计 算 机 学 报
CHINESE JOURNAL OF COMPUTERS

自动黑箱优化算法设计：进展与挑战

邱文杰 [1)] 郭泓树 [1)] 马泽原 [1)] 张军 [2)3)] 龚月姣 [1)]

[1)](华南理工大学计算机科学与工程学院 广州 510006)

[2)](南开大学人工智能学院 天津 300350)

[3)](汉阳大学电子与工程学院 安山 15588 韩国)

摘 要 工程与科学计算等领域中优化任务的规模和复杂性不断增长，传统依赖专家经验的人工优化算法设计在开发成本与泛化能力方面面临瓶颈。作为一种新兴的自动黑箱优化算法设计范式，元黑箱优化（MetaBBO）旨在从问题分布中学习可泛化的设计策略，逐渐成为智能优化与进化计算领域的研究热点。本文首先定义了元黑箱优化的基本概念与双层优化架构。随后，基于优化器空间与决策空间两类互补视角，系统化地梳理了五类核心元任务的研究进展：优化器空间中的算法“选择”、“配置”、“组装”与“生成”，以及决策空间中的“求解操作”。进一步，本文深入探讨了大语言模型在元黑箱优化中的应用前景与挑战，指出其在实现端到端一体化优化流程中的潜力与局限。最后，针对当前方法在问题理解、模型构建与训练机制等方面的不足，本文提出了构建多样化问题集、引入高效学习机制以及发展混合任务基座模型等多项研究展望， 旨在推动元黑箱优化朝向更高程度的通用性、自动化，灵活性与实用性发展，从而为现实复杂优化问题的高效求解提供先进算法支持。

关键词 黑箱优化；元黑箱优化；元学习；自动算法设计；进化计算

中图法分类号 TP 18

Automated Algorithm Design for Black-Box Optimization: Progress and Challenges

QIU Wen-Jie[1)] GUO Hong-Shu[1)] MA Ze-Yuan[1)] ZHANG Jun[2)3)] GONG Yue-Jiao[1)]

[1)](School of Computer Science and Engineering, South China University of Technology, GuangZhou 510006)

[2)] (College of Artificial Intelligence, Nankai University, Tianjin 300350)

[3)](School of Electrical Engineering, HanYang University, Ansan-si 15588 Republic of Korea)

Abstract As optimization problems in engineering and scientific computing continue to increase in scale and complexity, the traditional paradigm of manually crafting optimizers based on expert knowledge is increasingly constrained by high development costs and limited generalization. Meta-Black-Box Optimization (MetaBBO) is an emerging meta-learning-driven paradigm that seeks to automate black-box optimizer design by learning transferable strategies from a distribution of problem instances, and it has attracted growing attention in intelligent optimization and evolutionary computation. This survey first formalizes the core concepts of MetaBBO and its bi-level optimization framework. It then provides a systematic review of five key meta-tasks from two complementary

本课题得到国家自然科学基金面上项目 (62276100)、广东自然科学基金卓越青年团队项目 (2024B1515040010)、广州市基础与应用基础研究专题科技菁英“领航”项目 (2024A04J01361)、中央高校基本科研业务费专项资金 (2025ZYGXZR027)资助。邱文杰, 博士研究生, 主要研究领域为元黑箱优化、进化计算。 郭泓树, 博士研究生, 主要研究领域为进化计算、元黑箱优化、强化学习。马泽原, 博士研究生, 主要研究领域为进化计算、元黑箱优化、强化学习。张军, 博士, 教授, 主要研究领域为人工智能、计算智能、进化计算、群体智能。龚月姣（通讯作者），博士，教授，中国计算机学会 (CCF) 会员, 主要研究领域为元黑箱优化、进化计算。

viewpoints: the optimizer space, including algorithm selection, configuration, composition, and generation; and the decision space, focusing on solution manipulation. We further discuss opportunities and challenges in integrating large language models into MetaBBO, emphasizing their promise and current limitations for enabling end-to-end optimization workflows. Finally, in light of existing gaps in problem understanding, model construction, and training mechanisms, we outline future directions such as building more diverse problem suites, developing more efficient learning paradigms, and advancing hybrid foundation models for multi-task meta-optimization. These efforts aim to improve the generality, automation, flexibility, and practical impact of MetaBBO, thereby providing stronger algorithmic support for solving complex real-world optimization problems.

Keywords black-box optimization；meta-black-box optimization；meta-learning；automated algorithm design；evolutionary computation

1 引言

从自然界中物理系统倾向于演化至总能量最低的稳定状态，到人类社会中航空公司不断调整机组人员和飞机的飞行计划以最大程度降低运营成本，几十年来，优化问题一直是各领域研究者关注的中心问题之一。大量的优化技术推动着各领域在有限资源和复杂环境中寻找高效、经济和有利的方案，例如金融中的投资组合优化[1]、物流中的网络流优化[2]、医疗中的药物优化设计[3]、能源中的电网调度[4]、航天领域的发动机设计优化[5] 等。从问题的结构信息是否已知的角度进行分类，优化问题可以被分为“白箱优化”和“黑箱优化”两类。其中白箱优化中的“白”，指的是问题结构是已知的，即有显式数学表达式或梯度信息等，这可以利用基于梯度的算法 (如 Adam[6] 和 BFGS[7] 等) 进行有效优化。相反，黑箱优化中的问题结构是未知的，这导致无法利用问题结构信息来进行优化。黑箱优化中唯一的优化信息来源是输入的采样点和其对应输出的评估值组成的输入输出对, 这使得其优化更具挑战性。

包含进化算法和群体智能的进化计算，是一类无需依赖问题结构的优化算法，在过去几十年中被广泛应用于多种优化任务，其中黑箱优化是其典型应用之一。尽管传统进化计算在黑箱优化中已展现出良好效果，但其性能仍受限于“没有免费午餐”定理，即不存在在所有问题上均优于其他算法的通用方法。同时，进化计算对算法配置高度敏感，非最优或随意的参数设置可能显著削弱性能，甚至引发收敛失败或结果不稳定。这一方面缺乏通用最优算法，另一方面高度依赖参数配置的特性，使得传统方法往往需要依赖人工经验进行问题定制化设计。然而这样的人工设计完全依赖于专家经验，一方面会造成巨大的资源消耗（人力资源、时间资源、能源资源、经济资源等）[8]，另一方面也受限于专家个人经验。这些缺点，在缺少优化信息的黑箱优化问题中被进一步放大，专家难以从有限的信息中提取经验进行设计，往往需要大量的实验来丰富优化信息，导致更大的资源消耗且设计效果受限。

为提升算法性能，传统人工设计算法衍生出多种进化计算变体，包括离线超参数优化（offline hyper-parameter optimization）[9]、超启发式方法（hyper-heuristics）[10] 以及自适应机制（self-adaptive approaches）[11]。然而， 这些方法在实践中仍面临诸多挑战：（1）依赖专家知识：设计有效的自适应机制通常需要进化计算领域与目标问题领域的双重专家知识，增加了设计难度。（2） 设计效率低下：针对特定问题构建有效算法方案，往往需耗费数周甚至更长时间的反复调试与验证。（3） 泛化能力有限：大多数方法针对特定问题集设计自适应机制，导致其在跨任务迁移时效果受限。

然而，随着生产力持续提升，复杂优化需求激增。这些需求呈现两种形态：跨领域涌现异质化难题（如芯片设计、无人机群控制、汽车结构件优化

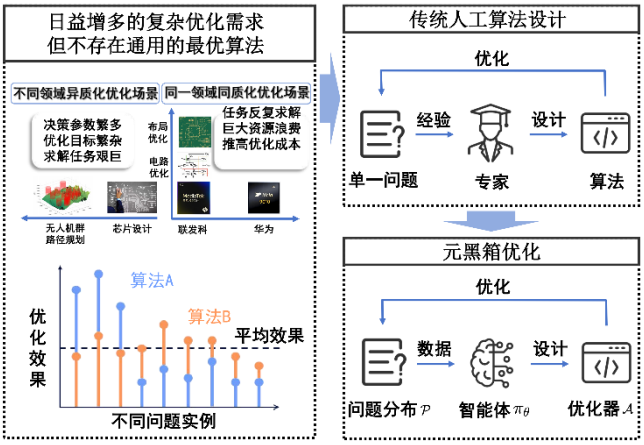

图1 元黑箱优化的研究背景与思路

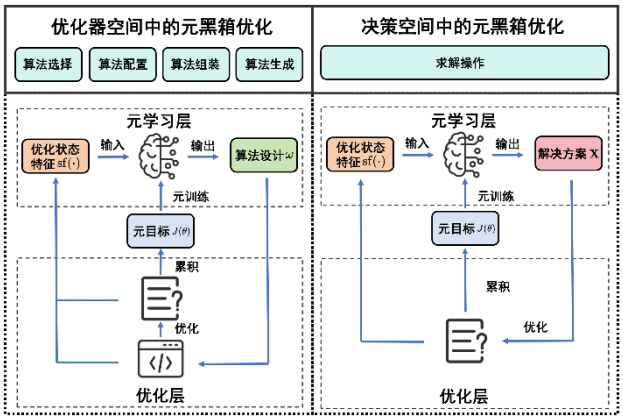

图2 优化器空间与决策空间中元黑箱优化架构

中的新型黑箱优化问题）；领域内存在同质化挑战（如芯片设计中反复面对的电路布图布局优化、模拟参数优化）。持续扩张的“同质化-异质化需求矩阵”，深刻暴露了传统方法在设计效率与泛化能力上的固有局限。

基于这一挑战，在 AI4Science 的浪潮下，一个自然的问题随之提出：能否实现自动化地设计具备更强泛化能力的黑箱优化算法[12-14]？自动设计的理念代表着从传统的人工设计范式向基于智能体的自动化设计范式转变，既有助于减少对专家知识的依赖， 也能够借助智能体强大的计算能力显著缩短算法开发周期。正如人类通过经验积累不断提升问题求解能力，元学习（Meta-Learning）使智能体能够在多个任务中提取共性经验，从而增强其在新任务中的适应与泛化能力[15]。元学习通常在一系列相关任务的分布上进行训练，被视为突破“没有免费午餐”定理限制的重要路径，也常被称为“学会学习（Learning to Learn）”。作为进化计算领域中新兴的研究方向， 元黑箱优化（Meta-Black-Box Optimization，MetaBBO）通过构建智能体并引入元学习机制，结合其在泛化与适应方面的能力，实现了在提升算法泛化性能的同时提高算法设计效率 (如在算法配置上，可由原有周级设计缩短至分钟级设计)[16]。图 1展示了元黑箱优化的研究背景与思路：通过训练元智能体在优化问题分布上学习一套通用的最大化优化器的整体性能的设计策略以替代传统人工设计。训练完成后，元智能体学习到的策略即可直接迁移至新问题，无需专家重新设计算法，从而在显著增强泛化能力的同时，降低了对专家知识的依赖，并通过自动化流程提升了算法设计效率。这一范式还常产生突破人类的设计，已在许多领域得到验证和应用。例如：EoH [17] 在经典数学难题圆堆砌问题上找到突破性的最优解；DQLGA [18] 在遥感数据离散化方案上不仅提高了搜索效率，还可以获得更优的分类准确率和断点数量；DeepMind 提出的 AlphaEvolve[19] 结合大语言模型和遗传算法来演化代码，在多个实际领域上取得了效果突破，如产生了更优的数据中心调度启发式方案使谷歌公司全球算力利用率提高 0.7 %）以及优化矩阵乘法内核使训练加速 1 % 等成果。

现有的元黑箱优化研究已经提出并讨论了许多有趣的想法。将现有研究依据元智能体的输出空间进行分类，可以分为优化器空间和决策空间两大类：（1） 优化器空间：元智能体为具体优化器的各个组件进行设计，或对抽象优化器进行实例化设计，因此输出的是优化器空间内的设计。（2） 决策空间: 元智能体本身作为优化器，通过训练更新其参数，因此输出的是决策空间内的解。两者最明显的区别在于，元智能体和优化器是否是统一为一体，图 2展示了两类空间中的元黑箱优化双层优化架构。再依据元智能体的算法设计任务（元任务）的类型对两类空间内的研究进行分类，优化器空间内的研究可以被分为四类：（1）算法选择: 元智能体从预定义的算法池中为优化器选择最适合当前问题的黑箱优化算法[20-24]。（2）算法配置：针对具体问题，元智能体为

优化器提供相应的超参数设置或算子组合[25-27]。（3）算法组装：元智能体从预定义的组件池中逐步选择算子和超参数来组合而成优化器进行优化[28-29]。（4）算法生成：元智能体直接生成完整的优化流程和对应的组成部分来构成一个全新的优化器[17,30]。而决策空间的研究与求解操作这一元任务密切相关[31]。在这一类元任务中，元智能体被训练为一个独立的黑箱优化算法，作为优化器直接在优化层中执行求解过程，最终输出解。

从元智能体所采用的学习范式来看，现有研究已探索了多种机制，包括强化学习、监督学习、神经进化以及上下文学习等。若从所面对的黑箱优化问题类型进行分类，元黑箱优化已被广泛应用于单目标优化[22,32-33]、多目标优化[23,34-35]、多模态优化[36-37]、多任务优化[38-39]、大规模优化[40-42]等多个方向。然而，从元任务，到元智能体的学习范式，再到所适应的优化问题类型，三者之间构成了高度多样化且交织复杂的组合空间。这种复杂性不仅加大了对元黑箱优化体系的理解难度，也对新进入该领域的研究者在系统学习、方法选择与实际开发方面提出了更高的挑战。与此同时，大语言模型的迅速发展为元黑箱优化注入了更具创造力与通用性的算法设计思路[43-46]，大量相关研究相继涌现，使得对其应用路径、方法特征与潜在优势进行系统梳理与评估变得尤为重要。这不仅有助于厘清大语言模型与传统元智能体方法之间的关系，也为推动元黑箱优化的进一步发展提供了清晰的理论与实践指引。

本文从元任务出发对元黑箱优化领域进行了全面的梳理。首先是在第2节给出了元黑箱优化的公式化定义，明确了其双层优化架构。第 3节和第4节分别从优化器空间和决策空间两个不同的角度出发，阐述了在两类空间中的五种不同元任务。第 5节对各类元任务进行了总结，并对其中使用的学习范式进行了对比。第 6节重点梳理了大语言模型在元黑箱优化中的应用。最后，第 7节指出了现有研究面临的挑战，并给出了对元黑箱优化未来研究方向的思考。

2 元黑箱优化的定义

在本节中，我们将首先给出元黑箱优化的通用定义和其双层优化架构，并基于定义解释该架构中各核心组件的功能与协同机制。此后，从通用定义出发，我们根据各类元任务的目的给出了其元黑箱优化的具体定义。

2.1 元黑箱优化的通用定义

元黑箱优化[16]是从“学会学习”中汲取了核心思想，并将其范式迁移至黑箱优化上。元黑箱优化相较于传统黑箱优化的显著的区别在于：其目标不再局限于单一优化问题，而是扩展至问题分布$\mathcal{P}$；设计主体由人类专家转变为智能体π_θ；设计依据也从人工经验转向由数据 s_i^t 所驱动的学习过程。正是这些关键转变，为元黑箱优化带来了更强的泛化能力与更高的设计效率。

由元学习层与优化层组成的双层优化架构，构成了元黑箱优化中通用的“学习－优化”闭环流程，已成为普遍采用的基本框架。该优化架构如图 2 所示，具体结构如下所述。元学习层由两大核心组件构成：元智能体 π_θ 和算法设计空间 Ω。其中，元智能体 π_θ是元黑箱优化中进行自动算法设计的智能体，具体可为一个神经网络或者 Q 表[47]等多种形式，由可学习的参数 θ 进行参数化。算法设计空间 Ω 与元任务密切相关，具有高度多样性，如元任务是算法配置任务，那么其算法设计空间便是相应的算法配置空间。优化层由四大核心组件构成：优化器 $\mathcal{A}$，优化问题分布 $\mathcal{P}$，状态特征提取方程 sf(·)和性能评估函数 perf(·)。

其中，优化器 $\mathcal{A}$同样与元任务密切相关，是一个灵活的抽象概念。它可以是任何一个具体的进化计算算法，也可以是一个包含多个算法的算法池，或者一个用于生成算法的框架，如元任务是算法选择任务，那么其优化器便是相应的算法池。优化问题分布$\mathcal{P}$，则代表了目标优化问题所处的问题分布。状态特征提取方程sf(·)是用以从每个优化步 t 中优化器$\mathcal{A}$ 与问题实例 $f_i \in \mathcal{P}$ 的交互结果来提取优化状态特征 s_i^t。性能评估函数 perf(·)则用于量化不同算

法设计方案的效果。元智能体的算法设计任务被称为元任务 $\mathcal{T}$，即元智能体对优化问题分布 $\mathcal{P}$ 上的问题实例在算法设计空间 Ω 内对优化器 $\mathcal{A}$ 进行算法设计，这可表示为三元组 $\mathcal{T}:=\{\mathcal{P},\mathcal{A},\Omega\}$。元智能体 π_θ 的训练目标被称为元目标 $J(\theta)$，是最大化其在优化问题分布 $\mathcal{P}$ 上的优化效果的期望：

$$J(\theta)=\mathbb{E}_{f\in\mathcal{P}}[\mathrm{R}(\mathcal{A},\pi_\theta,f)]$$
$$\approx\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{perf}\left(\mathcal{A},\omega_i^t,f_i\right)\quad\#(1)$$
$$\omega_i^t=\pi_\theta(s_i^t),\quad s_i^t=\mathrm{sf}(\mathcal{A},f_i,t)$$

其中，$\mathrm{R}(\cdot)$ 表示在某一问题实例 f 上优化器$\mathcal{A}$所获得的累积优化效果。元黑箱优化通常从 $\mathcal{P}$ 中采样有限数量的任务，构成训练集 $\{f_1,f_2,\dots,f_N\}$，并在每个问题上执行 T 步优化操作，用得到的平均优化效果作为对元目标的近似。训练过程中，首先从训练集中选取一个问题实例 f_i。在每个优化步骤 t，状态特征提取函数 $\mathrm{sf}(\cdot)$提取当前优化状态特征 s_i^t，作为元智能体 π_θ 的输入，并输出对应的算法设计 ω_i^t。该设计被应用至优化器$\mathcal{A}$，用于优化问题f_i，其优化效果由函数 $\mathrm{perf}(\cdot)$评估, 并得到相应的 $\mathrm{R}(\cdot)$用以元训练π_θ。随后， 新的优化状态 s_i^{t+1}被再次提取并送入元学习层，完成一次“学习-优化”闭环的迭代(即在一个问题实例 f_i 上学习一个优化步t)。

