN 算法	CNN(Block)	直接编码	2 种变异方式	无
Large-Scale 算法	Node	DNA Graph	11 种变异方式	无
Better Topologies 算法	Node	整数编码	5 种变异方式	无
Hierarchical Representation 算法	三级结构	有向无环图	3 种变异方式	无

CoDeepNEAT 算法使用 CIFAR-10 数据集进行实验，初始化阶段生成了 25 个蓝图和 45 个模块。在训练集上每个网络训练 8 个 epoch，然后进行评估，经过 72 代的进化之后结束训练。将得到的最优网络在所有训练图片上进行 300 个 epoch 的训练，最终得到错误率为 7.3%。

Genetic CNN 算法分别在 MNIST 数据集和 CIFAR-10 数据集上进行实验，初始种群数量为 20，执行 50 代进化操作。在 MNIST 数据集上，经过 50 代后，最佳个体的识别错误率由 0.41% 下降到 0.34%。实验在 Titan-X GPU 上进行，每个个体的训练时间平均为 2.5 min，整个进化过程大约需要 2 GPU-days。在 CIFAR-10 数据集上，经过 50 代后，最佳个体的识别正确率从 75.96% 上升到 77.06%，每个个体的训练时间平均为 0.4 h，整个进化过程大约需要 17 GPU-days。

CGP-CNN 算法使用 CIFAR-10 数据集进行实验，搜索 ConvBlock 和 ResBlock 2 种类型结构，其中，ConvSet 的进化代数为 500 代，ResSet 执行 300 代，实验在 2 个 NVIDIA GeForce GTX 1080 GPU 上进行。CGP-CNN 算法使用 ConvSet 模型时的错误率为 6.75%，使用 ResSet 模型时的错误率为 5.98%。在 ResSet 上完成 CNN 架构的优化大约需要 14 天。

Large-Scale 算法分别在 CIFAR-10 数据集和 CIFAR-100 数据集上进行实验，并且都进行 5 组实验，达到的最佳准确率分别为 94.6% 和 77%，参数数量分别为 5.4 M 和 40.4 M。在 CIFAR-10 数据集上的计算成本为 4×10^{20} FLOPS，在 CIFAR-100 数据集上的计算成本为 2×10^{20} FLOPS。

Better Topologies 算法分别在 CIFAR-10 数据集、CIFAR-100 数据集和 SVHN 数据集上进行实验。在 CIFAR-10 和 CIFAR-100 上使用了标准图像增强(翻转和裁剪)进行数据预处理，最大迭代次数为 400 epoch，在 SVHN 数据集上不使用数据增强，最大迭代次数为 180 epoch。当网络深度为 91 时，3 个数据集上的错误率分别为 5.19%、24.6% 和 1.85%。

Hierarchical Representation 算法使用 CIFAR-10 数据集和 ImageNet 数据集进行实验，得到的 Top-1 错误率分别为 3.6% 和 20.3%。

通过对这 7 种算法的实验设置和数据进行对比可知，随着神经架构搜索算法的发展，图像分类的准确率不断提高。为了在更大的数据集上进行实验，同时降低计算资源消耗并保证准确率，通过神经架构搜索算法设计的网络结构也会越来越复杂。虽然目前神经架构搜索算法能够设计出与人工神经网络性能相当甚至更好的网络结构，但是在保证准确率的情况下，如何降低计算资源消耗仍然是一个具有挑战性的任务。

4 算法存在的问题及其解决方案

目前，研究人员针对神经架构搜索主要探究其搜索空间和搜索策略 2 个方面。搜索空间决定了哪

些结构可以增加到最后的网络结构中，其通过预先设定好一些适合当前任务的结构，以有效减小搜索空间并简化搜索过程。但是，预先定义好的搜索空间必然会引入人为的偏好，因此，该方式限制了网络搜索的程度。在后续的研究中，可以增加搜索空间的范围，使得神经架构搜索可以搜索到更丰富多样的网络结构。除此之外，减少人为的参与以及约束条件，使得神经架构搜索的搜索过程更加自动化，也是一个重要的研究方向。

搜索策略也是搜索算法，其决定了用何种规则或者方法来利用搜索空间，使得算法可以快速、准确地找到最优的网络结构和参数配置。进化算法是其中一种搜索策略，虽然进化算法的种类很多，在针对不同问题时可以选用不同类型的进化算法，但是进化算法的进化策略却大同小异，主要步骤通常是初始化（采用不同的方式对基因进行编码）、变异、交叉、评估，直到循环至预先设定好的最优标准时停止进化并输出最终的结果。因为进化算法是一个迭代的过程，每迭代一次都会消耗很长的时间，所以提高搜索效率并减少时间和资源的消耗是该过程中需要面临的一个重要问题。为了提高进化算法的搜索效率，可以将进化算法和其他搜索策略（如强化学习）相结合，组合时学习强化学习中的奖励机制，通过与环境的交互获得奖励，然后通过获得的奖励大小学习行为，使得算法能够获得更大的奖励，将强化学习中与环境交互的形式运用在进化过程中，从而提高搜索效率。因此，在后续的研究中，将进化算法和其他多种搜索策略相结合也是一个重要的研究热点。

