软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2024,35(4):1732−1750 [doi: 10.13328/j.cnki.jos.007017]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于动态批量评估的绿色无梯度优化方法 *

钱 鸿[1,2], 舒 翔[2], 孙天祥[3], 邱锡鹏[3], 周爱民[1,2]

[1](华东师范大学 上海智能教育研究院, 上海 200062)

[2](华东师范大学 计算机科学与技术学院, 上海 200062)

[3](复旦大学 计算机科学技术学院, 上海 200438)

通信作者: 周爱民, E-mail: amzhou@cs.ecnu.edu.cn

摘 要: 在基于语言模型即服务的提示词黑盒微调、机器学习模型超参数调节等优化任务中, 由于解空间到性能指标之间的映射关系复杂多变, 难以显式地构建目标函数, 故常采用无梯度优化方法来实现寻优. 解的准确、稳定评估是有效实施无梯度优化方法的关键, 完成一次解的质量评估常要求在整个数据集上完整运行一次模型, 且优化过程有时需要大量评估解的质量. 随着机器学习模型复杂度以及训练样本量的不断增加, 准确、稳定的解的质量评估时间成本与计算代价越来越高昂, 这与绿色低碳机器学习与优化理念背道而驰. 有鉴于此, 提出了一种基于动态批量评估的绿色无梯度优化方法框架(green derivative-free optimization with dynamic batch evaluation, GRACE), 基于训练子集的相似性, 在优化过程中自适应动态调节评估解时使用的样本量, 使得 GRACE 在保证优化性能的同时, 降低优化成本与代价, 达到绿色低碳高效的目标. 在语言模型即服务提示词黑盒微调、模型超参数优化等实际任务上进行了实验验证, 通过与一系列对比方法以及 GRACE 消融退化版算法进行比较分析, 表明了 GRACE 的有效性、高效性、绿色低碳性. 超参数分析结果表明了其具备超参数稳健性.

关键词: 无梯度优化; 演化学习; 绿色低碳; 动态批量评估

中图法分类号: TP18

中文引用格式: 钱鸿, 舒翔, 孙天祥, 邱锡鹏, 周爱民. 基于动态批量评估的绿色无梯度优化方法. 软件学报, 2024, 35(4): 1732−1750. http://www.jos.org.cn/1000-9825/7017.htm

英文引用格式: Qian H, Shu X, Sun TX, Qiu XP, Zhou AM. Green Derivative-free Optimization Method with Dynamic Batch Evaluation. Ruan Jian Xue Bao/Journal of Software, 2024, 35(4): 1732−1750 (in Chinese). http://www.jos.org.cn/1000-9825/7017.htm

Green Derivative-free Optimization Method with Dynamic Batch Evaluation

QIAN Hong[1,2], SHU Xiang[2], SUN Tian-Xiang[3], QIU Xi-Peng[3], ZHOU Ai-Min[1,2]

[1](Shanghai Institute of AI Education, East China Normal University, Shanghai 200062, China)

[2](School of Computer Science and Technology, East China Normal University, Shanghai 200062, China)

[3](School of Computer Science, Fudan University, Shanghai 200438, China)

Abstract: Derivative-free optimization is commonly employed in tasks such as black-box tuning of language-model-as-a-service and hyper-parameter tuning of machine learning models, where the mapping between the solution space of the optimization task and the performance indicator is intricate and complex, making it challenging to explicitly formulate an objective function. Accurate and stable evaluation of solutions is crucial for derivative-free optimization methods. The evaluation of the quality of a solution often requires running the model on the entire dataset, and the optimization process sometimes requires a large number of evaluations of solution quality.

* 基金项目: 国家自然科学基金(62106076); 上海市“科技创新行动计划”人工智能科技支撑专项(22511105901); CCF-蚂蚁科研基金(CCF-AFSG RF20220205); 上海市自然科学基金(21ZR1420300)

本文由“绿色低碳机器学习研究与应用”专题特约编辑封举富教授、俞扬教授、刘淇教授推荐.