2.2 元黑箱优化中各类元任务的具体定义

由于优化器$\mathcal{A}$和算法设计空间Ω受元任务$\mathcal{T}$的影响，具有高度灵活性和多样性。我们从公式 1出发，对第 1节中介绍五类元任务进行公式化的定义。

（1）算法选择的通用范式：在算法选择中，元黑箱优化的目标是元智能体通过学习可以在每个优化步骤 t 能够从算法池 $\mathcal{A}$ 中选择出对于当前问题 f_i 效果最优的候选算法。此时，优化器的定义为一个包含 K 个候选黑箱优化算法组成的算法池 $\mathcal{A}=\{\mathcal{A}_1,\dots,\mathcal{A}_K\}$。对应地，算法设计空间的定义为索引集合 $\Omega=\{1,2,\dots,K\}^t$，表示在每个优化步 t 可从算法池中选择的算法索引。此时的元目标可被表示为：

$$J(\theta)\approx\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{perf}\left(\mathcal{A}[\omega_i^t],f_i\right)\#(2)$$

具体而言，其训练流程为：首先从训练集 $\{f_1,\dots,f_N\}$ 中采样一个问题实例 f_i。在每一个优化步t，优化层通过状态特征提取函数 sf 获取当前状态特征 s_i^t，并将其作为输入传递给元智能体 π_θ，后者输出一个算法索引 $\omega_i^t\in\Omega$。被选择的算法 $\mathcal{A}[\omega_i^t]$ 随后用于优化问题实例 f_i，其在该问题上的优化效果通过性能评估函数 perf 进行度量，公式为：$\mathrm{perf}(\mathcal{A},\omega_i^t,f_i)=\mathrm{perf}(\mathcal{A}[\omega_i^t],f_i)$。完成训练后，元智能体 π_θ 具备在未见问题上，根据状态特征自动从算法池中为优化器选择性能较优算法的能力，从而实现对新任务的泛化。

（2）算法配置的通用范式：在算法配置中，元黑箱优化的目标是元智能体π_θ学习到在每个优化步 t 下，为优化器$\mathcal{A}$动态配置在当前状态下对f_i最有效的方案。此时，优化器$\mathcal{A}$的定义是一个待配置的黑箱优化算法。对应地，算法设计空间的定义为 $\mathcal{A}$ 的配置空间，包含所有可能的超参数组合或算子配置方案。此时的元目标可被表示为：

$$J(\theta)\approx\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{perf}\left(\mathcal{A}.\mathrm{set}(\omega_i^t),f_i\right)\#(3)$$

具体而言，其训练流程为：给定一个问题实例 f_i，在每个优化步t，优化层通过状态特征提取函数 $\mathrm{sf}(\cdot)$ 提取当前状态 s_i^t。该状态作为输入传递至元智能体 π_θ，后者输出当前步的算法配置 $\omega_i^t=\pi_\theta(s_i^t)$。随后，优化器根据该配置进行更新，即执行 $\mathcal{A}.\mathrm{set}(\omega_i^t)$，并在当前优化步使用重新配置的 $\mathcal{A}$ 来优化问题实例 f_i。完成训练后，元智能体能够以动态的形式来进行高效的参数配置。相比传统依赖人工设计的自适应配置方法，该方式不仅显著提升了算法配置的效率，同时受益于双层优化架构，元智能体得以在大量问题实例上进行训练，从而提炼出可迁移于新问题的配置策略，显著增强了算法的泛化能力。

（3）算法组装的通用范式：在算法组装中，元黑箱优化的目标是元智能体学习到在每个优化步t下，通过从预定义的组件池中为 f_i 选择算子和超参数来组合而成优化器进行优化。此时，优化器$\mathcal{A}^t$即为元智能体输出的组件序列组成的黑箱优化算法

$\mathcal{A}^t=[\omega_i^1,\dots,\omega_i^m]^t=\pi_\theta(s_i^t)$，$m$为其输出的序列长度。对应地，算法设计空间的定义为预定义的组件池。此时的元目标可被表示为：

$$J(\theta)\approx\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{perf}\left(\mathcal{A}^t=[\omega_i^1,\dots,\omega_i^m]^t,f_i\right)\#(4)$$

具体而言，其训练流程为：针对某个问题实例 f_i，在每个优化步t，首先由状态特征提取函数 $\mathrm{sf}(\cdot)$ 提取当前状态 s_i^t。该状态作为输入传入元智能体 π_θ，后者输出长度为m的组件序列以组成新的黑箱优化算法作为优化器 $\mathcal{A}$ 进行优化。与算法选择和算法配置的显著区别在于，在算法组装中，元智能体输出之前并不存在一个实例化的黑箱优化算法。相反，完成训练后，元智能体通过组合各个组件创造出一个新的黑箱优化算法，作为优化器进行优化。该方式一定程度上突破了传统算法框架的限制，提升了算法设计的创新性。

（4）算法生成的通用范式：在算法生成中，元黑箱优化的目标是元智能体学习到在每个优化步 t 下，为f_i直接生成优化器$\mathcal{A}^t$，并能在优化层中实现高性能优化。此时，优化器$\mathcal{A}^t$即为元智能体生成的黑箱优化算法$\mathcal{A}^t=\omega_i^t=\pi_\theta(s_i^t)$。对应地，算法设计空间的定义为$\mathcal{A}$的表达空间，其可能是数学表达式空间，也可能是编程语言空间等。此时的元目标可被表示为：

$$J(\theta)\approx\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{perf}\left(\mathcal{A}^t=\omega_i^t,f_i\right)\#(5)$$

具体而言，其训练流程为：针对某个问题实例 f_i，在每个优化步t，首先由状态特征提取函数 $\mathrm{sf}(\cdot)$ 提取当前状态 s_i^t，该状态特征可包括：地形分析指标、函数的符号表示，或自然语言形式的问题描述。元智能体π_θ 接收 s_i^t，并输出一个具体的算法设计 ω_i^t，该设计可被解释为：一个数学表达式，即优化规则，或一段可直接运行的优化代码。

（5）求解操作的通用范式：在求解操作中，元黑箱优化的目标是将元智能体与优化器的功能融合为一体，即元智能体π_θ作为优化器接收s_i^t并输出当前步的候选解，实现对f_i的直接优化。此时，优化器$\mathcal{A}$即为元智能体π_θ（通常为神经网络），直接参与搜索与候选解生成过程。值得注意的是，与优化器空间中的元任务不同，在求解操作中，算法设计 ω 不再由元智能体输出，而是通过“训练即设计”的方式，在元智能体的训练过程中进行更新设计。因此对应地，算法设计空间Ω即为元智能体的参数空间，算法设计ω实际上为元智能体参数θ。此时的元目标可被表示为：

$$J(\theta)\approx\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{perf}\left(\mathbf{x}_i^t=\pi_\theta(s_i^t),f_i\right)\#(6)$$
$$\omega\equiv\theta,\quad\mathcal{A}\equiv\pi_\theta$$

具体而言，对于某个优化问题实例 f_i，在每一个优化步骤 t，首先通过状态特征提取函数 $\mathrm{sf}(\cdot)$ 获得当前状态特征 s_i^t。随后，元策略 π_θ 作为优化器接收 s_i^t 并输出当前步的候选解 $\mathbf{x}_i^t$，实现对 f_i 的直接优化。$\mathrm{perf}(\mathcal{A}=\pi_{\theta=\omega}(s_i^t),f_i)$ 表示基于当前状态 s_i^t 所生成的解 $\mathbf{x}_i^t$的优化效果。 在训练完成后，即可获得一个基于神经网络构建的黑箱优化器。该优化器具备迭代求解问题的能力，其行为等价于由人工设计的传统黑箱优化算法，但不依赖于显式的优化算子或固定结构。

3 优化器空间中的元黑箱优化算法

本节将结合第 2.2节中的各任务所定义的范式，按照算法设计空间的规模由小到大排列，依次介绍在优化器空间中的元任务：算法选择、算法配置、算法组装与算法生成的研究。图 3展示了元黑箱优化的总体研究分类。

3.1 算法选择

几十年来，受“没有免费午餐”理论的影响，没有一种通用的最优算法，使得关于算法选择的讨论经久不衰[20,48-49]。具体来说，这是因为不同算法的优化行为在不同问题中和优化过程中表现各异，从

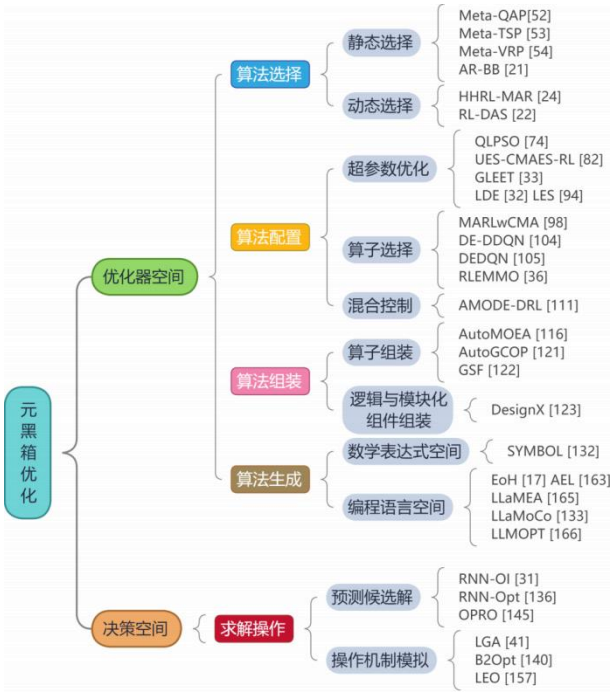

图3　元黑箱优化的研究分类与代表性工作

而导致显著的性能差异。因此，算法选择的目标是根据目标问题场景或者当前优化状态，从算法池中选择最合适的算法以实现更好的优化效果。在早期实践中，算法选择主要依赖人类专家基于经验和领域知识做出判断。然而，这一过程不仅劳动密集、效率低下，还受到知识和经验的限制，难以适应复杂动态的优化过程[50]。为突破这一瓶颈，研究者开始探索自动算法选择的方法，旨在一方面提升选择效率、降低人工成本，另一方面提升算法选择的细粒度，使其具备动态适应能力，更好平衡探索与开发[51]。特别的，在优化过程中实现实时、动态的算法切换，对人类专家而言具有较高挑战性，而元黑箱优化通过智能体进行自动学习与选择则成为一种有前景的解决路径。我们将以选择的细粒度为抓手，梳理现有元黑箱优化中算法选择的研究。

（1）静态算法选择：针对每一个具体问题实例，仅选择一个最合适的算法用于整个优化过程。在该设置下，算法选择是与优化步t无关，ω_i^t退化为常量ω_i，动态特征s_i^t退化为静态特征s_i。在该背景下，元智能体的学习范式与监督学习密切相关：学习有效元策略的一种直接方法是建立问题实例f_i的特征s_i与所选算法 ω_i 之间的映射关系 。由于算法池$\mathcal{A}$中的候选算法数量有限，许多早期研究将算法选择建模为分类任务：元智能体基于问题特征对问题实例进行分类，并将每一类别与与其在基准测试中性能最优的算法相对应[20, 21, 52 - 56]。具体而言，状态特征提取函数 sf 用于提取能有效区分问题实例的问题特征 s_i；再通过基准测试选出在 f_i 上表现最佳的算法作为标签，构成训练样本 (s_i, ω_i)。元策略 π_θ 被训练为分类器，其优化目标为最大化预测准确率，对应的性能评估函数 perf 即为分类精度。

在静态算法选择中，状态特征提取函数 sf 的设计影响了问题特征的有效性，是区分不同研究方法的核心因素之一，其具体形式通常依赖于所处理的问题类型。对于组合优化问题，Meta-QAP [52]、Meta-TSP [53] 和 Meta-VRP [54] 等研究构建了一个被称为组合优化问题元数据的信息集合。状态特征提取函数 sf 的作用是从每个问题实例中提取图结构相关的特征信息，例如节点属性、边的连接关系以及问题中的约束条件，并将这些信息统一存储于该元数据集合中，用于后续的策略学习与算法选择。对于连续单目标和多目标优化问题，探索性地形分析（Exploratory Landscape Analysis, ELA）已广泛应用于研究中[57]，用于描述问题实例的空间特征，如帕累托优势、凸性、以及问题的局部峰值和谷值[20,54,56]。状态特征提取函数 sf 利用 ELA 技术从问题实例中提取特征信息。在这些研究中，元智能体通常采用基础的分类模型， 如支持向量机（SVM）[58]、K 近邻（KNN）[59] 和多层感知机（MLP）[60]，用于预测问题实例的类别。此外，为了进一步探讨问题结构与优化器性能之间的关系， 文献[21] 中的 AR-BB 引入了符号回归技术， 以弥补黑箱问题显式表达式的缺失，并利用长短期记忆网络（Long Short-Term Memory, LSTM）[61]进行自回归预测，从而选择合适优化算法。

（2）动态算法选择：静态算法选择只考虑了问题特征，没有考虑优化状态，使算法不能满足优化过程中各阶段的要求。许多最新的元黑箱优化研究已经开始探索从逐一问题的静态算法选择到优化过程中

动态算法选择的拓展[22-24]。与第 2.2中介绍的一致：针对每一个具体问题实例，在每一优化步选择一个最合适的算法。在该背景下，元智能体的学习范式与强化学习密切相关：动态算法选择被建模为马尔可夫决策过程（Markov Decision Process，MDP）[62]，从而元智能体通过强化学习来学习动态选择。从静态到动态算法选择，显著区别之一在于选择细粒度的提升，这对应的是算法设计空间和状态空间的扩大。

在动态算法选择中，状态空间的离散与连续是区分不同研究方法的核心因素之一，这直接影响了到其所使用的强化学习方法。对于离散状态空间，表格 Q 学习是常用的强化学习方法[47] 。HHRL-MAR 引入了模拟退火算法中的 ε-greedy 策略：当生成的随机数大于预设阈值时，选择 Q 表中具有最大 Q 值的算法作为当前动作[24]。对于连续状态空间，元智能体通过许多深度强化学习方法进行学习，如 DQN [63]，DDQN [64] 和 PPO[65] 等。基于这些深度强化学习的元黑箱优化方法虽在学习效率上不如表格 Q 学习的高效，但其能够捕获更加细粒的状态与动作之间的关系。一个代表性的工作是 RL-DAS[22]，其使用 PPO 来进行训练，优化状态特征不仅包括静态问题特征，还融合了当前优化过程中的连续优化状态特征，以支持更灵活的动态算法选择。实验结果表明，RL-DAS 学习到的动态算法选择策略在多个基准测试中均优于算法池中任一单一黑箱优化算法，验证了在元黑箱优化框架下引入动态算法选择机制所带来的性能优势。

算法选择的典型应用场景主要集中在跨场景、工况多变的工程环境中，例如汽车结构抗撞性设计、供水网络优化等。在此类问题中，算法选择以“优化器为粒度”展开，因而具有较强的落地性。具体而言，研究者通常预置少量性能互补的优化器，再由元智能体根据实时绩效指标（如改善率、可行率、计算耗时） 以及问题实例的结构性特征（如规模大小、约束稠密度），在候选算法之间动态切换。这一模式的优势在于：无需改动底层优化器的内部细节即可实现动态调度，因而集成成本低、迁移门槛小、鲁棒性强。实证研究也验证了算法选择在多个复杂场景下的稳健性，例如在 18 个真实多目标工程问题[66] 以及二次分配问题[67] 的实验中，算法选择均表现出显著优越的泛化性与稳定性。

3.2 算法配置

算法选择在元黑箱优化中属于相对粗粒度的设计方式，其无法根据问题实例进一步调整所选算法的内部配置，因此整体性能受限于单一算法的能力。为了突破这一限制，亟需拓展至更大算法设计空间，进行更细粒度的算法设计，使优化过程中算法能够更加动态、精细地匹配问题特征。

算法配置是优化领域中的核心问题之一，因为几乎所有黑箱优化算法都依赖于一组对性能具有显著影响的超参数[68] 和可选算子[69]。为实现该过程的自动化，过去几十年间，研究者提出了大量具备自适应配置机制的黑箱优化算法，并持续推动该方向的发展。例如， JADE [70] 和 APSO[71] 等算法通过收集历史优化数据，统计评估超参数设置的潜力及算子使用的成功率，从而动态调整算法配置。这类方法在本文引言中已有讨论，其主要局限在于泛化能力有限、设计效率低下等问题。元黑箱优化通过使用元学习来训练算法配置的元策略，在优化过程中，元智能体动态调整算法的配置，从而克服了传统人工算法配置的局限性。

在元黑箱优化中，针对算法配置任务已有大量相关研究。为更系统地梳理现有工作，我们依据元学习层中算法设计空间 Ω 的类型，将算法配置方法划分为以下三类：1）超参数优化：元智能体用于动态调整优化器中的超参数设置；2）算子选择：元智能体用于动态选择优化器中的可选算子；3）混合控制：元智能体同时处理超参数配置与算子选择。与动态算法选择类似，在算法配置中，其常被建模成马尔可夫决策过程，元智能体的学习范式与强化学习密切相关。在各类方法中，我们以设计细粒度为主线，从状态空间与算法设计空间的离散性与连续性两个维度出发，对当前研究进行归纳与比较。