除了上述 2 种问题及其解决方案之外，算法未来的发展方向也是一个很重要的研究内容。一方面，本文根据复杂程度将算法分为基于神经元的 NAS、基于 DNN 的 NAS 和基于 CNN 的 NAS，这种分类方式是根据神经网络的发展过程而进行的分析和总结。因此，在后续的研究过程中，也可以根据神经网络的后续发展来扩展 NAS 的多样性，如针对 RNN 的神经架构搜索或者基于更复杂、更深层的神经网络的 NAS 等；另一方面，目前基于自动化深度学习的有关图像方面的问题大多是关于图像分类的，因此，如何将自动化深度学习扩展到其他图像处理方面，如图像补全、图像修复等，甚至语音处理以及自然语言处理等领域，也是今后一个非常重要的研究热点。

5　结束语

本文阐述进化算法在神经网络架构搜索中的应用，根据构成神经网络最小单元的不同对神经网络架构搜索算法进行简单的分类，并介绍 7 种具有代表性的基于进化算法的神经架构搜索算法的特点和应用现状。基于无梯度进化的神经架构搜索算法具有广阔的发展前景，针对特定数据集设计的网络结构性能可以达到与人工设计的神经网络相同的水平。但是，搜索网络结构需要耗费大量的计算资源，因此，如何在大数据集上降低基于进化算法的神经架构搜索算法的计算资源消耗，将是下一步的主要研究方向。

参考文献

[1] WANG Jianzong，QU Xiaoyang. Dive into AutoML and AutoDL: building automated platforms for machine learning and deep learning [M]. Beijing: China Machine Press，2019. (in Chinese)
王健宗，瞿晓阳. 深入理解 AutoML 和 AutoDL：构建自动化机器学习与深度学习平台 [M]. 北京：机械工业出版社，2019.

[2] QUANMING Y，MENGSHUO W，HUGO J E，et al. Taking human out of learning applications: a survey on automated machine learning [EB/OL]. [2020-01-15]. https://arxiv.org/abs/1810.13306v1.

[3] ZHOU Zhihua. Machine learning [M]. Beijing: Tsinghua University Press，2016. (in Chinese)
周志华. 机器学习 [M]. 北京：清华大学出版社，2016.

[4] JENNINGS N R，WOOLDRIDGE M J. Foundations of machine learning [M]. Cambridge，USA: MIT Press，2012.

[5] ALPAYDIN E. Introduction to machine learning [M]. Cambridge，USA: MIT Press，2004.

[6] FORSYTH D A，PONCE J. Computer vision: a modern approach [M]. [S. l.]: Prentice Hall Professional Technical Reference，2002.

[7] GARCÍA S，LUENGO J，HERRERA F. Data reduction [M]// BENGIO Y，COURVILLE A. Intelligent systems reference library. Berlin，Germany: Springer，2014.

[8] HE Jun，GE Hong，WANG Yufeng. Overview of image segmentation algorithms [J]. Computer Engineering and Science，2009，31(12): 58-61. (in Chinese)
何俊，葛红，王玉峰. 图像分割算法研究综述 [J]. 计算机工程与科学，2009，31(12): 58-61.

[9] COLLOBERT R，WESTON J，BOTTOU L，et al. Natural language processing (almost) from scratch [J]. Journal of Machine Learning Research，2011，12: 2493-2537.

[10] LECUN Y，BENGIO Y，HINTON G. Deep learning [J]. Nature，2015，521(7553): 436-444.

[11] GOODFELLOW I，BENGIO Y，COURVILLE A. Deep learning [M]. Cambridge，USA: MIT press，2016.

[12] SEIDE F, LI G, CHEN X, et al. Feature engineering in context-dependent deep neural networks for conversational speech transcription [C]//Proceedings of 2011 IEEE Workshop on Automatic Speech Recognition & Understanding. Washington D. C., USA: IEEE Press, 2011: 24-29.

[13] CLAESKENS G, HJORT N L. Model selection and model averaging [M]. Cambridge, UK: Cambridge University Press, 2001.

[14] KHURANA U, NARGESIAN F, SAMULOWITZ H, et al. Automating feature engineering [J]. Transformation, 2016, 18(10): 1-10.