收稿时间: 2023-05-15; 修改时间: 2023-07-07; 采用时间: 2023-08-24; jos 在线出版时间: 2023-09-11

CNKI 网络首发时间: 2023-11-24

The growing complexity of machine learning models and the expanding size of training datasets result in escalating time and computational costs for accurate and stable solution evaluation, contradicting the principle of green and low-carbon machine learning and optimization. In view of this, this study proposes a green derivative-free optimization framework with dynamic batch evaluation (GRACE). Based on the similarity of training subsets, GRACE adaptively and dynamically adjusts the sample size used for evaluating solutions during the optimization process, thereby ensuring optimization performance while reducing optimization costs and computational expenses, achieving the goal of green, low-carbon, and efficient optimization. Experiments are conducted on tasks such as black-box tuning of language-model-as-a-service and hyper-parameter optimization of models. By comparing with the comparative methods and the degraded versions of GRACE, the effectiveness, efficiency, and green and low-carbon merits of GRACE are verified. The results also show the hyper-parameter robustness of GRACE.

Key words: derivative-free optimization; evolutionary learning; green and low-carbon; dynamic batch evaluation

机器学习模型不断发展, 为智能系统应用开辟了新的可能性[1,2], 这些模型在各个领域均取得了较好的性能, 包括图像理解[3−5]、语音识别[6,7]、自然语言处理[8−11]、强化学习[12−14]等. 随着机器学习的快速发展, 模型的参数量正在急剧上升, 训练模型所需的数据集体量越来越大, 资源需求也不断增加. 例如近年来语言大模型的兴起, 自然语言处理(natural language processing, NLP)领域中的模型规模扩大, 复杂度显著增加[8,9,15]. 大模型具有成百上千亿参数量, 并且需要大量的计算资源进行训练和评估[15−17]. 虽然这些模型在各种 NLP 任务上取得了令人瞩目的性能, 但其模型训练与优化过程却极具挑战性并且计算成本高昂.

在此背景下, 语言模型即服务(language-model-as-a-service, LMaaS)等新范式被提出[15,16]. LMaaS 基于提示词微调(prompt fine-tune)的思想, 提出了一种新颖的解决方案, 即黑盒微调(black-box tuning, BBT)[13]. BBT 将预训练语言模型放在服务端, 并将其视作一种只提供前向推理 API 接口的服务, 通过在客户端调用接口, 在本地微调表征提示词(prompt)的特定参数[18,19], 即可在不更改大模型结构以及无需重新训练大模型的情况下, 绿色、有效地完成下游特定任务上的推理. BBT 以其良好的性能和绿色低能耗的特点, 已被广泛应用在问答、情感分析和文本分类[20]等 NLP 任务中. 在语言模型即服务[12,13]、模型超参数优化[21]等任务中, 因解空间到性能目标空间之间的映射关系复杂多变, 难以显式地构建出需要优化的目标函数. 例如, 在语言模型即服务任务中, 模型被部署在远程的服务端, 模型的评估结果仅可通过调用推理接口获得, 待优化的提示词参数与模型输出的性能指标之间的关系是难以用数学表达式明确刻画的. 此时, 常采用无梯度优化方法[22−24]完成优化, 即以采样与试错的方式来搜索参数空间, 进而找到最优的参数配置.

无梯度优化方法[22−24]通过在解空间(solution space)中采样解并对解的质量(即目标函数值)进行评估来实现优化, 其优化过程不依赖目标函数的梯度或者海森矩阵等信息. 这类优化方式的优点是具有以一定概率全局寻优的潜力, 可以处理不可微、不连续、非凸等复杂优化问题, 只要能够对解的质量进行评估即可使用该类优化方法, 因而具有一定的通用性[23]. 大多数无梯度优化方法具有相似的算法流程, 它们首先在解空间中随机采样解来初始化解集合, 在对采样到的解进行目标函数值评估后, 通过无梯度优化方法显式或隐式地建立关于潜在目标函数的代理模型或采样模型, 然后根据某种机制从该模型中采样新解加以评估, 这些已评估的解则会用来更新模型. 无梯度优化方法迭代采样与更新模型, 以期能够不断提升解的质量.