1）超参数优化：在进化计算中，探索－开发平衡是影响优化性能的核心因素之一[72]。探索旨在引导算法跳出局部最优，通过搜索空间中的多样性提升全局搜索能力；而开发则专注于围绕当前优良解进行局部改进，以提升解的精度。两者之间的平衡直接决定了算法的收敛速度与全局最优解的发现能力。超参数优化的目标在于根据优化过程的不同阶段，动态调整算法的参数配置，以实现探索与开发之间的有效权衡。与算法选择中的逐例选择类似，早期的研究多聚焦于为每个问题实例配置一个固定的参数组合。而随着研究的推进，当前主流方法已广泛采用公式 3 所定义的动态配置范式，即在每个优化步上根据状态特征调整参数。这使得算法能更好平衡探索与开发，从而增强了整体优化性能。因此在该设置下，算法设计空间Ω即$\mathcal{A}$的超参数空间。

a）离散状态空间与离散算法设计空间：在离散状态空间中进行离散算法设计本质上是一种人工配置算法的范式：专家根据经验将状态空间划分为若干档次，并据此调整一系列离散的超参数值。许多研究尝试通过学习方法来替代这一人工配置的过程，主要涉及两种强化学习方法：表格 Q-学习[47] 和 SARSA [73]。这两种基于价值的强化学习方法通过与环境的交互维护一个 Q 表，并在优化过程中不断更新其条目。一个重要的优势是，这些方法采用简单的 Q 表结构，能够高效收敛并保持一定的有效性。然而，这类方法仅适用于状态空间和动作空间（算法设计空间）均为离散（有限）的马尔可夫决策过程。在这些研究中，优化状态和算法设计分别被预定义为 Q 表的行和列。在优化层中，对于每个优化步骤 t，元智能体根据当前状态特征 s_t 和Q表中相应的条目，选择一个算法设计空间中的动作 ω_t。随后，通过执行该动作，生成一个状态转移 $\langle s_t, \omega_t, \mathrm{perf}(s_t, \omega_t, f), s_{t+1}\rangle$，并使用公式 7更新 Q 表：

$$Q(s^t, \omega^t) = \mathrm{perf}(s^t, \omega^t, f) + \gamma \max_{\omega \in \Omega} Q(s^{t+1}, \omega) \#(7)$$

以代表性工作 QLPSO 为例，在该方法中，元任务是算法配置，具体而言，任务是动态调整粒子群的拓扑结构[74]。QLPSO 在环形拓扑的粒子群中运行，其中状态空间 $\{L2, L4, L8, L10\}$ 表示粒子的邻域大小。对应的四种动作包括保持当前邻域大小或转换为其他大小。当拓扑调整成功并带来性能提升时，元智能体将获得奖励。类似的研究，如 RLDE[75]、qLDE[26]、RLMODE [76]、RLNS[77]、QFA[78]、QL-(S)M-OPSO[79]、LRMODE [80]、RL-SHADE [81] 等，均采用了类似的策略进行算法配置与优化。

b）连续状态空间与离散算法设计空间：尽管表格 Q 学习和 SARSA 方法简单且有效，但在一些元黑箱优化场景中，研究者追求连续优化状态，以实现更灵活的算法设计。这些场景中的马尔可夫决策过程具有无限状态空间，然而 Q 表结构无法有效处理这种情况。因此， 基于神经网络的 Q 智能体（例如 DQN 和 DDQN）被引入，以应对连续状态特征。在这种方法中，Q 智能体通过最小化目标 Q 函数与预测 Q 函数之间的估计误差来进行更新。其损失函数如下所示：

$$L(\theta) = \frac{1}{2}\left[Q_\theta(s^t, \omega^t) - \left(\mathrm{perf}(s^t, \omega^t, f) + \gamma \max_{\omega \in \Omega} Q_\theta(s^{t+1}, \omega)\right)\right]^2 \#(8)$$

以UES-CMAES-RL 为例，在该方法中，状态特征包括初始点与最终点之间欧几里得距离等 65 个连续特征，用以衡量采样点是否在搜索空间中移动以及是否产生良好的搜索效果[82]。元智能体采用一个基于多层感知机的 Q 智能体，该智能体包含三层隐藏层，分别具有 100、75 和 50 个神经元。在元学习层中，元智能体根据输入的状态特征，从由种群规模等 7 个超参数组合生成的 12 种预定义算法配置中选择一种作为当前的算法设计。该步骤后的性能提升作为奖励，用于通过公式 8更新元智能体。此外，类似方法的研究还包括 R2-RLMOEA[23], MADAC [83], CEDE-DRL [84] 等。

c）连续状态空间与连续算法设计空间：然而，无论状态特征的划分方式或参数组合如何调整，基于离散的状态特征空间与算法设计空间进行配置的粒度依然较为有限，难以实现对连续状态和参数的精细控制。在强化学习成功应用于连续控制之后，部

分基于强化学习的元黑箱优化研究转向了基于策略梯度的强化学习方法（如 REINFORCE[85]，A2C [86]，PPO [65] 等），以支持连续状态空间和算法设计。与离散空间相比，连续空间使得这些方法能够灵活地控制优化层中优化器的行为，从而提升整体优化性能。在这种方法中，引入了一个基于神经网络的元智能体 π_θ，该网络根据输入的状态特征输出算法设计空间上的概率分布。用于更新 π_θ 的梯度 $\nabla_\theta J(\theta)$ 计算公式如下：

$$\nabla_\theta J(\theta) = -\nabla_\theta \log \pi_\theta(\omega^t \mid s^t) \times \left(\sum_{t'=t}^{T} \gamma^{t'-t} \operatorname{perf}\left(s^{t'}, \omega^{t'}, f\right) \right) \quad (9)$$

以代表性工作 GLEET 为例，在该方法中，状态特征由一组结构化的特征集合组成，包括若干低层优化状态信息，如解空间/目标空间的密度、性能提升指标等[33]。对于群体中的每个粒子，采用三层 Transformer 架构的策略网络来输出粒子群优化算法中更新规则的系数 c_1 和 c_2 的后验高斯分布。在当前优化步骤中， 从输出的分布中采样具体的系数值，并根据优化结果给予相应的奖励。当一个优化回合（T 步）完成后，通过累计每一步梯度的总和（见公式 9）更新元智能体 π_θ。

除了强化学习外，元智能体同样可以通过监督学习来学习超参数配置。但一个显著区别是，监督学习需要通过直接梯度下降来训练元智能体，因此使用的训练问题是“白箱”且可微的合成问题。而在测试阶段，仅需通过与目标问题的有限次交互，获取由采样点 $\mathbf{X}_i^t$ 及其对应的评估值 $f_i(\mathbf{X}_i^t)$ 构成的状态特征，随后将其输入至预训练的元智能体，即可直接获得对应的算法配置。该过程中无需依赖梯度信息或其他关于问题结构的先验知识，因此可以直接用于黑箱优化问题。LDE 首次引入基于长短期记忆网络（Long Short-Term Memory, LSTM）的元智能体，通过监督学习，在每个优化步骤中提取隐藏状态特征，并在连续区间 $[0,1]$ 内输出超参数 F 和 Cr的取值[32]。此时状态特征空间和算法设计空间便从离散空间跃迁至连续空间，显著提升了参数配置的灵活性与表达能力。与基于强化学习的方法（如 GLEET）相比，LDE 在算法配置的探索能力上表现有限。此外，其训练过程依赖于预先收集的“白箱”问题及其标签，这一需求不仅增加了对专家经验的依赖，也显著提升了数据准备的时间成本。类似方法的元黑箱优化研究还包括 LADE[87]、LTO [88]、RLEPSO [89]、RL-PSO [90]、RLAM [91] 等。

神经进化（Neuroevolution，NE）作为机器学习的一个子领域，区别于监督学习在“白箱”问题上通过梯度下降训练网络，其神经网络是通过进化计算方法更新的[92]。因此和强化学习一致，不需要问题的梯度信息。特别的是，进化算法还被证明是一种可替代强化学习来解决马尔可夫决策过程的方法，尤其适用于那些具有长期影响的情形[93]。这启发了将神经进化用于元黑箱优化中。LES 是具有代表性的基于神经进化的参数配置算法[94] 。在训练过程中，其元学习层维护着一组基于 Transformer 架构的神经网络 $\{\pi_{\theta_1}, \ldots, \pi_{\theta_K}\}$。每个神经网络以优化过程的状态特征作为输入，输出优化层中具体优化器（如 CMA-ES）的关键控制参数（例如学习率与权重系数）。这些由神经网络输出参数实例化的优化器在训练集上的性能表现即作为对应网络的适应度值。随后，利用进化算法对参数集合 $\{\theta_k\}$ 进行迭代更新，实现对元智能体的训练。类似方法的元黑箱优化研究还包括 LQD[95] 、MetaDE [96] 等。

2) 算子选择：除了超参数，优化算法中所采用的算子同样对探索－开发平衡产生重要影响。算子选择的目标是在优化过程中动态切换优化器中所使用的算子，以更好适应问题特性与搜索阶段。现有大多数相关研究普遍构建一个离散的算法设计空间 Ω, 即若干典型变异算子组成的算子池，如DE/rand/2、DE/best/2 和 DE/current-to-pbest/1 等[97] 。为了适应不同类型的问题，相关方法主要在状态特征空间的表示形式与算子池构成上存在技术差异。由于算子池是自然离散的，因此根据这些工作的状态特征空间的表示形式，现有的工作可以分为两种主要形式：离散形式和连续形式。

a）离散状态空间：MARLwCMA 中首先引入了两个关键指标：连续两个优化步之间的种群多样性变化与性能提升幅度，作为动态算子选择的有效特征[98]。尽管这两个指标本身为连续变量， 但分别被离散化为五个等级，用于构建状态特征空间。基于离散化后的状态，元智能体从包含三个候选算子的算子池中选择最合适的算子进行下一步优化。在离散状态空间下，利用多个离散维度的组合以扩展算法设计空间是一种常见策略。例如，RLHDE 引入了解空间与目标空间中种群相对于初始种群和参考值的相对密度，分别衡量当前收敛趋势与性能改善。这两个密度指标被离散化为五级和四级，共同构成 5×4 = 20 种不同的优化状态[99]。与此同时，RLHDE 的算子池包含六种不同的变异算子，进一步提升了优化行为的多样性与适应性。类似方法的元黑箱优化研究还包括 RLMPSO[100]、DE-RLFR [101]、RLEA-SSC [102] 、RL-CORCO [103] 等。然而， 离散化方式在维度数量上存在权衡：一方面，多个离散维度的组合仍难以实现连续空间所具备的细粒度；另一方面，过多的离散维度会降低表格 Q 学习等方法的训练效率并引发不稳定性。因此，在设计中需要对离散维度数量加以控制。

b）连续状态空间：相比于离散形式，连续状态特征空间能够实现更精细的状态建模，为优化过程提供更准确的状态表示，从而使元策略能够做出更具针对性的决策。以代表性工作 DE-DDQN 为例，其提出了一种全面的的状态提取方程，总共计算出 99 个连续特征：前 19 个特征描述了优化进展和目标优化问题的特性，而其余 80 个特征是描述算子池中四个变异算子优化潜力的统计信息[104]。基于多层感知机网络的元智能体生成候选变异算子的 Q 值，并选择 Q 值最大的算子来进行下一步优化。在 DE-DDQN 的基础上，DEDQN 进一步指出，仅将状态特征空间从离散拓展为连续仍不足以全面提升策略性能[105]。为使元策略获得更充分且精炼的状态信息，DEDQN 强调特征选择的有效性与表达力，指出 DE-DDQN 中部分特征存在冗余，且难以刻画局部地形特性。为扩展至多目标优化， MOEA/D-DQN [106] 将 MOEA/D [107] 中的参考向量信息引入优化状态特征中。具体而言，对于某一解 x（对应的参考向量权重为 w），其状态特征被构建为 x,w 的组合表示。MOEA/D-DQN 的算法设计空间中有四种算子组合，涉及两个变异算子和两个交叉算子。元策略根据当前状态特征动态选择最合适的算子组合以指导下一步优化。针对多模态优化问题， RLEMMO 引入聚类机制以捕捉种群的邻域结构信息，并将其作为状态建模的重要组成部分[36]。状态特征空间由解的优化进展、种群的分布特征和以及基于聚类的邻域特征构成。类似方法的元黑箱优化研究还包括 SA-DQN-DE [108]、HF[109]、LCC-CMAES [42] 、RLDE-AFL [110] 等。

3) 混合控制：超参数优化与算子选择的共同目标在于通过配置调整实现优化器中探索与开发之间的平衡。为进一步提升性能，部分研究尝试同时控制二者，即混合控制。一种直接的实现方式是对两类配置任务分别建模，并由两个相互独立的元智能体分别进行控制。例如， AMODE-DRL 中分别采用基于 DDPG 的元智能体实现超参数配置，以及基于 DDQN 的元智能体完成变异算子选择，从而在强化学习框架下实现混合控制[111]。部分元黑箱优化研究进一步探索了算法配置的统一建模视角，特别是对超参数优化与算子选择的结合表现出浓厚兴趣。这一方向的核心动因在于：在联合包含超参数与算子的算法设计空间中进行策略学习，相较于对两者分别建模，有望获得更优的配置效果与更强的适应能力。然而， 混合控制也带来了显著挑战。一方面，联合空间维度高、结构复杂，显著提升了学习任务的难度；另一方面，元策略需具备同时建模多种异构决策变量（如连续超参数与离散算子选择）的能力。因此，设计精良的状态特征、学习框架与策略结构对于确保有效学习至关重要。

在当前的元黑箱优化现实应用中，算法配置是最为常见的研究模式。其典型应用场景主要集中于：优化问题所采用的算法框架已经固定，但整体性能对算子设置与超参数选择高度敏感。在此背景下，如

何通过智能化配置来提升算法表现，成为核心问题。这一模式已在多类真实与工业场景中得到验证。例如，在深空与行星际轨道优化的应用中展示了在极端非线性条件下通过动态配置提升收敛效率的可能性[99]；在电磁探测卫星任务调度中的应用则体现了在资源受限环境下，配置策略对可行解质量和任务完成率的关键作用[112]；在大规模热电联产经济—排放调度的应用中，合理配置算子与超参数有效平衡了经济成本与排放约束[113]；而在钢铁连续退火多目标排程等高度复杂的工业调度问题的应用中，算法配置的引入显著增强了优化器的稳定性与求解效率[111]。综上， 算法配置的现实应用表明，其在“固定框架 + 灵活调参”的场景下具有很强的实用价值，不仅能提升优化性能，还能有效降低专家依赖。

3.3 算法组装

我们已介绍的两类元任务——算法选择与算法配置，其核心目标均在于通过元智能体对算法、超参数与算子等关键要素进行动态调控，以实现在优化过程中对探索与开发行为的自适应平衡，从而在一定程度上缓解“没有免费午餐”定理所带来的算法性能局限。然而， 两者的设计是“自上而下”的：需要在已有的人工设计的算法框架下进行，即优化器$\mathcal{A}$是已有的人工设计的算法，这使得其依旧受限于一定的人工设计。而算法组装是“自下而上”的：通过构建模块化流程模板，使元智能体能够通过组件的组合，构成完整的新优化算法[29]。

由算子和超参数构成的组件是算法组装中的核心概念之一，其一开始在多目标优化领域被研究人员所关注[114]。如适应度分配、多样性保存和精英主义等操作在许多算法中被广泛的讨论与应用，研究人员尝试将各操作作为独立的组件，并构建统一的模板[115]。相比于整体设计一个算法，研究人员发现通过组件的组合方式来开发算法可以：(1) 通过将已有的算法通过替换组件的方式来提高性能；(2) 灵活地组合组件以快速应用于新的应用场景[116]。这一想法很快流行了起来，如在布尔可满足性问题求解器[117]、模拟退火算法[118]、随机局部搜索算法[119]等上进行了探索，并且组件的细粒度逐步提高[120]。然而这些研究都是人工算法组装，仅是提供了统一的组件模板，组合依旧需要人工进行。GCOP 将算法组装建模成组合优化问题，这为通过自动组合组件以构成新算法来优化问题提供了必要的理论基础[29]。

在将各组件构建好统一的模板后，研究人员开始探索自动化的算法组装。早期方法如 AutoMOEA 将其视为组合优化问题，使用 irace 算法来实现组件的自动搜索组合[116]。然而， 此类方法在泛化能力与灵活性方面存在显著不足，且难以扩展至高维的组件空间（即算法设计空间）。为克服上述局限， 后续研究开始采用学习驱动的方法来实现自动组合，催生了元黑箱优化中的算法组装任务。AutoGCOP 在 GCOP 框架的基础上引入表格型强化学习策略，实现了对组件的自动化组合[121]。然而，该方法仍局限于组合优化问题，适用范围较窄。一个代表性工作是 GSF，它探索了 DQN 与 PPO 两种深度强化学习方法来提升在更复杂的状态特征空间与更大的算法设计空间下的组合能力与泛化性能[122]。区别于 AutoGCOP 仅关注于组合优化问题，GSF 通过为进化算法预定义一个通用的算法模板，并结合强化学习策略，从候选算子池中为模板的各个阶段选择合适的算子进行填充，从而生成完整的优化流程。一个创新性的工作是DesignX，其引入了一个由两个基于 Transformer 的强化学习智能体组成的协同系统，首次实现了对算法结构组合与超参数控制两个子任务的联合学习[123]。该方法构建了当前表达能力最强、组合自由度最高的进化计算模块化系统之一，模块库涵盖遗传算法、进化策略、粒子群优化与差分进化等多类优化范式，包含共计 10 个大类、116 个子模块，具备高度的组合灵活性与统一性，能够广泛表达多种优化器结构，可在实现实现算法选择与配置等任务基础上实现算法组装，具备一定的跨元任务能力。此外，在多个优化场景（包括人工合成问题与现实问题）中，DesignX 所生成的优化器在性能上稳定超越人工设计优化器（如 CMA-ES[124]、SHADE [125]、NL-SHADE-LBC [126] ）及多个现有元黑箱优化方法

（如 GLEET[33]、GLHF[40]），展现出端到端自动算法设计的卓越的泛化能力与问题适应性。类似方法的元黑箱优化研究还有 ALDes[127], AutoOpt[128] 等。

在元黑箱优化的现实应用中，算法组装常见于问题结构高度复杂、且需要跨环节定制优化流程的场景。与算法选择和配置不同，算法组装的目标并非仅在固定算法间切换或对算子/参数进行微调，而是通过对多个算子、逻辑模块与优化步骤的组合，形成适配不同任务需求的整体求解流程。这种方式既能保证一定程度的可解释性，又能够在跨场景应用中展现出较强的灵活性与适配能力。已有研究表明，算法组装模式在多个真实复杂工业优化问题中展现出优势。例如，在港口卡车动态调度的应用中算法需要在多目标、多约束条件下协调车辆路径与调度规则，算法组装能够实现灵活的调度逻辑组合[129]；在云计算与工作流调度场景中，算法组装通过在任务分配、资源管理和负载均衡间灵活切换策略，提高了整体吞吐与系统稳定性[130]；而在复杂装配与物流协同中，算法组装则通过跨环节的优化模块重组，显著提升了供应链与生产调度的协调效率[131]。综上，算法组装作为一种“跨环节、可定制”的优化模式，为复杂工业环境中的自动化调度与资源配置提供了新的解决思路，展现出兼顾可解释性与泛化性的独特优势。