[15] SHANG D Y, SUN H, ZENG Q L. A reinforcement-algorithm framework for automatic model selection [J]. IOP Conference Series: Earth and Environmental Science, 2020, 440: 022060.

[16] OLSON R S, BARTLEY N, URBANOWICZ R J, et al. Evaluation of a tree-based pipeline optimization tool for automating data science [C]//Proceedings of 2016 Genetic and Evolutionary Computation Conference. New York, USA: ACM Press, 2016: 485-492.

[17] LECUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition [J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[18] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet classification with deep convolutional neural networks [J]. Communications of the ACM, 2017, 60(6): 84-90.

[19] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition [EB/OL]. [2020-01-25]. https://arxiv.org/pdf/1409.1556.pdf.

[20] SZEGEDY C, LIU W, JIA Y Q, et al. Going deeper with convolutions [C]//Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition. Washington D.C., USA: IEEE Press, 2015: 1-9.

[21] SZEGEDY C, IOFFE S, VANHOUCKE V, et al. Inception-v4, inception-ResNet and the impact of residual connections on learning [EB/OL]. [2020-01-25]. https://arxiv.org/abs/1602.07261.

[22] HE K M, ZHANG X Y, REN S Q, et al. Deep residual learning for image recognition [C]//Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Washington D. C., USA: IEEE Press, 2016: 770-778.

[23] ELSKEN T, METZEN J H, HUTTER F. Neural architecture search [EB/OL]. [2020-01-25]. https://arxiv.org/abs/1808.05377.

[24] BAKER B, GUPTA O, NAIK N, et al. Designing neural network architectures using reinforcement learning [EB/OL]. [2020-01-25]. https://arxiv.org/abs/1611.02167.

[25] ZOPH B, LE Q. Neural architecture search with reinforcement learning [EB/OL]. [2020-01-25]. https://arxiv.org/abs/1611.01578.

[26] SHIN R, PACKER C, SONG D. Differentiable neural network architecture search [EB/OL]. [2020-01-25]. https://openreview.net/pdf?id=BJ-MRKkwG.

[27] LIU H X, KAREN S, YIMING Y. Darts: differentiable architecture search [EB/OL]. [2020-01-25]. https://arxiv.org/abs/1806.09055.

[28] HOLLAND J H. Genetic algorithms and adaptation [M]. Berlin, Germany: Springer, 1984.

[29] BEYER H G, SCHWEFEL H. Evolution strategies-a comprehensive introduction [J]. Natural Computing, 2002, 1(1): 3-52.

[30] FOGEL D B, FOGEL L J, ATMAR J W. Meta-evolutionary programming [EB/OL]. [2020-01-25]. https://ieeexplore.ieee.org/document/186507.

[31] KOZA J R, POLI R. Genetic programming [M]. Berlin, Germany: Springer, 2005.

[32] GOMEZ F J, MIIKKULAINEN R. Active guidance for a finless rocket using neuro evolution [M]. Berlin, Germany: Springer, 2003.

[33] STANLEY K O, MIIKKULAINEN R. Evolving neural networks through augmenting topologies [J]. Evolutionary Computation, 2002, 10(2): 99-127.

[34] MIIKKULAINEN R, LIANG J, MEYERSON E, et al. Evolving deep neural networks [M]. [S.l.]: Elsevier, 2019.

[35] XIE L X, YUILLE A. Genetic CNN [C]//Proceedings of 2017 IEEE International Conference on Computer Vision. Washington D. C., USA: IEEE Press, 2017: 1379-1388.

[36] MILLER J F, HARDING S L. Cartesian genetic programming [C]//Proceedings of GECCO '09. New York, USA: ACM Press, 2009: 15-26.

[37] SUGANUMA M, SHIRAKAWA S, NAGAO T. A genetic programming approach to designing convolutional neural network architectures [C]//Proceedings of Genetic and Evolutionary Computation Conference. Berlin, Germany: Springer, 2017: 497-504.

[38] REAL E, MOORE S, SELLE A, et al. Large-scale evolution of image classifiers [C]//Proceedings of the 34th International Conference on Machine Learning. Washington D. C., USA: IEEE Press, 2017: 56-89.

[39] ZHANG H L, KIRANYAZ S, GABBOUJ M. Finding better topologies for deep convolutional neural networks by evolution [EB/OL]. [2020-01-25]. https://arxiv.org/abs/1809.03242.

[40] LIU H X, SIMONYAN K, VINYALS O, et al. Hierarchical representations for efficient architecture search [EB/OL]. [2020-01-25]. https://arxiv.org/abs/1711.00436.

编辑 吴云芳