通过无梯度优化的过程可以看出, 解的准确、高效评估对无梯度优化算法至关重要. 完成一次解的质量评估常需在整个数据集上完整运行一次模型, 且优化过程有时要求大量评估解的质量. 在如今模型越来越复杂、训练数据集体量越来越庞大的背景下, 解的评估速度显著变慢、计算代价不断攀升, 代理模型或采样模型更新所需能耗迅速增长, 无梯度优化对解空间搜索的时间成本和计算资源需求日益增加. 尽管更大的训练集可以提升解的评估的准确性与稳定性, 但会导致更高的评估成本[25−27]. 因此, 如何在不损害无梯度优化算法性能的条件下, 设计算法自适应动态调整评估解所用的样本, 从而减小评估所用的样本量, 降低评估代价, 保持评估的准确与稳定性, 是当前亟需解决的关键问题.

为了降低评估代价、使无梯度优化绿色低碳, 一方面, 在解的评估时应该选择尽可能小的训练数据子集, 以缩短评估时间、降低评估能耗; 另一方面, 为了实现较好的优化性能, 需要使用尽可能大的训练数据子集进行评估, 并且要让优化算法见到足够多的训练数据, 达到准确且稳定的解的评估. 类似的困难也出

现在梯度优化方法中, 例如, 梯度下降通常用于学习神经网络的权重和偏置, 当训练集很大时, 由于每步梯度都需要在整个训练集上计算, 梯度下降的迭代成本高昂. 为此, 随机梯度下降(SGD)[28]和小批量梯度下降(MBGD)[29]等优化策略被提出. 相较于全量训练样本集上的梯度下降法, 这些优化策略可以在不明显影响解的质量的同时, 显著减少算法迭代时间与计算成本, 达到绿色高效的目的. 然而, 黑盒目标函数难以直接照搬梯度方法进行求解[30], 与梯度优化中使用的批量梯度下降算法不同, 无梯度优化方法中自适应动态调整评估所需训练数据子集大小的策略目前较为匮乏. 尽管已有静态地设定不同大小的训练数据子集作为不同评估保真度的优化方法[25,31], 以及将小规模训练子集上的评估建模成全量训练集上评估的带噪优化等方法[32,33], 但设计出能主动、自适应地动态选择训练样本子集以实现绿色有效动态批量评估的无梯度优化方法仍迫在眉睫.

鉴于此, 本文提出一种基于动态批量评估的绿色无梯度优化方法框架(green derivative-free optimization with dynamic batch evaluation, GRACE). 优化过程中, GRACE 会动态调整训练子集中的样本数, 并根据不同训练子集下的解的评估情况, 动态决定使用哪批样本构成训练子集用于当前解的评估, 进而在优化过程中只使用有代表性的少批量样本, 在较低的评估代价下实现较好的优化性能, 达到绿色低碳高效优化的目标. 具体而言, 为了充分利用解的评估信息, GRACE 构造了动态评估表这一特殊的数据结构, 以实时记录不同样本子集上解的评估值. 解的评估值既要返回给黑盒优化器, 又要用于评估不同样本子集之间的相似程度. 算法通过比较不同样本子集对同一个解的评估值之间的差异, 评估样本子集之间的关系, 进而构造批量相似树来表征不同样本子集之间的相似程度. 在优化过程中, 算法每隔一定的时间会动态更新批量相似树的结构, 每次优化前通过批量索引选择策略从批量相似树中选择出差异性较大、代表性较强的样本子集, 使得解的评估过程中使用的样本子集尽可能仅包括高度凝练的、具有代表性的少批量样本, 实现绿色低碳且高效准确的优化过程. 我们在语言模型即服务提示词黑盒微调(BBT)和模型超参数优化任务上开展了系列实验, 通过与多种对比方法及 GRACE 算法的消融版本进行比较分析, 实验结果展示了 GRACE 算法在运行效率、优化有效性以及绿色低碳性等方面的优势.