3.4 算法生成

算法组装是摆脱固定算法结构与优化流程的一种重要路径，拓展了元黑箱优化中的算法设计空间。然而，除此之外，还有一个更具开放性的问题逐渐受到关注：基于学习的系统能否根据目标问题场景自动生成具备竞争性性能的黑箱优化算法？这一方向被称为算法生成任务，是元黑箱优化中的一类新兴元任务。与算法组装不同，算法生成通过元学习得到一个参数化的元智能体，该元智能体可用于自动生成新的优化算法。生成结果可以是多种形式的优化器 $\mathcal{A}$，包括但不限于：一组数学表达式、一段程序代码等。换言之，算法生成中，元智能体的作用是直接构造优化器，而非对组件进行组合以形成优化器。其同样是一个“自上而下”的设计，输出是独立可执行的新的优化算法。

在元黑箱优化的算法生成任务中，元学习层中的算法设计空间 Ω 即为算法表示空间，用于表示待生成优化器的结构与行为。具体而言，Ω 可被定义为：(1) 数学表达式空间：以符号形式表示优化规则；(2) 编程语言空间：以特定语法结构定义优化函数或程序片段。这些形式反映了人类表达优化算法的两种主要方式：数学建模与编程实现。

首先，我们介绍基于数学表达式构建 Ω 的研究方法。此类方法的动因在于：GSF[122] 与 ALDes[127] 的算法设计空间高度依赖人工模板，可能限制了更具创新性的算法结构探索。为突破这一限制，SYMBOL 提出将黑箱优化算法的更新规则分解为基本数学符号与操作数的组合表达。具体而言， SYMBOL 构建了一个符号标记集，涵盖进化算法常用的基础元素，如 $\{+,-,\times,x,x^*,x^-,x^i,\Delta x,x^r,c\}$，分别对应基本运算符、当前解、最优解、随机解、步长调整等常见概念。在生成机制上， SYMBOL 设计了一个基于 LSTM 的元策略，通过自回归方式生成符号序列。每条生成序列可以解析为一条可执行的更新方程，用于在优化层中对目标问题实例进行迭代求解[132]。与传统固定结构不同，SYMBOL 支持针对每个问题实例、每个优化步骤动态生成更新规则，从而赋予优化器一定的自适应能力。实验结果表明， SYMBOL 在多个基准测试中均优于现有多种主流黑箱优化算法，展现出先进的优化性能。

其次，我们介绍基于编程语言构建 Ω 的研究方向。此类方法的动因在于：尽管基于数学表达式构建 Ω 显著提升了算法结构的灵活性，但生成的表达式仍需通过映射转换为可执行代码，受限于固定的代码空间与编程语言的结构。为此，近期研究提出直接以编程语言作为算法表示方式，借助大语言模型作为元策略，利用其强大的语言理解与生成能力，端到端生成可直接执行的优化器，从而进一步缩短从算法设计到应用部署的链路。尽管相关研究均依赖大语言模型，但在具体的学习方法、生成流程以及

面向的目标问题类型上存在差异。为了系统梳理这一方向，我们将在第 6 节中统一介绍大语言模型在元黑箱优化中的应用与发展。总之，用于算法生成的元黑箱优化工作在更具表达力的算法设计空间中运行。一些研究的实验结果表明，生成的算法与人工设计的算法相当，甚至更优。生成的黑箱优化算法不仅可以用于解决优化问题，还可以供人类专家进一步分析，为开发新的优化技术提供新见解[133]。

在元黑箱优化的应用中，算法生成的典型场景集中于缺乏可复用的领域启发式规则，或需要快速孵化新型优化器的情境。算法生成强调“从零到一”的自动化过程，这一范式为解决传统方法难以覆盖的复杂任务提供了新的可能性。已有研究在多个领域展现出算法生成的潜力。例如，在电力系统机组组合[134] 中，生成的优化器能够绕过传统启发式方法的局限，高效探索调度与约束间的复杂交互；在动态车间调度问题[135] 中， 自动生成的调度规则展示了较强的适应性与竞争力。此外，EoH [17] 与 AlphaEvolve [19] 等前沿工作，更是在缺乏成熟高效算法的传统复杂优化任务中取得突破性成果，进一步彰显了算法生成在现实场景中的巨大潜力。作为元黑箱优化最具创新性的方向之一，算法生成能够打破对专家知识和经验性启发的依赖，推动优化器设计迈向全自动化与创新驱动的新阶段。

4 决策空间中的元黑箱优化算法

截至目前，我们已介绍了在优化器空间中的四类元任务的研究。这四类方法具有一个直观的共性特征。无论是“自上而下”选择或配置一个黑箱优化算法，还是“自下而上”构建一个黑箱优化算法，其优化层中均依赖一个具体的黑箱优化算法作为基础优化器。然而在训练与应用过程中需同时管理元智能体与优化器，增加了系统的计算与实现成本。为缓解上述问题，部分研究提出了求解操作这一新型元任务。该元任务不再是在优化器空间中进行，而是直接在决策空间中进行。具体来说，其打破了对优化层中优化器的依赖，通过将元智能体本身建构为优化器，直接在优化过程中生成候选解，因此元智能体的输出实际上是在决策空间内。元学习层与优化层在结构上得以融合，使元智能体能够独立完成搜索与决策，从而简化了系统结构、降低了计算成本，并为更灵活的优化行为学习提供了可能性。

在求解操作任务中，一种直观的建模思路是将元学习过程视为一个候选解预测问题，通过神经网络直接输出当前步的解。由于优化过程具有迭代性，天然带有时间依赖性，因此可采用时序建模结构（如 RNN 或 LSTM）作为元策略架构，使模型能够在连续优化步骤中逐步生成候选解[31,136-137]。其对应的数学公式相对简单明了：

$$X^t, h^t = \pi_\theta(X^{t-1}, Y^{t-1}, h^{t-1}), Y^t = f_i(X^t) \#(10)$$

其中π_θ是一个RNN或LSTM网络，h_t是隐藏状态。 该思路最早由 RNN-OI 提出，其采用 LSTM 结构作为元策略，在每步迭代中直接预测候选解[31]。对于训练集中每个问题 f_i，RNN-OI 随机初始化一个初始解 X_0 并获取其评估值 Y_0，随后通过 LSTM 网络迭代生成下一步解，形成完整的优化轨迹。此时需要通过监督学习来训练基于LSTM网络的元智能体，因此使用的训练问题是“白箱”且可微的合成问题。perf(·)即为对应的损失函数：

$$L(\theta) = \mathbb{E}_{f, y_{1:T-1}}\left[\sum_{t=1}^{T} \min\left\{f(\mathrm{x}_t) - \min_{i<t}\big(f(\mathrm{x}_i)\big), 0\right\}\right] \#(11)$$

每一步的优化效果通过 $\mathrm{perf}(\cdot) = L(\theta)$ 计算以评估性能提升，并作为训练信号反馈至网络。一旦训练完成，后续无需依赖梯度或其他关于问题结构的先验知识，LSTM 即可作为优化器应用于黑箱优化问题上，并按照公式 10执行端到端的优化过程。由于端到端的推断过程，RNN-OI 在运行时间上通常优于传统人工设计的黑箱优化算法。类似的研究包括 RNN-Opt[136]、LTO-POMDP [137]、MELBA [138]、RIBBO [139] 等。

然而，基于 RNN 或 Transformer 的求解操作方法仍面临两方面的挑战：泛化能力有限与可解释性不足。一方面，现有方法的状态特征空间多依赖原始种群信息，易导致元策略对训练问题过拟合，难

以推广至未知问题。另一方面，所学习的元策略通常表现为“黑盒”系统，人类难以理解其参数 θ 与优化行为之间的因果关系，从而阻碍了对其学习机制与行为模式的深入分析。为应对上述问题，过去两年中涌现出一系列更具可解释性与泛化性的元黑箱优化方法。在特征建模方面，这些研究提出使用更高层次的抽象特征替代原始信息，如解空间与目标空间的分布特征、评估值的排序信息，以及体现优化动态的时序特征，以增强策略的任务迁移能力。在模型架构方面，相关工作进一步模块化地分解求解操作过程，将元策略的行为与经典进化算法（如遗传算法、差分进化、进化策略） 中的核心机制显式对齐。不同于直接预测候选解的单一结构，这类架构将优化行为划分为多个可控子模块，如“选择”、“变异”、“交叉”、“更新”等操作单元，使元策略能够在更清晰的设计空间中演化，并提升算法的可解释性与行为分析能力。例如， LGA 设计了两个基于注意力机制的神经网络模块，分别模拟遗传算法中的选择算子与变异率的自适应机制[41]。其中， 选择模块在父代与子代种群之间应用交叉注意力，所生成的注意力得分矩阵用于指导候选个体的选择概率；变异率模块则在子代种群内部施加自注意力机制，用于动态调整每个个体的变异强度，并且使用神经进化进行训练。在此基础上，B2Opt 提出了一种更完整的、端到端可训练的网络架构，显式模拟遗传算法中的主要操作过程，包括交叉、变异与选择[140]，其使用监督学习进行训练。

随着大语言模型的快速发展，其在自然语言理解与推理方面的强大能力为求解操作任务带来了新的研究机遇。近期研究广泛探索了上下文学习（in-context learning, ICL）在黑箱优化中的应用[45-46]，具体做法是通过精心设计的提示词，引导通用语言模型模拟优化器行为，从而在无需参数更新的情况下迭代生成候选解。这一方向将黑箱优化问题转化为语言模型的推理任务，使得优化过程本质上变为一个“提示－响应”循环，即模型通过理解历史优化轨迹与问题表述，在自然语言上下文中隐式执行优化逻辑。我们将在第 6节统一介绍大语言模型在元黑箱优化的应用。

在元黑箱优化的应用中，决策空间的典型应用场景主要集中于那些能够获取高保真仿真或丰富历史数据，并且更强调实时性与端到端性能的问题。在这类场景中，智能体不再局限于优化器空间的配置与设计，而是能够直接输出问题的候选解，从而实现更加直接和灵活的优化。已有的研究探索了在城市交通信号控制[141]、多机器人任务调配[142] 和动态柔性车间调度[1] 等复杂现实优化问题上的应用。

5 各类元任务总结与学习范式对比

5.1 各类元任务的总结

在元黑箱优化中，算法选择、算法配置、算法组装、算法生成与求解操作构成了五类基本元任务。尽管它们在具体组件设计上存在差异，但并非相互孤立，而是共同贯穿于元黑箱优化的核心目标之中：在减少对专家知识依赖的同时，逐步提升优化算法的整体性能。这种性能提升包含两个方向：一是增强泛化性，使优化器能够适应广泛的目标问题分布；二是提高针对性，根据具体问题实例动态定制优化策略。双层优化架构为这一对矛盾的性质提供了统一的实现机制：元智能体在目标问题分布上进行学习，提升了算法的泛化能力；而扩大算法设计空间、提升设计灵活度，使得针对性优化成为可能。贯穿五类元任务发展的两大特征是： 自动化程度的逐步提高与设计灵活度的持续上升。这种演进不仅体现在从算法选择到算法生成的跨越，也体现在每一类元任务内部的持续深化与细粒度演化。

图 4 展示了优化器空间中从传统人工设计到自动算法生成的跨越。除此之外，以算法生成任务为例，从数学表达式到编程语言构建 Ω，这一发展同样体现了自动化程度的逐步提升。端到端的全自动算法设计流程是未来研究的趋势，它不仅提高了算法设计效率，降低了资源开销，还促进了智能体在更复杂环境中的学习与适应能力。此外， 自动化的提升有助于进化计算在现实中的应用，尤其在一

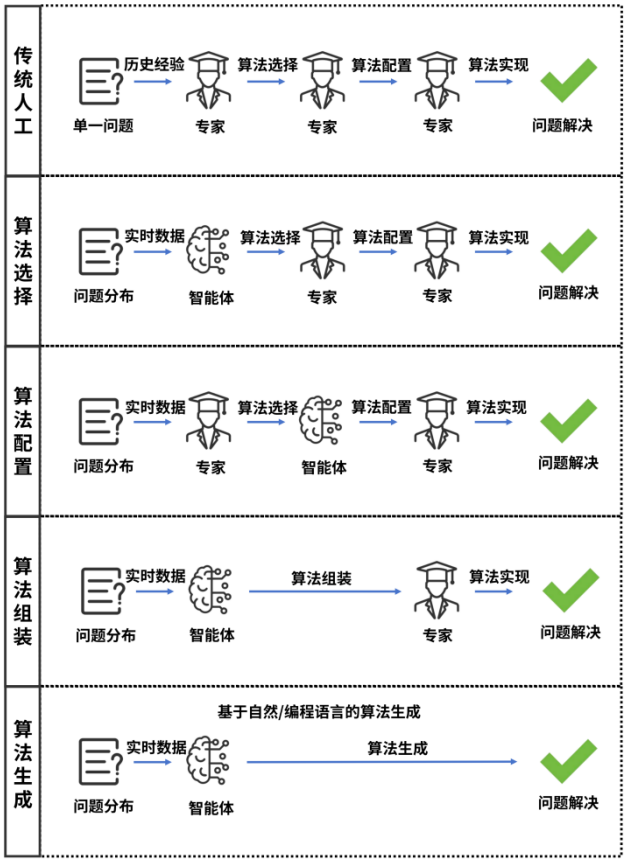

图4 逐步提高的自动化：传统人工到算法生成

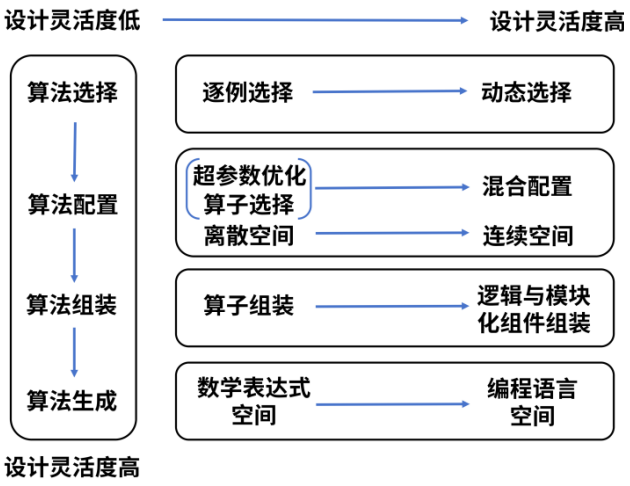

图5 持续上升的设计灵活度

些特殊优化场景中，如无人机群实时路径规划，供应链实时优化等等。这类场景要求算法能够实时调整并生成解决方案，这对于传统人工设计方法而言是一项巨大挑战。

图 5 从两个维度展示了优化器空间内中设计灵活度的持续提升。一个是纵向维度，代表元任务间

表 1 优化器空间与决策空间的方法对比

指标	优化器空间	决策空间
算法设计灵活度	高：支持模块化设计和定制，易于创新。	低：受限于预设的网络框架，缺乏对算法构建的自主控制力。
算法探索空间	有限：受限于预设定的算法空间。	广阔：深度网络可以逼近任意函数，具有强大的表达能力。
专家知识依赖	高：需要较多领域知识来设计算法框架。	低：数据驱动，对领域知识依赖低。
问题领域适应性	强：参数和流程可实时、显式调整， 响应问题变化。	弱：依赖固定网络结构，难以迅速调整以适应开放问题。
可扩展性	较好：易于构建维度无关算法、亦可通过增改模块扩展功能。	较弱：深度网络对输入与结构变更较敏感，扩展成本较高。
端到端优化能力	较弱：子部件设计需人工参与。	较强：直接优化最终目标。
可解释性	较强：算法机制透明。	较弱： 算法成为黑箱。
实时性与效率	一般：算法组合与调试可能导致效率低。	高：训练后网络响应快，适合实时应用。
输出结果多样性	多：可输出不同风格的结果。	单一：聚焦于特定任务的最优解。

的层级拓展：从智能体做选择的视角来看，优化器空间中的四类问题可被统一建模，但其选择所处的空间维度逐级扩展。算法选择与算法配置主要位于算法模板空间，算法组装则扩展到组件与逻辑空间，而算法生成则进入数学符号/程序设计空间。因此， 设计自由度的提升本质上是智能体做选择的空间维度的不断提升。另一个横向维度，代表单一元任务内的方法拓展。具体体现在两个方面：其一是算法设计空间的连续化，例如从逐例选择到动态选择，离散维度的配置拓展到连续维度；其二是决策对象的拓展，例如算法配置从单一算子选择扩展到混合配置，算法组装从算子组装发展为包含逻辑的模块化组件组装。总而言之，在优化器空间中设计灵活度的增加反映在算法设计空间的不断扩大，使得能够为具体优化场景量身定制更具针对性的算法，从而缓解“没有免费午餐”定理的影响。

由于优化器空间与决策空间的方法在设计理念上存在差异，表 1 从设计灵活度等九个方面对比与总结了在两类空间中进行自动黑箱优化算法设计的不同。总的来说，优化器空间的方法具有透明且可解释的设计过程，使研究者能够深入理解算法背后的机制。这种可控性使设计器能够根据特定需求灵活调整算法的参数和结构，从而增强了算法的适应性和创新能力。相对而言，决策空间的方法则在表达能力和高效的端到端优化能力方面表现突出，但其内部机制难以直观理解，缺乏对算法设计过程的自主控制能力，在面对新任务时的跨域自适应调节能力也相对较弱。

5.2 各类学习范式的对比

设计灵活度的提高也对学习范式和智能体提出了更高要求，需要更先进的学习方法与更强大的智能体来满足其学习需求。我们对现有的常见的四种学习范式：强化学习 (RL)、监督学习 (SL)、神经进化 (NE)、上下文学习 (ICL) 进行了对比。通过深入研究基于这四种学习范式的元黑箱优化方法的原始论文中提出的实验结果，并比较它们的具体实现，我们总结了一系列描述学习范式特征的重要指标。我们使用表 2 中的七个指标进行了比较。这些指标包括：（1） 开发难度，以代码实现的复杂度来衡量。（2）专业知识依赖，表示开发人员是否必须具备一定的专家级知识。（3） 数据利用率，以同一批数据获得的理论性能增益来衡量。（4） 训练效率，以训练时间来衡量。（5）推理效率，以解决优化问题所需的时间来衡量。（6） 优化表现，经验优化结果。

（1）在开发难度方面，若已具备可直接调用的大语言模型，上下文学习无需额外的参数训练与模型结构改造，其主要工作集中于提示词设计与少量示例组织，因此整体开发流程相对简化，开发难度最低。强化学习与监督学习这两类方法在构建训练流程时，均有大量成熟的算法框架与工具库可供使用。这使得其开发难度相对适中，关键挑战更多来自任务建模与特征设计，而非训练流程本身。相比之下，神经进化方法的开发难度最高。由于进化计算需要

表 2 学习范式比较

属性	RL	SL	NE	ICL
开发难度 ↓	★★	★★	★★★	★
专家知识依赖 ↓	★★	★★	★★★	★
数据利用率 ↑	★★★	★★	★★	★
训练效率 ↑	★★	★★	★	–
推理效率 ↑	★★	★★	★★	★
优化表现 *↑	★★★	★★★	★★	★

* 参考 MetaBox 等研究的测评[16,143-144]。

多轮迭代优化种群，同时涉及大量神经网络或优化器实例的评估，开发者须处理复杂的算子耦合关系与超参数调优问题。这不仅对计算资源提出更高要求，也对稳定高效训练等带来严峻挑战。

（2）在专家知识依赖方面，神经进化方法依赖于进化计算算法对元智能体进行训练。开发者需要深入理解神经网络架构设计、个体编码方案以及各类进化算子与超参数。由于训练依赖进化计算过程，专家知识在设计与调优中起到核心作用。强化学习方法主要依赖于状态表示与奖励函数设计。前者需要把复杂优化问题抽象成合适的状态空间，后者则要求奖励能合理引导优化器学习，这两者均需要较强的领域建模能力。监督学习关键在于高质量数据集的收集与标签标注。在现实优化问题中，获取大规模标注样本往往需要专家参与或仿真环境支持，因此其知识依赖主要体现在数据准备环节。上下文学习的专家知识需求相对最低。其主要依赖于提示词设计，并且大多基于自然语言描述和少量示例。因此相比其他方法，其专家知识较少。

（3）在数据利用率上，强化学习在无监督环境下训练元智能体，可通过时间差分/自举机制以及价值函数/优势函数估计，从同一交互轨迹中提取多重、密集的学习信号。此外，离线强化学习还能够反复利用历史数据，进一步提升样本利用率。而在元黑箱优化的背景下，监督学习的数据利用率总体较好，特别是在静态算法选择等任务中，数据集可多次用于训练[52-54]。然而，当前以 GLHF[40]、RNN-Opt [136] 为代表的研究中，虽然在单次监督信号中利用效率良好，但样本往往只能使用一次，难以重复利用。同

时，其对分布外数据的泛化能力有限，相比强化学习仍存在差距。神经进化每一代进化仅依赖采样点与适应度值来驱动更新，与监督学习类似，样本通常只能使用一次，数据利用率相对较低。而上下文学习依赖于有限的上下文窗口，对历史数据的复用主要通过提示词设计实现。但由于上下文长度限制和模型的遗忘效应，其持续利用历史数据的能力有限，因此数据利用率最低。

（4）在训练效率方面，强化学习一方面借助梯度化更新与经验回放机制，能够在参数更新频率上保持较高水平；另一方面，其支持在多个环境实例间并行采样，显著提升了总体效率。监督学习则依托梯度下降与批量并行等成熟技术，同样具有较高的效率表现。神经进化通常依赖大量的候选个体评估来驱动种群迭代，而这些评估往往开销较大，使得总体效率低于强化学习与监督学习。而上下文学习依赖于已预训练的大语言模型，因此推理阶段仅需通过提示词与少量示例即可完成任务配置，无需再进行参数更新与额外训练，几乎不存在训练开销。

（5）但在推理效率上，前三种学习范式多为一次前向/简单采样，延迟小、吞吐高，而上下文学习的依赖大语言模型的推理过程，本身计算开销较大。为了提升生成结果的可靠性，常常需要较长的上下文输入以及多轮推理/采样，这显著增加了推理延迟与计算负担，在实时性要求较高的应用场景中可能成为瓶颈。

（6）在优化表现方面，现有研究的测评结果表明[16,143-144]，不同学习范式在性能、稳定性与适用性上存在显著差异。总体而言，基于强化学习的方法表现最为稳健，以 RL-DAS [22]、DE-DDQN [104]、RLEPSO [89]、GLEET [33]等为代表的研究在多数元黑箱优化基准上均优于其他方法，尤其在动态环境与分布外任务中展现出更强的自适应能力，因此成为当前的“高线”基准。相比之下，监督学习方法在具备高质量训练数据或教师轨迹的条件下（如 GLHF[40]、RIBBO [139]），能够达到接近甚至匹敌强化学习的性能，并在静态算法选择等任务中具有较高的准确性和稳定性，但在纯黑箱、无教师的设定下，受制于长序列难训和割裂训练等问题，基于传统监督学习的方法（如 RNN-OI[31]）的表现明显逊色。神经进化方法虽然在搜索空间探索与复杂非凸问题求解方面具备独特优势，但在收敛速度和训练稳定性上通常落后于前两类方法，且其高昂的评估开销使其在多任务训练与大规模应用中难以保持样本效率与迭代速度。上下文学习则依赖大语言模型的即插即用能力，能够在零样本或少样本场景下提供一定的灵活性，但受限于提示工程、上下文长度以及模型固有缺陷，其代表性方法（如 OPRO [145]、LMEA [146]）往往需要大规模提示迭代才能取得稳定性能，即便经过指令微调，仍存在 5%－10% 的代码错误率和数学推理脆弱性，导致其在稳定性和工程可用性上均难以与其他方法相媲美。综合来看，强化学习与监督学习在复杂优化任务中展现出更优与更稳健的综合性能，而神经进化与上下文学习则更多体现为探索性和灵活性上的补充。

总而言之，每种学习范式都有其优缺点，开发时需要结合数据量和元任务等多方面进行综合考虑。

6 大语言模型在元黑箱优化中的应用

尽管现有元黑箱优化研究已广泛采用强化学习、监督学习与神经进化等学习范式，在算法选择、配置等方面取得了重要进展，但从“问题描述（输入）—优化方案（输出）”的端到端视角审视，上述范式仍普遍受到两类结构性约束。其一，问题描述往往依赖严格的接口定义与可执行代码，这在保证可控性与可复现性的同时，提高了使用门槛并强化了对领域与编程知识的依赖。其二，优化方案通常被限定为结构化的动作空间或预先参数化的搜索空间（例如固定算子库、有限超参数集合或特定表示形式），从而在一定程度上限制了算法设计的表达能力，尤其难以直接生成跨编程语言、跨框架的可执行程序与更开放形态的求解流程。

近年来，大语言模型（Large Language Models, LLMs）凭借其卓越的语言理解与生成能力，大语言模型在数学推理[147]、代码生成[148]及科学发

现[149-150]等诸多高阶认知任务中展现出强大的统一表征与生成能力，推动了多个研究领域的技术革新，已成为人工智能领域最具代表性的技术之一。在优化领域，已有工作从不同角度验证了大语言模型处理优化相关任务的潜力[43,151-152]。基于大语言模型的元黑箱优化的优势在于，人类对问题与算法的描述本身是在语言空间内进行的，而 LLM 在预训练阶段已获得了对语言的理解能力。使得用户几乎不需要专业知识——只需用自然语言描述优化问题，大语言模型即可完成其余部分。这可以降低优化算法的使用门槛，使优化算法能够在各行各业即插即用。并且其输出形式更为开放，可生成不同编程语言与框架下的可执行实现，并支持对程序结构与复杂度的迭代改写与优化。如何利用好大语言模型在预训练阶段所蕴含的通用知识与推理能力是研究的关键。在元黑箱优化中，大语言模型的研究主要集中于求解操作和算法生成这两个元任务，区别在于大语言模型在优化过程中所担任的角色和具体如何构建有效的上下文提示。

6.1 大语言模型在求解操作中的应用

在求解操作任务中，大语言模型被直接用作优化器，其通过将优化历史作为上下文提示输入，从而迭代生成候选解以优化目标问题。一个开创性的工作是 OPRO，它首先为大语言模型提供问题表述和历史优化轨迹的自然语言描述，然后通过反复提示大语言模型根据提供的上下文来进行迭代优化[145]。对于每次迭代，任务描述根据具体问题类型来定制，包括用自然语言描述的目标函数和约束条件。上下文示例包括当前迭代之前的优化轨迹。任务指令明确指定大语言模型的目标为生成优于当前历史最优解的候选解，这一过程本质上等价于在进化算法中扮演搜索算子的角色。类似的研究还包括 LMX[153]，EvoLLM[154]，MOEA/D-LLM[155]等。为摒弃传统大语言模型方法中通过温度参数调节随机性的方式[156]，LEO 提出了一种结构化机制，通过构建“探索池”与“开发池”以实现对探索与开发的有效平衡，从而兼顾解的多样性与收敛性[157]。尽管现有研究在低维优化问题上展示了大语言模型在求解操作中的潜力，但该范式仍面临显著的计算开销问题[43]。一方面，求解操作虽然减少了对手动算法设计的依赖，但在处理更高维度或结构更复杂的问题时，其迭代式交互机制可能导致大规模文本生成与频繁调用模型，从而显著拉高推理时间与成本。另一方面，大语言模型的上下文窗口存在长度限制，复杂问题下快速增长的提示信息可能超出其上下文承载能力，进一步影响优化过程的稳定性与效率。

6.2 大语言模型在算法生成中的应用

在算法生成任务中，大语言模型被视为算法生成器，通过结构化的提示词输入，引导其生成完整的优化算法。生成的算法随后被实例化并用于优化目标问题，实现了从自然语言描述到可执行优化流程的自动转化。这一范式虽近年来因大语言模型的发展而获得广泛关注，但其思想可追溯至遗传编程，即在源代码空间中演化计算机程序的早期研究[158]。

在此基础上，函数设计成为最早探索的方向之一，即通过大语言模型生成优化算法中某一关键代码片段。利用 CodeLLM 的语义推理能力进行程序演化，已有多项代表性研究取得初步成果。例如，FunSearch[159]和 EUREKA[160]分别在启发式函数自动生成和强化学习中的奖励设计方面做出了尝试。继这些研究之后，研究人员开始探讨利用 CodeLLM 生成完整竞争性优化程序的可能性。OptiMUS 使用模块化结构的大语言模型智能体来制定和解决（混合整数）线性规划问题[161-162]。AEL[163]和 EoH[17]是两个代表性的工作，其灵感源自大语言模型的演化能力，均将多个算法维护为一个解种群，并通过促使大语言模型对先前算法的代码实现执行变异和交叉操作来演化这些算法。进一步地，ReEvo 引入短期与长期反射机制，以更有效地利用历史生成算法的信息，提升演化效率与生成质量[164]。LLaMEA[165]将该范式扩展到连续黑箱优化问题。LLaMEA 构建了一个基于性能指标与运行时反馈的迭代式框架，通过大语言模型对算法代码进行生成、变异与选择，实现对优化算法的持续演化。LLMOPT[166]与

MEoH [167] 探索了在多目标优化问题上结合大语言模型与进化计算进行迭代演化以实现优化算法自动生成的可行性。两者均致力于构建一个以大语言模型为核心的演化框架，通过引导模型对候选算法进行生成与改进，从而逐步进化出适用于多目标优化场景的高性能优化器。类似的研究还包括 Hsevo[168]，LLM-LNS [169] 等。

除了在求解操作和算法生成两个元任务中，大语言模型还在算法选择[170]、特征工程[171]、超参数优化[172] 和问题集生成[173] 等方面有所应用。

6.3 大语言模型当前在应用中的局限性

大语言模型在上述两个方面的应用在降低算法设计成本与使用门槛方面展现出一定潜力，但现有大语言模型在计算开销等方面仍存在着许多局限。

（1）计算开销高昂：以 OPRO[145] 为例，其在包含约 6k 个低维问题实例的问题集上，平均消耗 token达 115k [133]，使得单次运行的开销可达十余至数十元人民币量级。并且能解决的问题局限于 20 维以下的低维优化问题，难以直接扩展到更高维或更复杂约束的黑箱优化任务。就优化性能而言，在CEC [174] 等标准复合低维基准测试上，现有实验结果表明 OPRO 的整体表现显著弱于非大语言模型的元黑箱优化方法。例如，在相同评测设定下，其效果仅为 RL-DAS[22] 的三分之一量级 [16,144]。

（2）优化领域知识缺失：当前通用大语言模型在黑箱优化算法设计中普遍缺乏关键的领域知识与归纳偏置，会产生推理幻觉，方案同质化等问题。已有研究观察到，GPT-4 在为多类复合基准问题生成求解代码时表现出显著的“算法偏置”，对大量不同问题实例都一致倾向于选择同一种特定的优化器（SLSQP[175]），同时，其单次生成程序的不可执行比例高达约 70 % [133]。现有的研究已对此开展了探索[133,176]，一类代表性思路是使用 LaTeX、Python 等形式化且结构化的语言，而非自然语言，对优化问题进行描述，从而减少语义歧义并提升问题表述的精确性与一致性。在此基础上，将问题空间映射至模型的隐表示空间后进行指令微调，使不同问题之间的相似性不再仅由表层文本描述决定，而能够更好地反映其在优化结构与求解难度层面的内在关联，从而提升算法匹配的合理性。此外，通过对大语言模型进行领域相关的知识注入，可显著增强其对优化概念、算法结构与程序逻辑的理解能力。

（3）设计的可解释性不足：除上述两个代表性问题外，大语言模型在元黑箱优化中的可解释性同样逐渐成为研究关注的重点。一方面，由于算法设计过程本身具有高度的结构复杂性与长程依赖性，仅给出最终生成的算法往往难以揭示其设计动机与演化逻辑。为此，部分研究尝试在算法生成的同时，引导大语言模型输出对其设计决策的反思、解释或点评信息，从而构建一种以自然语言形式表达的“语言梯度”，以辅助理解模型在不同设计阶段的偏好与调整方向 [164]。另一方面，也有研究从程序分析的角度出发，利用抽象语法树（Abstract Syntax Tree）等结构化表示，对算法代码的生成与演化轨迹进行可视化分析，以刻画算法结构如何随迭代发生变化，从而为理解算法生成过程提供更直观的解释手段[177]。

综上所述，现阶段的大语言模型的研究需要在系统性分析的基础上稳步推进，而非对其能力进行过度夸大或在缺乏充分论证的情况下盲目应用。

7 当前挑战与未来研究方向

自动化程度、灵活性与通用性构成了元黑箱优化方法发展的核心驱动力，也是其区别于传统人工设计算法的部分关键特征。现有的研究在这三方面仍然存在着不少局限性与挑战，本节我们将从各个实现此目标的潜在方法出发，指出现有研究的不足，并提出未来的研究方向。

7.1 问题理解能力提升的三重路径：问题集构建、知识注入与多模态输入

模型能否准确、全面地理解目标优化问题，是其能否生成高质量、专业化算法设计的关键前提。为提升模型的理解与建模能力，优化问题集的构建、知识注入以及信息输入方式的革新构成了三条核心路

径。

（1）问题集方面，相较于依赖专家经验进行手动设计的传统优化算法，元黑箱优化方法以数据驱动的方式实现自动算法设计，因此对大规模、高质量且多样化的训练问题集提出了更高要求。然而， 目前可用于训练的数据集仍较为匮乏。一方面，由于传统的黑箱优化算法不存在训练阶段，多数传统基准问题集往往并不具有在目标问题分布上抽样形成问题实例这一功能[178]。这些基准问题集在当前元黑箱优化研究中常作为构建具有这一功能的基准问题集的基础，又或用来作为元黑箱优化算法的测试集。但这也使得现有的元黑箱优化基准问题集的构建还依赖于原有基准问题集，尚未有一个可以自动依据目标问题来构建其对应的目标问题分布，并在此分布上抽样以形成问题实例的基准问题集。另一方面，当前多数元黑箱优化研究主要聚焦于低维、无约束的连续黑箱优化问题，与实际应用中愈发复杂的任务场景（如带约束、多目标、大规模优化问题）之间存在显著脱节。更为关键的是，现有的大多数问题集本身往往是由专家人工设计的复合问题，这在一定程度上与元黑箱优化“摆脱专家依赖”的初衷相悖，构成了方法理念与训练资源之间的矛盾。如何自动构建大规模、高质量且多样化的问题集？一种可能的方式是利用智能体自动构建针对性的问题实例。特别是，借助神经网络强大的表示与建模能力，有望在捕捉目标问题分布特征的基础上，实现问题实例的自动生成，从而构建规模化、分布合理、类型多样的问题训练集。这类问题集的可用性是推动元黑箱优化走向自主智能算法设计的关键基础。

（2）在知识注入方面，大语言模型为元黑箱优化任务提供了新范式。现有的大语言模型通常并未针对黑箱优化任务进行领域特定的知识注入，缺乏对优化理论、求解策略及问题结构的深层理解。这一限制制约了其在元黑箱优化任务中的性能表现。尽管开发面向该领域的大模型或进行从头训练可作为潜在解决方案，但这类方法面临极高的计算资源需求与数据准备成本，实际应用中难以推广。如何以低成本的实现对大语言模型的优化领域知识注入呢？为解决这一问题，近年来出现了针对性地对通用大语言模型进行高效微调的研究方向。其中， LLaMoCo 提供了一个值得关注的路径：通过指令微调将通用大语言模型转化为具备专家级优化建模与编程能力的垂直模型[133]。实验证明，微调后的中小规模模型（如 codeGen-350M）在优化算法生成任务上甚至优于未经微调的大型模型（如 GPT-4），表明特定领域知识在模型能力发挥中起到关键作用。另一种具有前景的路径是结合检索增强生成（Retrieval-Augmented Generation, RAG）技术构建外部知识库[179]。通过构建包含经典优化问题、物理建模、工程背景等内容的高质量 RAG 库，大语言模型可在生成过程中动态调用领域知识，以增强问题理解与求解的针对性。