本文首先介绍相关工作, 回顾无梯度优化基础知识, 定义问题形式, 然后详细介绍本文提出的基于动态批量评估的绿色无梯度优化方法, 设计实验并分析结果, 最后总结全文.

1 相关工作

目前已有一些以静态方式选择样本子集用于解的评估的优化方法, 相关研究通过将不同大小的训练子集建模为多保真度评估或带噪评估来处理该问题.

- 多保真度优化

当优化过程中存在多个不同准确度和成本的评估器时, 基于不同准确度的评估器的优化问题称为多保真度优化问题[25−27,34−39], 可采用多保真度优化方法来权衡评估准确性和评估成本之间的矛盾(高保真度评估准确但费时费力、低保真度评估不准确但省时省力). 例如, BOCA 算法[34]是一种典型的多保真度优化方法, 其能够有效地共享不同保真度的评估信息, 进而有针对性地选择评估来源以获得更好的优化性能. 其他多保真度优化方法, 例如使用 Hyperband 调度算法选择保真度评估源的 BOHB 算法[35]、使用 Successive Halving 资源分配算法确定保真度评估源的 ASHA 算法[36]、基于神经网络[34,37]或熵搜索[38,39]的方法等, 均在超参数优化等任务上展现出较好的结果. 多保真度优化可将在不同大小的训练子集上的解的评估作为不同保真度的评估源, 通过对低潜力解使用低保真度评估器、高潜力解使用高保真度评估器来权衡优化的效率与性能, 但现有方法往往难以在优化过程中动态自适应调整用于当前解的评估的训练子集的大小.

- 噪声优化

对于解的评估, 在全量训练集的不同子集上的评估结果往往不同, 此时可将这种评估偏差建模成带噪评估, 采用噪声优化方法进行处理[32,33,40]. 噪声优化方法通过将不同训练子集评估结果之间的差异建模为带噪的目标函数来处理训练子集评估不精确的问题. 在无梯度优化中常用的噪声处理方法有: 通过多次采样取平

均值来降低噪声影响[41]、通过阈值选择来处理噪声[32]、通过值抑制方法(如 SSRACOS 算法[42])控制受噪声影响的评估值等. 这些方法均可用来处理使用训练子集评估造成的不精确问题, 例如, 可以采用多次评估取平均值的降噪方法, 即在多个不同的训练子集上评估后, 将其评估的平均值作为评估值, 这样可减少不同训练子集之间的差异性对优化性能的影响, 降低了评估的不确定性. 但此类方法也面临在优化过程中难以动态自适应调节用于当前解评估的训练子集大小的问题.

- 梯度优化中的批量策略

随着 SGD [28]和 MBGD 算法的问世, 诸多高效的基于样本子集的梯度优化算法被提出[29,40,43,44], 这些工作往往将全量数据集划分为若干个小的批量, 并在批量上进行代价较低的批量梯度优化, 通过设计批量策略实现高效优化. 例如: 随机平均梯度下降(SAG)算法[40]在训练期间记录每个批量上的梯度信息, 进而估计完整批量的平均梯度, 实现高效梯度优化; 随机方差缩减梯度下降(SVRG)[43]不需要长期存储每个批量梯度信息, 而是通过方差缩减技巧提高梯度估计质量以加速收敛, 达到更好的优化性能; SAGA 算法[44]在借鉴了 SAG 算法中梯度估计方法的同时, 消除了 SVRG 算法的复杂步骤, 兼具了两者的优势. 尽管基于小批量样本子集进行优化会在一定程度上带来解的收敛性问题, 但随着梯度优化方法的不断发展, 基于批量的梯度优化方法愈加完善, 目前已经能够准确高效地处理梯度优化问题. 与梯度优化方法形成对比的是, 无梯度优化中聚焦批量策略的工作匮乏, 当前鲜有针对批量优化的算法.