3）在信息输入方面，引入多模态输入机制被认为是未来的重要方向[180]。以大语言模型为例， 当前大多数利用大语言模型的元黑箱优化任务中仍依赖纯自然语言作为输入媒介，然而这一方式存在显著局限。首先，对于多数复杂的优化问题而言，使用自然语言准确、全面地描述问题本身具有不小挑战。一方面，不同用户在自然语言表达上的习惯、专业水平与领域认知存在显著差异，导致问题描述在语义层面高度多样且易含歧义；另一方面，诸多实际黑箱优化问题（如工业调度、结构设计等）本质上具有高度结构化特征，难以用自然语言直接表达其全部约束、目标与环境信息，从而对用户提出了较高的表述门槛。这种语义表达的非结构性也在一定程度上限制了大语言模型对问题的充分理解与有效建模。更为关键的是，在现实优化场景中，自然语言并非最常见的信息形式。大量优化问题的输入与反馈以表格、图像、时序数据等结构化或感知模态的形式存在，典型如财务数据、实验观测结果等。这类信息往往在优化建模前就已经存在并被用于其他任务（如统计分析），因此构建以多模态为基础的输入接口，不仅可以为大语言模型提供更为丰富、准确的问题表征，还能大幅降低用户的表达负担与使用门槛。

7.2 零人工干预的模块化元黑箱优化系统

尽管当前元黑箱优化方法已在算法配置、选择、组合与生成等方面实现了不同程度的自动化，但其整体流程仍在多个关键模块上依赖专家经验。例如，如第 7.1 节所述的问题集构建，尚需大量人工干预。此外，状态特征构建、奖励函数设计等模块也高度依赖领域专家的先验知识与设计经验。这种依赖不仅抬高了元黑箱优化系统的开发门槛，也在一定程度上限制了其通用性与可扩展性。如何进一步降低元黑箱优化各模块的所需专家经验呢？理想的元黑箱优化方法应当具备结构与功能上的“涌现性”，即相关组件能够在训练过程中自发形成，或通过预训练获得可在多任务场景中迁移的通用型模块。通过构建具有高度可插拔性与自适应性的模块系统，可有效摆脱人工设定与先验依赖，使元黑箱优化向“零人工干预”范式演进。以状态特征构建为例， 目前广泛采用的特征类型包括 ELA [57] 和 FLA [181] 等人工构造特征。近年来的研究如 DeepELA [182] 和 NeurELA [183] 提供了另一种思路：利用神经网络从问题样本中直接提取状态特征，不仅保持了特征的判别能力，还大幅降低了计算成本。这一进展表明，数据驱动的特征学习方法相比传统手工设计在效率与效果之间可实现更优平衡。这一思路为元黑箱优化的未来发展提供了启发：是否可以将更多关键模块——包括状态建模、策略表示、奖励构建乃至优化目标重构——全面转化为数据驱动的自动学习过程？如果能借助深度学习、迁移学习等技术逐步构建出具备自组织、自学习与自演化能力的模块化系统，元黑箱优化将真正迈向自主智能设计新阶段。

7.3 面向复杂任务的高效学习与训练机制

面对更大规模、更复杂的问题场景，元智能体亟需更强大的学习方式，以有效缓解训练过程中带来的计算与资源压力。现有大多数研究集中于低维连续黑箱优化问题，使得效率问题尚不显著，但在大规模优化任务中，函数评估次数常达 10^6 量级，是低维问题的十至百倍，且评估耗时通常随问题复杂度呈指数级增长这一评估成本的显著膨胀不仅导致训练资源的大量消耗，也严重限制了现有方法在高复杂度任务中的可扩展性与实际可用性[184-185] 。如何在资源受限场景下实现对复杂任务的学习与模型训练呢?

在高效学习方面，可探索如具备跨优化步泛化能力的元学习方法，使得智能体无需经历长序列优化过程即可学习到高质量设计策略。此外，引入迁移学习[186]、多任务学习[187] 和持续学习[188] 等先进的训练机制，可促使元智能体在较为简单或低维的问题场景中获得通用性强的表示能力，进而迁移至更复杂、多样的问题中进行算法设计： Q-Mamba 探索了离线学习， 以离线数据为核心，混合“开发”与“探索”轨迹，并用保守 Q-learning（CQL 类）缓解分布移位； 同时将高维联合超参分解为自回归序列，用 Mamba 进行长依赖建模，将复杂决策拆为序列子决策，降低维度诅咒并增强对长程依赖与策略一致性的建模能力，实现对跨域任务的零样本迁移[189]；LiBOG 探索了终身/持续学习，将多分布问题视为非平稳任务序列， 跨任务共享状态与动作并采用 EWC（参数级巩固）与 EBC（行为级巩固），并利用归一化的景观指标与尺度无关的目标，减少任务间量纲差异引入的伪差异，在持续暴露分布漂移的过程中抑制灾难性遗忘、提升长期稳健性[190]。这些探索展示了出除扩大问题集的多样性外，进一步提升智能体分布外泛化的可能性，以降低训练资源的需求，推动元黑箱优化方法在更广泛应用场景中的落地。除此之外，多样化的元目标等设计，如在现已有的元目标上再添加上最小化算法的计算复杂度等要求，亦为训练更强大的元智能体提供了潜在途径。

在训练机制方面，MetaBox [143-144] 作为首个专门用于元黑箱优化方法开发与评估的基准平台，在元黑箱优化并行训练与测试层面开展了探索：基于“Tianshou”[191] 开发了“vectorized environment”实现了并行化训练，而在测试则基于“Ray”[192] 实现了并行化测试的效果。该平台提供的系统化、高效的并行训练与测试方案，在实际运行中分别将效率提升了约 10 倍与 40 倍，有效缓解了复杂任务下的训练瓶

颈。除此之外， 在优化算法理论层面，Matrix-Based Evolutionary Computation[193] 将演化算子重写为稠密/稀疏矩阵运算以充分发挥 GPU 的并行优势。在优化算法并行评估层面，EvoX[194] 在工程实现上将“进化算法的一次迭代”重构为可并行的工作流，将种群与适应度均以张量形式组织，可实现候选个体评估的自动并行。这些在优化层上并行化的探索可进一步加快元黑箱优化的算法训练与测试的效率。未来一方面可在并行训练机制上进一步优化，如针对同构问题实例集采用向量化与批处理，而异构问题实例集则采用多线程异步并行等技术；另一方面，可探索将硬件信息纳入状态空间，使得算法设计过程不仅考虑任务的特点，还能充分考虑硬件资源，从而从算法-硬件协同设计的视角进一步提高现有元黑箱优化的训练与测试的效率。值得一提的是，随着国产 GPU/AI 加速器生态的成熟（如昇腾、摩尔线程等），元黑箱优化的训练与评估过程可通过国产化部署并结合算子级并行化、通信/调度优化等工程手段，以构成“国产化部署 + 算法—硬件协同优化”体系，为后续低成本、可推广的算法实现提供了现实路径。

7.4 从建模到求解的端到端一体化元黑箱优化

值得关注的是，当前元黑箱优化研究多聚焦于优化器自身的设计与生成，普遍默认目标优化问题已经被明确建模且可直接进行评估。然而在实际应用中，问题建模往往才是优化流程的首要环节，无论是依赖专家手工构建，还是基于数据驱动的建模方法。因此，优化器设计与问题建模之间的断裂构成了一个关键的研究空白。面向实际问题场景，理想的元黑箱优化方法应具备从问题端到方案端的整体感知与处理能力，涵盖问题建模、特征提取与定制化算法设计的完整闭环。构建具备全流程感知与处理能力的端到端一体化优化系统，将为提升模型在真实复杂任务中的适应能力与问题解决效率提供广阔的研究空间。

然而，实现这一路径面临诸多挑战。如第 5节所述，现有元黑箱优化方法在不同元任务中呈现出逐步增强的设计自由度，表现为从静态算法选择、到动态参数配置，再到算法生成。但即便是在最具灵活性的算法生成任务中，当前方法大多仍局限于对优化器某一两个维度的局部设计，如变异机制等，尚难支持多层级统一建模与全结构生成。理想状态下，元黑箱优化系统应具备跨层级的自动化设计能力，涵盖优化器整体流程的结构组织、算子模块的组合调用、模块内部超参数的自适应调节，乃至不同编程语言（如 Python、Matlab）下的自动代码生成。

大语言模型在此背景下展现出潜在的革命性优势。其卓越的语言理解与结构化生成能力，使其具备承载“问题建模—方案设计—代码实现”全流程的能力，成为构建端到端元黑箱优化系统的有力候选。然而，当前大语言模型在该方向的应用仍面临显著挑战，亟需系统性突破。一个典型的例子是，大语言模型的研究普遍存在“大模型做小事”的低效困境[195-197]。在多个元任务中，所需执行的功能本质上较为简单，传统模型或轻量级方法已足以胜任。但大语言模型的部署往往伴随着极高的资源消耗，其推理过程依赖冗长的提示词构造与复杂的上下文建模，从而带来巨大的计算成本与能源开销。上述事实表明：在元优化任务中，大语言模型的真正价值更可能体现于需要语义理解、结构推理与跨任务策略生成的复杂环节，而非用于重复执行低层次、可替代的搜索与配置计算。除此之外，国产大模型能力的快速进步为元黑箱优化提供了更低成本的推理路径。部分国产模型（如 DeepSeek [198]，Qwen [199]，Kimi [200] 等）已通过分时段折扣等手段降低 API 使用成本，从而在需要大量迭代调用的“生成—评估—再生成”闭环中缓解推理费用压力。与此同时， 相比依赖海外闭源模型的在线调用，国产模型还更便于在本地进行部署与调用侧优化，使得推理成本与延迟进一步受控。

基于上述认知，构建端到端一体化元黑箱优化系统，不仅为大语言模型提供了“大模型做大事”的落地场景，也为优化算法设计与实际问题解决建立了更加紧密的协同机制。未来研究可围绕多模态问

题表示、结构化特征建模、跨层级生成能力以及高效推理机制等关键方向，结合当前已有的微调机制与知识增强方式，全面挖掘大语言模型在优化任务中的潜力，实现真正意义上的从问题到解决方案的端到端智能生成流程。

7.5 通用性进化的混合任务基座模型范式

尽管元黑箱优化已在多个任务上取得显著进展，但在通用性方面仍面临诸多挑战。一方面，现有研究往往聚焦于某一特定优化器的算法设计，缺乏跨优化器的适应能力。这种优化器依赖性限制了方法在其他演化策略（如粒子群优化、遗传算法等）上的适配能力，进而在优化器迁移与重用中引入性能不确定性。为提升方法的泛化能力，亟需构建更高层次的统一元黑箱优化框架，能够定义跨优化器的元任务并提供共享设计空间，从而支持多类优化器的模块化重组与策略迁移。部分研究已展示出通用性框架的可行性，例如，ConfigX [201] 和 DesignX[123] 构建了一个模块化的算法配置系统，在“算法结构 × 问题实例”的联合优化任务空间中进行多任务训练，从而学习出跨优化器通用的算法设计与配置策略。

另一方面，现有方法多聚焦于某一特定类型的问题场景（如无约束单目标优化），而现实中的优化问题则往往具有多样化与复杂化的特征，例如大规模优化、带约束问题、多模态问题等。若为每种问题类型分别训练独立智能体，不仅训练成本高昂，亦削弱了算法系统的可推广性。如何以低成本的进一步提升元黑箱优化方法的跨问题场景的迁移能力呢？一种更具前景的方式是构建混合任务基座模型[202]，在统一模型架构下实现对多类优化任务的端到端支持。为了实现这个目标，一方面我们可从架构上对现有的元黑箱优化拓展：将“元学习层—优化层”的双层范式推广为动态多层体系，如引入混合专家模型（Mixture of Experts，MoE），通过构建不同类别的“专家”，包含高维空间解耦专家、约束处理专家和多目标协同专家等，通过“路由器（router）”动态选择最合适的专家进行任务处理，提高算法设计的适应性和效率。另一方面，可对我们面对的任务—“黑箱”优化问题进一步外延以覆盖更多的优化任务：如允许有限结构与信息作为可选先验知识接入架构之中，将黑箱问题拓展为灰箱问题以更高效地求解。此外，在算法设计层面，如何平衡设计的自动化与可解释性呢？ 第 5.2节中的表 1 展示了优化器空间与决策空间中的两类方法具有显著的互补性，因此一个重要研究方向探讨是两者的有机融合。例如，可以在优化器空间搭建算法基础骨架，辅以决策器空间提供的优秀候选解加速学习过程，从而在保留灵活、透明的设计机制的同时，实现各指标的多方位平衡。

正如强化学习之父理查德·萨顿在《The Bitter Lesson》中提到的：“从苦涩的教训中学到的一件事是通用方法的巨大力量（One thing that should be learned from the bitter lesson is the great power of general purpose methods）”[203]。理想的元黑箱优化方法应具备在面对不同类型优化问题时，拥有能够自动设计出相应算法的能力，并在不断应用中持续提升自身的设计能力。

8 总结

在本综述中，我们系统回顾了近年来元黑箱优化领域的研究进展。作为黑箱优化与进化计算社区中新兴且重要的研究方向，元黑箱优化提出了一种前景广阔的范式转变：将传统的人工设计范式转变为基于智能体的自动化设计范式。依托双层优化框架，元黑箱优化中的神经网络元智能体能够通过元学习实现优化器的选择与配置，进而支持优化器的组装、模仿乃至生成。这一框架在提升设计效率的同时，有效统一了算法的泛化性与针对性，显著降低了对专家知识的依赖。

本综述首先明确了元黑箱优化的数学定义及其双层结构工作流程，随后从优化器空间与决策空间两个维度，系统分析了五类核心元任务：算法选择、算法配置、算法组装、算法生成与求解操作。在此基础上，我们从自动化程度和设计灵活度等角度对五类元任务进行了总结，并对其中使用的学习范式进行了系统地对比。此外，我们深入探讨了大语言

模型在元黑箱优化中的应用潜力与面临的挑战，指出其在实现全流程、端到端自动化算法设计中的关键价值。

最后，为推动元黑箱优化朝更高自动化水平、更强结构灵活性与更广泛通用性发展，我们结合已有成果，梳理了当前面临的核心瓶颈，并从问题理解、学习机制与模型架构等层面提出了多个值得深入探索的研究方向。我们期待这些研究的持续推进，能够加速元黑箱优化方法在应对日益复杂的现实优化问题中的理论突破与实际落地。

参 考 文 献

[1] LIU W, ZHANG Y, LIU K, et al. Evolutionary multi-objective optimisation for large-scale portfolio selection with both random and uncertain returns. IEEE Transactions on Evolutionary Computation, 2025, 29(1): 76-90.

[2] LENG K, LI S. Distribution path optimization for intelligent logistics vehicles of urban rail transportation using vrp optimization model. IEEE Transactions on Intelligent Transportation Systems, 2021, 23(2): 1661-1669.

[3] CHOUNG O H, VIANELLO R, SEGLER M, et al. Extracting medicinal chemistry intuition via preference machine learning. Nature Communications, 2023, 14 (1): 6651.

[4] LIU Y, WU L, YANG Y, et al. Secured reserve scheduling of pumped-storage hydropower plants in iso day-ahead market. IEEE Transactions on Power Systems, 2021, 36(6): 5722-5733.

[5] PASTOR G, HARTFIELD R, DOZIER G, et al. Statistically enhanced evolutionary algorithm for aerospace design applications//AIAA AVIATION 2022 Forum. Chicago, USA, 2022: 3789.

[6] KINGMA D P. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[7] BROYDEN C G. The convergence of a class of double-rank minimization algorithms 1. general considerations. IMA Journal of Applied Mathematics, 1970, 6(1): 76-90.

[8] MIRHOSEINI A, GOLDIE A, YAZGANM, et al. A graph placement methodology for fast chip design. Nature, 2021, 594(7862): 207-212.

[9] AKIBA T, SANO S, YANASE T, et al. Optuna: A next-generation hyperparameter optimization framework //Proceedings ofthe 25th ACM SIGKDD international conference on knowledge discovery & data mining. Anchorage, USA, 2019: 2623-2631.

[10] PAPPA G L, OCHOA G, HYDE M R, et al. Contrasting meta-learning and hyper-heuristic research: the role of evolutionary algorithms. Genetic Programming and Evolvable Machines, 2014, 15(1): 3-35.

[11] GONG Y J, LI J J, ZHOU Y, et al. Genetic learning particle swarm optimization. IEEE transactions on cybernetics, 2015, 46(10): 2277-2290.

[12] KAI-WEN L, TAO Z, RUI W, et al. Research reviews of com-binatorial optimization methods based on deep reinforcement learning. Acta Automatica Sinica, 2021, 47(11): 2521-2537 (in Chinese)
（李凯文, 张涛, 王锐, 等. 基于深度强化学习的组合优化研究进展. 自动化学报, 2021, 47(11): 2521-2537.）.

[13] WENJIE Y, TINGTING L, BEN N. Review of automated design of intelligent optimization algorithm. INFORMATION AND CONTROL, 2025, 54(2): 206-225 (in Chinese)
（蚁文洁, 刘婷婷, 牛奔. 智能优化算法自动设计综述. 信息与控制, 2025, 54(2): 206-225）.

[14] WEIJIE M, WEILI L, JINGHUI Z. Evolutionary many-task optimization framework based on machine learning . Chinese Journal of Computers, 2024, 47(1): 29-51 (in Chinese)
（麦伟杰, 刘伟莉, 钟竞辉. 基于机器学习的演化多任务优化框架. 计算机学报, 2024, 47(1): 29-51）.

[15] FANZHANG L, YANG L, PENGXIANG W, et al. A survey on recent advances in meta-learning. Chinese Journal of Computers, 2021, 44(2): 422-446 (in Chinese)
（李凡长, 刘洋, 吴鹏翔, 等. 元学习研究综述. 计算机学报, 2021, 44(2): 422-446）.

[16] MA Z, GUO H, GONG Y J, et al. Toward automated algorithm design: A survey and practical guide to

meta-black-box-optimization. IEEE Transactions on Evolutionary Computation, 2025.

[17] LIU F, TONG X, YUAN M, et al. Evolution ofheuristics: towards efficient automatic algorithm design using large language model//Proceedings of the 41st International Conference on Machine Learning. Vienna, Austria, 2024: 32201-32223.

[18] CHEN Q, DING W. A genetic algorithm based on deep q-learning in optimization of remote sensing data discretization. IEEE Transactions on Evolutionary Computation, 2024, 29(5): 2173-2187.

[19] NOVIKOV A, VŨ N, EISENBERGER M, et al. AlphaEvolve: a coding agent for scientific and algorithmic discovery. arXiv preprint arXiv:2506.13131, 2025.

[20] KERSCHKE P, HOOS H H, NEUMANN F, et al. Automated algorithm selection: Survey and perspectives . Evolutionary computation, 2019, 27(1): 3-45.

[21] TIAN Y, PENG S, ZHANG X, et al. A recommender system for metaheuristic algorithms for continuous optimization based on deep recurrent neural networks. IEEE transactions on artificial intelligence, 2020, 1(1): 5-18.

[22] GUO H, MA Y, MA Z, et al. Deep reinforcement learning for dynamic algorithm selection: A proof-of-principle study on differential evolution. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2024, 54(7): 4247-4259.

[23] TAHERNEZHAD-JAVAZM F, RANKIN D, BOIS N D, et al. R2 indicator and deep reinforcement learning enhanced adaptive multi-objective evolutionary algorithm . arXiv preprint arXiv: 2404.08161, 2024.

[24] ZHU N, ZHAO F, CAO J. A hyperheuristic and reinforcement learning guided meta-heuristic algorithm recommendation//2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD). Tianjin, China, 2024: 1061-1066.

[25] SCHEDE E, BRANDT J, TORNEDE A, et al. A survey ofmethods for automated algorithm configuration . Journal ofArtificial Intelligence Research, 2022, 75: 425-487.

[26] HUYNH T N, DO D T, LEE J. Q-learning-based parameter control in differential evolution for structural optimization. Applied Soft Computing, 2021, 107: 107464.

[27] XINYU Z, ZIYUE Y, WEIFENG G, et al. Adaptive multi-neighborhood artificial bee colony algorithm based on reinforcement learning. Chinese Journal of Computers, 2024, 47(7): 1521-1546 (in Chinese)

（周新宇, 尹子悦, 高卫峰, 等. 一种基于强化学习的自适应多邻域人工蜂群算法. 计算机学报, 2024, 47(7): 1521-1546）.

[28] PILLAYN, QU R. Automated design ofmachine learning and search algorithms. Cham, Switzerland: Springer International Publishing; 2021.

[29] QU R, KENDALL G, PILLAY N. The general combinatorial optimization problem: Towards automated algorithm design. IEEE Computational Intelligence Magazine, 2020, 15(2): 14-23.

[30] AHMADITESHNIZI A, GAO W, UDELL M. Optimus: scalable optimization modeling with (mi) lp solvers and large language models//Proceedings of the 41st International Conference on Machine Learning. Vienna, Austria, 2024: 577-596.

[31] CHEN Y, HOFFMAN M W, COLMENAREJO S G, et al. Learning to learn without gradient descent by gradient descent//International Conference on Machine Learning. Sydney, Australia, 2017: 748-756.

[32] SUN J, LIU X, BÄCK T, et al. Learning adaptive differential evolution algorithm from optimization experiences by policy gradient. IEEE Transactions on Evolutionary Computation, 2021, 25(4): 666-680.

[33] MA Z, CHEN J, GUO H, et al. Auto-configuring exploration-exploitation tradeoff in evolutionary computation via deep reinforcement learning// Proceedings of the Genetic and Evolutionary Computation Conference. Melbourne, Australia, 2024: 1497-1505.

[34] WANG J, ZHENG Y, ZHANG Z, et al. A novel multi-state reinforcement learning-based multi-objective evolutionary algorithm. Information Sciences, 2025, 688: 121397.

[35] RUOCHEN L, JIANXIA L, JING L, et al. A survey on dynamic multi-objective optimization. Chinese Journal of Computers, 2020, 43(7): 1246-1278 (in Chinese)
（刘若辰, 李建霞, 刘静, 等. 动态多目标优化研究综述. 计算机学报, 2020, 43(7): 1246-1278）.

[36] LIAN H, MA Z, GUO H, et al. Rlemmo: Evolutionary multimodal optimization assisted by deep reinforcement learning//Proceedings of the Genetic and Evolutionary Computation Conference. Melbourne, Australia, 2024: 683-693.

[37] MA Z, LIAN H, QIU W, et al. Accurate peak detection in multimodal optimization via approximated landscape learning//Proceedings of the Genetic and Evolutionary Computation Conference. Málaga, Spain, 2025: 1127 – 1136.

[38] WU S H, HUANG Y, WU X, et al. Learning to transfer for evolutionary multitasking. IEEE Transactions on Cybernetics, 2025, 55(7): 3342-3355.

[39] HUANG Y, LV X, WU S, et al. Advancing automated knowledge transfer in evolutionary multitasking via large language models. arXiv preprint arXiv: 2409.04270, 2024.

[40] LI X, WU K, BETTEREST LI Y, et al. Glhf: General learned evolutionary algorithm via hyper functions. arXiv preprint arXiv: 2405.03728, 2024.

[41] LANGE R, SCHAUL T, CHEN Y, LU C, ZAHAVY T, DALIBARD V, FLENNERHAG S. Discovering attention-based genetic algorithms via meta-black-box optimization //Proceedings of the Genetic and Evolutionary Computation Conference. Lisbon, Portugal, 2023: 929–937.

[42] GUO H, QIU W, MA Z, et al. Advancing cma-es with learning-based cooperative coevolution for scalable optimization. arXiv preprint arXiv: 2504.17578, 2025.

[43] WU X, WU S H, WU J, et al. Evolutionary computation in the era of large language model: Survey and roadmap. IEEE Transactions on Evolutionary Computation, 2024, 29(2): 534-554.

[44] CHAUHAN D, DUTTA B, BALA I, et al. Evolutionary computation and large language models: A survey of methods, synergies, and applications. arXiv preprint arXiv: 2505.15741, 2025.

[45] YU H, LIU J. Deep insights into automated optimization with large language models and evolutionary algorithms . arXiv preprint arXiv: 2410.20848, 2024.

[46] LIU F, YAO Y, GUO P, et al. A systematic survey on large language models for algorithm design. arXiv preprint arXiv: 2410.14716, 2024.

[47] WATKINS C J, DAYAN P. Q-learning. Machine learning, 1992, 8: 279-292.

[48] CENIKJ G, NIKOLIKJ A, PETELIN G, et al. A survey of meta-features used for automated selection of algorithms for black-box single-objective continuous optimization . arXiv preprint arXiv: 2406.06629, 2024.

[49] YONGWEI Z, LEI W. Metaheuristic algorithm selection system for continuous black-boxoptimization problems based on collaborative filterin. Control and Decision, 2020, 35(6): 1297-1306 (in Chinese)
（张永韡, 汪镭. 基于协同过滤的连续黑箱优化问题元启发算法选择. 控制与决策, 2020, 35(6): 1297-1306）.

[50] VAN RIJN J N, HUTTER F. Hyperparameter importance across datasets//Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. London, UK, 2018: 2367-2376.

[51] KOTTHOFF L. Algorithm selection for combinatorial search problems: a survey//DE RAEDT L, O'SULLIVAN B, editors. Data mining and constraint programming: foundations of a cross-disciplinary approach. Cham, Switzerland: Springer; 2016: 149-190.

[52] SMITH-MILES K A. Towards insightful algorithm selection for optimisation using meta-learning concepts //2008 IEEE international joint conference on neural networks. Hong Kong, China, 2008: 4118-4124.

[53] KANDA J Y, DE CARVALHO A C, HRUSCHKA E R, et al. Using meta-learning to recommend meta-heuristics for the traveling salesman problem//2011 10th international conference on machine learning and applications and workshops: Vol. 1. Honolulu, USA, 2011: 346-351.

[54] TIAN Y, PENG S, RODEMANN T, et al. Automated selection of evolutionary multi-objective optimization algorithms//2019 IEEE Symposium Series on Computational Intelligence (SSCI). Xiamen, China, 2019: 3225-3232.

[55] GUTIERREZ-RODRÍGUEZ A E, CONANT-PABLOS S E, ORTIZ-BAYLISS J C, et al. Selecting meta-heuristics for solving vehicle routing problems with time windows via meta-learning. Expert Systems with Applications, 2019, 118: 470-481.

[56] BISCHL B, MERSMANN O, TRAUTMANN H, et al. Algorithm selection based on exploratory landscape analysis and cost-sensitive learning//Proceedings of the 14th annual conference on Genetic and evolutionary computation. Philadelphia, USA, 2012: 313-320.

[57] MERSMANN O, BISCHL B, TRAUTMANN H, et al. Exploratory landscape analysis//Proceedings of the 13th annual conference on Genetic and evolutionary computation. Dublin, Ireland, 2011: 829-836.

[58] BOSER B E, GUYON I M, VAPNIK V N. A training algorithm for optimal margin classifiers//Proceedings of the fifth annual workshop on Computational learning theory. Pittsburgh, USA, 1992: 144-152.

[59] FIX E. Discriminatory analysis: nonparametric discrimination, consistency properties: Vol. 1. San Antonio, USA: USAF school of Aviation Medicine, 1985.

[60] RUMELHART D E, HINTON G E, WILLIAMS R J. Learning representations by back-propagating errors. nature, 1986, 323(6088): 533-536.

[61] HOCHREITER S, SCHMIDHUBER J. Long short-term memory. Neural computation, 1997, 9(8): 1735-1780.

[62] PUTERMAN M L. Markov decision processes //HEYMAN D P, SOBEL M J, editors. Handbooks in Operations Research and Management Science, Vol. 2. Amsterdam, Netherlands: North-Holland; 1990: 331–434.

[63] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning. nature, 2015, 518(7540): 529-533.

[64] VAN HASSELT H, GUEZ A, SILVER D. Deep reinforcement learning with double q-learning// Proceedings of the AAAI conference on artificial intelligence: Vol. 30. Palo Alto, USA, 2016.

[65] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal policy optimization algorithms. arXiv preprint arXiv: 1707.06347, 2017.

[66] DE CARVALHO V R, ÖZCAN E, SICHMAN J S. Comparative analysis of selection hyper-heuristics for real-world multi-objective optimization problems. Applied Sciences, 2021, 11(19): 9153.

[67] ZHAO F, GENG Z, ZHANG J, et al. A q-learning-based multi-objective hyper-heuristic algorithm with fuzzy policy decision technology. Expert Systems with Applications, 2025, 277: 127232.

[68] EGGENSPERGER K, MÜLLER P, MALLIK N, et al. Hpobench: A collection of reproducible multi-fidelity benchmark problems for hpo. arXiv preprint arXiv: 2109.06716, 2021.

[69] FIALHO Á. Adaptive operator selection for optimization . Université Paris Sud-Paris XI, 2010.

[70] ZHANG J, SANDERSON A C. Jade: adaptive differential evolution with optional external archive. IEEE Transactions on evolutionary computation, 2009, 13(5): 945-958.

[71] ZHAN Z H, ZHANG J, LI Y, et al. Adaptive particle swarm optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 2009, 39(6): 1362-1381.

[72] ČREPINŠEK M, LIU S H, MERNIK M. Exploration and exploitation in evolutionary algorithms: A survey. ACM computing surveys (CSUR), 2013, 45(3): 1-33.

[73] SUTTON R S. Learning to predict by the methods of temporal differences. Machine learning, 1988, 3(1): 9-44.

[74] XU Y, PI D. A reinforcement learning-based communication topology in particle swarm optimization . Neural Computing and Applications, 2020, 32(14): 10007-10032.

[75] HU Z, GONG W, LI S. Reinforcement learning-based differential evolution for parameters extraction of photovoltaic models. Energy Reports, 2021, 7: 916-928.

[76] KUKKONEN S, LAMPINEN J. A differential evolution algorithm for constrained multi-objective optimization: Initial assessment//Proceedings of the IASTED International Conference on Artificial Intelligence and Applications. Innsbruck, Austria, 2004: 96-102.

[77] HONG J, SHEN B, PAN A. A reinforcement learning-based neighborhood search operator for multi-modal optimization and its applications. Expert Systems with Applications, 2024, 246: 123150.

[78] SADHU A K, KONAR A, BHATTACHARJEE T, et al. Synergism of firefly algorithm and q-learning for robot arm path planning. Swarm and Evolutionary Computation, 2018, 43: 50-68.

[79] LIU Y, LU H, CHENG S, et al. An adaptive online parameter control algorithm for particle swarm optimization based on reinforcement learning//2019 IEEE congress on evolutionary computation (CEC). Wellington, New Zealand, 2019: 815-822.

[80] HAN Y, PENG H, MEI C, et al. Multi-strategy multi-objective differential evolutionary algorithm with reinforcement learning. Knowledge-Based Systems, 2023, 277: 110801.

[81] FISTER I, FISTER D, FISTER JR I. Reinforcement learning-based differential evolution for global optimization//Differential Evolution: From Theory to Practice. Cham, Switzerland: Springer, 2022: 43-75.

[82] BOLUFÉ-RÖHLER A, XU B. Deep reinforcement learning for smart restarts in exploration-only exploitation-only hybrid metaheuristics//Metaheuristics International Conference. Lorient, France, 2024: 19-34.

[83] XUE K, XU J, YUAN L, et al. Multi-agent dynamic algorithm configuration. Advances in Neural Information Processing Systems, 2022, 35: 20147-20161.

[84] HU Z, GONG W, PEDRYCZ W, et al. Deep reinforcement learning assisted co-evolutionary differential evolution for constrained optimization. Swarm and Evolutionary Computation, 2023, 83: 101387.

[85] WILLIAMS R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 1992, 8(3): 229-256.

[86] MNIH V, BADIA A P, MIRZA M, et al. Asynchronous methods for deep reinforcement learning//International conference on machine learning. New York, USA, 2016: 1928- 1937.

[87] LIU X, SUN J, ZHANG Q, et al. Learning to learn evolutionary algorithm: A learnable differential evolution. IEEE Transactions on Emerging Topics in Computational Intelligence, 2023, 7(6): 1605-1620.

[88] SHALA G, BIEDENKAPP A, AWAD N, et al. Learning step-size adaptation in cma-es//Parallel Problem Solving from Nature–PPSN XVI: 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5-9, 2020, Proceedings, Part I 16. Springer, 2020: 691-706.

[89] YIN S, LIU Y, GONG G, et al. Rlepso: Reinforcement learning based ensemble particle swarm optimizer// Proceedings of the 2021 4th International Conference on Algorithms, Computing and Artificial Intelligence. Sanya, China, 2021: 1-6.

[90] WU D, WANG G G. Employing reinforcement learning to enhance particle swarm optimization methods . Engineering Optimization, 2022, 54(2): 329-348.

[91] YIN S, JIN M, LU H, et al. Reinforcement-learning-based parameter adaptation method for particle swarm optimization. Complex & Intelligent Systems, 2023, 9 (5): 5585-5609.

[92] WHITLEY D, STARKWEATHER T, BOGART C. Genetic algorithms and neural networks: Optimizing connections and connectivity. Parallel computing, 1990, 14(3): 347-361.

[93] SALIMANS T, HO J, CHEN X, et al. Evolution strategies as a scalable alternative to reinforcement learning. arXiv preprint arXiv: 1703.03864, 2017.

[94] LANGE R T, SCHAUL T, CHEN Y, et al. Discovering evolution strategies via meta-black-box optimization// The Eleventh International Conference on Learning Representations. Kigali, Rwanda, 2023.

[95] FALDOR M, LANGE R T, CULLY A. Discovering quality-diversity algorithms via meta-black-box optimization. arXiv preprint arXiv: 2502.02190, 2025.

[96] CHEN M, FENG C, CHENG R. Metade: Evolving differential evolution by differential evolution. IEEE Transactions on Evolutionary Computation, 2025.

[97] DAS S, SUGANTHAN P N. Differential evolution: A survey of the state-of-the-art. IEEE transactions on evolutionary computation, 2010, 15(1): 4-31.

[98] SALLAM K M, ELSAYED S M, CHAKRABORTTY R K, et al. Evolutionary framework with reinforcement learning-based mutation adaptation. IEEE Access, 2020, 8: 194045-194071.

[99] PENG L, YUAN Z, DAI G, et al. Reinforcement learning-based hybrid differential evolution for global optimization ofinterplanetary trajectory design. Swarm and Evolutionary Computation, 2023, 81: 101351.

[100] SAMMA H, LIM C P, SALEH J M. A new reinforcement learning-based memetic particle swarm optimizer. Applied Soft Computing, 2016, 43: 276-297.

[101] LI Z, SHI L, YUE C, et al. Differential evolution based on reinforcement learning with fitness ranking for solving multimodal multiobjective problems. Swarm and Evolutionary Computation, 2019, 49: 234-244.

[102] XIA H, LI C, ZENG S, et al. A reinforcement-learning-based evolutionary algorithm using solution space clustering for multimodal optimization problems //2021 IEEE Congress on Evolutionary Computation (CEC). Kraków, Poland, 2021: 1938-1945.

[103] HU Z, GONG W. Constrained evolutionary optimization based on reinforcement learning using the objective function and constraints. Knowledge-Based Systems, 2022, 237: 107731.

[104] SHARMA M, KOMNINOSA, LÓPEZ-IBÁÑEZ M, et al. Deep reinforcement learning based parameter control in differential evolution//Proceedings of the genetic and evolutionary computation conference. Prague, Czech Republic, 2019: 709-717.

[105] TAN Z, LI K. Differential evolution with mixed mutation strategy based on deep reinforcement learning. Applied Soft Computing, 2021, 111: 107678.

[106] TIAN Y, LI X, MA H, et al. Deep reinforcement learning based adaptive operator selection for evolutionary multi-objective optimization. IEEE Transactions on Emerging Topics in Computational Intelligence, 2022, 7(4): 1051-1064.

[107] ZHANG Q, LI H. Moea/d: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on evolutionary computation, 2007, 11(6): 712-731.

[108] LIAO Z, PANG Q, GU Q. Differential evolution based on strategy adaptation and deep reinforcement learning for multimodal optimization problems. Swarm and Evolutionary Computation, 2024, 87: 101568.

[109] PEI J, LIU J, MEI Y. Learning from offline and online experiences: A hybrid adaptive operator selection framework//Proceedings of the Genetic and Evolutionary Computation Conference. Melbourne, Australia, 2024: 1017-1025.

[110] GUO H, MA S, HUANG Z, et al. Reinforcement learning-based self-adaptive differential evolution through automated landscape feature learning. arXiv preprint arXiv: 2503.18061, 2025.

[111] LI T, MENG Y, TANG L. Scheduling of continuous annealing with a multi-objective differential evolution algorithm based on deep reinforcement learning. IEEE Transactions on Automation Science and Engineering, 2023, 21(2): 1767-1780.

[112] SONG Y, WEI L, YANG Q, et al. Rl-ga: A reinforcement learning-based genetic algorithm for electromagnetic detection satellite scheduling problem. Swarm and Evolutionary Computation, 2023, 77: 101236.

[113] CHEN X, FANG S, LI K. Reinforcement-learning-based multi-objective differential evolution algorithm for large-scale combined heat and power economic emission dispatch. Energies, 2023, 16(9): 3753.

[114] ZITZLER E, KÜNZLI S. Indicator-based selection in multiobjective search//International conference on parallel problem solving from nature. Birmingham, UK, 2004: 832-842.

[115] LIEFOOGHE A, JOURDAN L, TALBI E G. A software framework based on a conceptual unified model for evolutionary multiobjective optimization: Paradiseo-moeo. European Journal ofOperational Research, 2011, 209(2): 104-112.

[116] BEZERRA L C, LÓPEZ-IBÁNEZ M, STÜTZLE T. Automatic component-wise design of multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation, 2015, 20(3): 403-417.

[117] KHUDABUKHSH A R, XU L, HOOS H H, et al. Satenstein: Automatically building local search sat solvers

from components. Artificial Intelligence, 2016, 232: 20-42.

[118] FRANZIN A, STÜTZLE T. Revisiting simulated annealing: A component-based analysis. Computers & operations research, 2019, 104: 191-206.

[119] PAGNOZZI F, STÜTZLE T. Automatic design of hybrid stochastic local search algorithms for permutation flowshop problems. European journal of operational research, 2019, 276(2): 409-421.

[120] DREO J, LIEFOOGHE A, VEREL S, et al. Paradiseo: from a modular framework for evolutionary computation to the automated design of metaheuristics: 22 years of paradiseo//Proceedings ofthe genetic and evolutionary computation conference companion. Melbourne, Australia, 2021: 1522-1530.

[121] MENG W, QU R. Automated design ofsearch algorithms: Learning on algorithmic components. Expert Systems with Applications, 2021, 185: 115493.

[122] YI W, QU R, JIAO L, et al. Automated design of metaheuristics using reinforcement learning within a novel general search framework. IEEE Transactions on Evolutionary Computation, 2022, 27(4): 1072-1084.

[123] GUO H, MA Z, MA Y, et al. Designx: Human-competitive algorithm designer for black-box optimization . arXiv preprint arXiv: 2505.17866, 2025.

[124] HANSEN N. The CMA evolution strategy: a comparing review. Towards a new evolutionary computation: Advances in the estimation ofdistribution algorithms，2016: 75- 102.

[125] TANABE R, FUKUNAGA A. Success-history based parameter adaptation for differential evolution//2013 IEEE congress on evolutionary computation. 2013: 71-78.

[126] STANOVOV V, AKHMEDOVA S, SEMENKIN E. Nl-shade-lbc algorithm with linear parameter adaptation bias change for cec 2022 numerical optimization//2022 IEEE Congress on Evolutionary Computation. Cancún, Mexico, 2022: 01-08.

[127] ZHAO Q, LIU T, YAN B, et al. Automated metaheuristic algorithm design with autoregressive learning. IEEE Transactions on Evolutionary Computation, 2024, 29(5): 2004-2018.

[128] ZHAO Q, YAN B, HU T, et al. Autoopt: A general framework for automatically designing metaheuristic optimization algorithms with diverse structures. IEEE Transactions on Emerging Topics in Computational Intelligence, 2025, 9(5): 3690-3703.

[129] CHEN X, BAI R, QU R, et al. Deep reinforcement learning assisted genetic programming ensemble hyper-heuristics for dynamic scheduling of container port trucks. IEEE Transactions on Evolutionary Computation, 2025, 29(4): 1371-1385.

[130] ZAKI T, ZEITRÄG Y, NEVES R, et al. A cooperative coevolutionary genetic programming hyper-heuristic for multi-objective makespan and cost optimization in cloud workflow scheduling. Computers & Operations Research, 2024, 172: 106805.

[131] SITAHONG A, YUAN Y, LI M, et al. Learning dispatching rules via novel genetic programming with feature selection in energy-aware dynamic job-shop scheduling. Scientific Reports, 2023, 13(1): 8558.

[132] CHEN J, MA Z, GUO H, et al. Symbol: Generating flexible black-box optimizers through symbolic equation learning. arXiv preprint arXiv: 2402.02355, 2024.

[133] MA Z, GUO H, CHEN J, et al. Llamoco: Instruction tuning of large language models for optimization code generation. arXiv preprint arXiv: 2403.01131, 2024.

[134] LV J, CUI C, ZHANG S, et al. Automated heuristic design for unit commitment using large language models . arXiv preprint arXiv: 2506.12495, 2025.

[135] HUANG J, LI X, GAO L, et al. Automatic programming via large language models with population self-evolution for dynamic job shop scheduling problem. arXiv preprint arXiv: 2410.22657, 2024.

[136] TV V, MALHOTRA P, NARWARIYA J, et al. Meta-learning for black-box optimization//Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Würzburg, Germany, 2019: 366-381.

[137] GOMES H S, LÉGER B, GAGNÉ C. Meta learning black-box population-based optimizers. arXiv preprint arXiv: 2103.03526, 2021.

[138] CHAYBOUTI S, DOS SANTOS L, MALHERBE C, et al. Meta-learning of black-box solvers using deep reinforcement learning//NeurIPS 2022, MetaLearn Workshop. New Orleans, USA, 2022.

[139] SONG L, GAO C, XUE K, et al. Reinforced in-context black-box optimization. arXiv preprint arXiv: 2402.17423, 2024.

[140] LI X, WU K, ZHANG X, et al. B2opt: Learning to optimize black-box optimization with little budget //Proceedings of the AAAI Conference on Artificial Intelligence: Vol. 39. Philadelphia, Pennsylvania, 2025: 18502-18510.

[141] YUN T, LEE K, YUN S, et al. An offline meta black-box optimization framework for adaptive design of urban traffic light management systems//Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. Barcelona, Spain, 2024: 6202-6213.

[142] YU Y, TANG Q, JIANG Q, et al. A deep reinforcement learning-assisted multimodal multi-objective bi-level optimization method for multi-robot task allocation. IEEE Transactions on Evolutionary Computation, 2025, 29(3): 574-588.

[143] MA Z, GUO H, CHEN J, et al. Metabox: A benchmark platform for meta-black-box optimization with reinforcement learning. Advances in Neural Information Processing Systems, 2023, 36: 10775-10795.

[144] MA Z, GONG Y J, GUO H, et al. Metabox-v2: A unified benchmark platform for meta-black-box optimization. arXiv preprint arXiv: 2505.17745, 2025.

[145] YANG C, WANG X, LU Y, et al. Large language models as optimizers//The Twelfth International Conference on Learning Representations. Kigali, Rwanda, 2023.

[146] LIU S, CHEN C, QU X, et al. Large language models as evolutionary optimizers//2024 IEEE Congress on Evolutionary Computation (CEC). Yokohama, Japan, 2024: 1-8.

[147] AHN J, VERMA R, LOU R, et al. Large language models for mathematical reasoning: Progresses and challenges . arXiv preprint arXiv: 2402.00157, 2024.

[148] JIANG J, WANG F, SHEN J, et al. A survey on large language models for code generation. arXiv preprint arXiv: 2406.00515, 2024.

[149] WANG H, FU T, DU Y, et al. Scientific discovery in the age of artificial intelligence. Nature, 2023, 620(7972): 47-60.

[150] JUMPER J, EVANS R, PRITZELA, et al. Highly accurate protein structure prediction with alphafold. nature, 2021, 596(7873): 583-589.

[151] GUO P F, CHEN Y H, TSAI Y D, et al. Towards optimizing with large language models. arXiv preprint arXiv: 2310.05204, 2023.

[152] HUANG B, WU X, ZHOU Y, et al. Exploring the true potential: Evaluating the black-box optimization capability of large language models. arXiv preprint arXiv: 2404.06290, 2024.

[153] MEYERSON E, NELSON M J, BRADLEY H, et al. Language model crossover: Variation through few-shot prompting. ACM Transactions on Evolutionary Learning, 2024, 4(4): 1-40.

[154] LANGE R, TIAN Y, TANG Y. Large language models as evolution strategies//Proceedings ofthe Genetic and Evolutionary Computation Conference Companion. Melbourne, Australia, 2024: 579-582.

[155] LIU F, LIN X, YAO S, et al. Large language model for multiobjective evolutionary optimization// International Conference on Evolutionary Multi-Criterion Optimization. Springer, Canberra, Australia 2025: 178-191.

[156] ZHU Y, LI J, LI G, et al. Hot or cold? adaptive temperature sampling for code generation with large language models //Proceedings of the AAAI Conference on Artificial Intelligence: Vol. 38. Vancouver, Canada, 2024: 437-445.

[157] BRAHMACHARY S, JOSHI SM, PANDA A, et al. Large language model-based evolutionary optimizer: Reasoning with elitism. Neurocomputing, 2025, 622: 129272.

[158] KOZA J R. Genetic programming as a means for programming computers by natural selection. Statistics and computing, 1994, 4: 87-112.

[159] ROMERA-PAREDES B, BAREKATAIN M, NOVIKOV A, et al. Mathematical discoveries from program search with large language models. Nature, 2024, 625(7995): 468-475.

[160] MA Y J, LIANG W, WANG G, et al. Eureka: Human-level reward design via coding large language models. arXiv preprint arXiv: 2310.12931, 2023.

[161] LEHMAN J, GORDON J, JAIN S, et al. Evolution through large models//Handbook of evolutionary machine learning. Singapore: Springer, 2023: 331-366.

[162] FU Y, ZHANG Y, YU Z, et al. Gpt4aigchip: Towards next-generation ai accelerator design automation via large language models//2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD). San Francisco, USA, 2023: 1-9.

[163] LIU F, TONG X, YUAN M, et al. Algorithm evolution using large language model. arXiv preprint arXiv: 2311.15249, 2023.

[164] YE H, WANG J, CAO Z, et al. Reevo: Large language models as hyper-heuristics with reflective evolution //The Thirty-eighth Annual Conference on Neural Information Processing Systems. Vancouver, Canada, 2024.

[165] VAN STEIN N, BÄCK T. Llamea: A large language model evolutionary algorithm for automatically generating metaheuristics. IEEE Transactions on Evolutionary Computation, 2024, 29(2): 331-345.

[166] HUANG Y, WU S, ZHANG W, et al. Autonomous multi-objective optimization using large language model . IEEE Transactions on Evolutionary Computation, 2025.

[167] YAO S, LIU F, LIN X, et al. Multi-objective evolution of heuristic using large language model//Proceedings of the AAAI Conference on Artificial Intelligence: Vol. 39. Philadelphia, Pennsylvania, 2025: 27144-27152.

[168] DAT P V T, DOAN L, BINH H T T. Hsevo: Elevating automatic heuristic design with diversity-driven harmony search and genetic algorithm using llms//Proceedings of the AAAI Conference on Artificial Intelligence: Vol. 39. Philadelphia, Pennsylvania, 2025: 26931-26938.

[169] YE H, XU H, YAN A, et al. Large language model-driven large neighborhood search for large-scale milp problems //Forty-second International Conference on Machine Learning. Vancouver, Canada, 2025.

[170] WU X, ZHONG Y, WU J, et al. Large language model-enhanced algorithm selection: Towards comprehensive algorithm representation//The 33rd International Joint Conference on Artificial Intelligence. Jeju, South Korea, 2024: 1-10.

[171] KRISTIADI A, STRIETH-KALTHOFF F, SKRETA M, et al. A sober look at llms for material discovery: Are they actually good for bayesian optimization over molecules? //International Conference on Machine Learning. PMLR, Vienna, Austria, 2024: 25603-25622.

[172] CUSTODE L L, CARAFFINI F, YAMAN A, et al. An investigation on the use of large language models for hyperparameter tuning in evolutionary algorithms// Proceedings ofthe Genetic and Evolutionary Computation Conference Companion. Melbourne, Australiam, 2024: 1838-1845.

[173] BROIDE L, STERN R. Evogpt: Enhancing test suite robustness via llm-based generation and genetic optimization. arXiv preprint arXiv: 2505.12424, 2025.

[174] MOHAMED A W, HADI A A, MOHAMED A K, et al. Problem definitions and evaluation criteria for the CEC 2021 special session and competition on single objective bound constrained numerical optimization. Melbourne, Australia: School of Computing Technologies, RMIT University; 2021: 1–20.

[175] KRAFT D. A software package for sequential quadratic programming. Cologne, Germany: Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt; 1988.

[176] JIANG C, SHU X, QIAN H, et al. Llmopt: Learning to define and solve general optimization problems from scratch. arXiv preprint arXiv: 2410.13213, 2024.

[177] VAN STEINN, V. KONONOVA A, KOTTHOFF L, et al. Code evolution graphs: Understanding large language model driven design of algorithms//Proceedings ofthe Genetic and Evolutionary Computation Conference. Málaga, Spain, 2025: 943-951.

[178] OMIDVAR M N, LI X, TANG K. Designing benchmark problems for large-scale continuous optimization. Information Sciences, 2015, 316: 419-436.

[179] LEWIS P, PEREZ E, PIKTUS A, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks . Advances in neural information processing systems, 2020, 33: 9459-9474.

[180] BALTRUŠAITIS T, AHUJA C, MORENCY L P. Multimodal machine learning: A survey and taxonomy . IEEE transactions on pattern analysis and machine intelligence, 2018, 41(2): 423-443.

[181] WRIGHT S. The roles of mutation, inbreeding, crossbreeding, and selection in evolution //Proceedings of the Sixth International Congress of Genetics. New York, USA, 1932: 356–366.

[182] SEILER M V, KERSCHKE P, TRAUTMANN H. Deep-ela: Deep exploratory landscape analysis with self-supervised pretrained transformers for single-and multi-objective continuous optimization problems. Evolutionary Computation, 2025, 33(4): 513-540.

[183] MA Z, CHEN J, GUO H, et al. Neural exploratory landscape analysis for meta-black-box-optimization// The Thirteenth International Conference on Learning Representations. Singapore, 2025.

[184] OMIDVAR M N, LI X, YAO X. A review of population-based metaheuristics for large-scale black-box global optimization—part i. IEEE Transactions on Evolutionary Computation, 2021, 26(5): 802-822.

[185] QIU W, GUO H, MA Z, et al. A novel two-phase cooperative co-evolution framework for large-scale global optimization with complex overlapping. arXiv preprint arXiv: 2503.21797, 2025.

[186] TAN C, SUN F, KONG T, et al. A survey on deep transfer learning//Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, 2018: 270-279.

[187] CARUNA R. Multitask learning: A knowledge-based source of inductive bias//Machine learning: Proceedings of the tenth international conference. Amherst, USA, 1993: 41-48.

[188] KIRKPATRICK J, PASCANU R, RABINOWITZ N, et al. Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences, 2017, 114(13): 3521-3526.

[189] MA Z, CAO Z, JIANG Z, et al. Meta-black-box-optimization through offline q-function learning //Forty-second International Conference on Machine Learning. Vienna, Austria, 2025.

[190] PEI J, MEI Y, LIU J, et al. Libog: Lifelong learning for black-box optimizer generation. arXiv preprint arXiv: 2505.13025, 2025.

[191] WENG J, CHEN H, YAN D, et al. Tianshou: A highly modularized deep reinforcement learning library . Journal ofMachine Learning Research, 2022, 23(267): 1-6.

[192] MORITZ P, NISHIHARA R, WANG S, et al. Ray: A distributed framework for emerging {AI} applications //13th USENIX symposium on operating systems design and implementation (OSDI 18). Carlsbad, USA, 2018: 561-577.

[193] ZHAN Z H, ZHANG J, LIN Y, et al. Matrix-based evolutionary computation. IEEE Transactions on Emerging Topics in Computational Intelligence, 2021, 6 (2): 315-328.

[194] HUANG B, CHENG R, LI Z, et al. Evox: A distributed gpu-accelerated framework for scalable evolutionary computation. IEEE Transactions on Evolutionary Computation, 2024, 29(5): 1649-1662.

[195] SHEN L, MISHRA A, KHASHABI D. Do pretrained transformers learn in-context by gradient descent?. arXiv preprint arXiv: 2310.08540, 2023.

[196] PATTERSON D, GONZALEZ J, LE Q, et al. Carbon emissions and large neural network training. arXiv preprint arXiv: 2104.10350, 2021.

[197] BUBECK S, CHADRASEKARAN V, ELDAN R, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. arXiv preprint arXiv:2303.12712, 2023.

[198] LIU A, FENG B, XUE B, et al. Deepseek-v3 technical report. arXiv preprint arXiv: 2412.19437, 2024.

[199] BAI J, BAI S, CHU Y, et al. Qwen technical report. arXiv preprint arXiv: 2309.16609, 2023.

[200] TEAM K, BAI Y, BAO Y, et al. Kimi k2: Open agentic intelligence. arXiv preprint arXiv: 2507.20534, 2025.

[201] GUO H, MA Z, CHEN J, et al. Configx: Modular configuration for evolutionary algorithms via multitask reinforcement learning//Proceedings of the AAAI Conference on Artificial Intelligence: Vol. 39. Philadelphia, Pennsylvania, 2025: 26982-26990.

[202] BOMMASANI R, HUDSON DA, ADELIE, et al. On the opportunities and risks of foundation models. arXiv preprint arXiv: 2108.07258, 2021.

[203] SUTTON R. The bitter lesson. Incomplete Ideas (blog), 2019, 13(1): 38.

QIU Wen-Jie, Ph.D. candidate. His research interests include meta-black box optimization, evolutionary computation.

GUO Hong-Shu, Ph.D. candidate. His research interests include evolutionary computation, meta-black box optimization, reinforcement learning.

MA Ze-Yuan, Ph.D. candidate. His research interests include evolutionary computation, meta-black-box optimization, reinforcement learning.

ZHANG Jun, Ph.D., professor. His research interests include artificial intelligence, computational intelligence, evolutionary computation, swarm intelligence.

GONG Yue-Jiao, Ph.D., professor. Her research interests include meta-black-box optimization, evolutionary computation.