2 基础知识

对于变量$\boldsymbol{\theta}\in\Theta\subseteq\mathbb{R}^d$, 寻找黑盒函数 $f(\boldsymbol{\theta})$的最优值(最大值或最小值)的问题称为无梯度优化问题. 本文以最小化为例, 无梯度优化问题可被抽象为

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\Theta\subseteq\mathbb{R}^d} f(\boldsymbol{\theta}),$$

其中, $f(\boldsymbol{\theta})$是黑盒目标函数, d 维变量$\boldsymbol{\theta}\in\Theta$称为解, Θ称为解空间.

无梯度优化(又称黑盒优化、零阶优化)[22−24]是通过对解空间进行采样并评估解的质量(即目标函数的值)来完成优化任务的. 大多数无梯度优化方法的算法具有类似的框架, 如图 1 所示. 首先, 随机在解空间中采样解来初始化解集. 在评估样本解的目标函数值后, 无梯度优化方法显式或隐式地建立关于潜在目标函数的模型. 然后, 根据特定机制从模型中采样出待评估的新解. 最后, 将这些完成评估的解用于更新模型. 无梯度优化方法迭代进行采样与模型更新过程, 不断提高解的质量, 直到算法满足终止条件, 以期找到最优解.

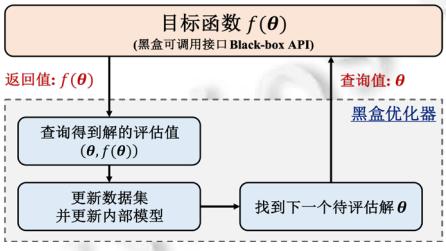


图 1 无梯度优化框架图

无梯度优化的代表性方法包括演化策略(evolutionary strategy, ES)[45−47]、贝叶斯优化(Bayesian optimization, BO)[48,49]、基于分支定界法的乐观优化(optimistic optimization)[50−53]、交叉熵方法(cross-entropy method)[54]等. 其中, 演化策略隶属于演化算法(evolutionary algorithm, EA), 是一种基于种群的优化方法, 无需目标函数的梯度信息即可搜索解空间进行寻优[46,55,56]. 在演化策略中, 搜索过程始于由一组候选解组成的初始种群. 种群中的每个解都沿着特定的轨迹探索解空间, 每次迭代称为种群的一次演化, 由此产生的解集称为种群的一代, 迭代过程中的种群会权衡探索与利用并趋于最优解. 演化策略具有在解空间中寻找全局最优解的潜力.

在演化策略过程中, 构成解集的个体解将经历算子操作, 例如变异、杂交和选择, 这些操作产生新的解进

而形成新一代的种群. 与自然进化类似, 由于变异和杂交产生的新个体并不总是适应环境的, 自然选择在这个过程中发挥了关键作用, 即: 一般只允许适应性强的个体(评估值较好的解)存活, 淘汰适应性较差的个体(评估值较差的解). 幸存的个体形成一个新的种群代(子代), 循环迭代上述过程, 直至收敛到满意的解.

相较于普通的演化策略, 协方差矩阵自适应演化策略(covariance matrix adaptation evolution strategy, CMA-ES)[45]是一种考虑解空间各维度相关性的高效演化策略, 具有更快的收敛速度和更好的收敛效果. 演化策略通常使用高斯分布建模采样模型, 从由均值向量和协方差矩阵表示的多元高斯分布中采样, 进而生成新的候选解种群. 普通的演化策略常通过将随机变异向量与均值向量相加, 并使用固定不变的协方差矩阵来控制搜索种群的分布. CMA-ES 算法对这一过程进行改进, 使用随迭代自适应变化的协方差矩阵控制搜索分布, 通过动态的协方差矩阵表示搜索空间中不同变量之间的相关性, 使优化算法能够更快收敛找到优质解. 此外, CMA-ES 算法在保留种群高质量候选解的同时, 还会生成特征丰富多样的候选解, 进而有效地权衡对解空间的探索与对已有解信息的利用. CMA-ES 算法还具有处理高维优化和噪声优化问题的潜力. 因此, 在本文实验部分, 我们选用 CMA-ES 算法作为 GRACE 框架中无梯度优化的具体实施算法.

3 问题定义

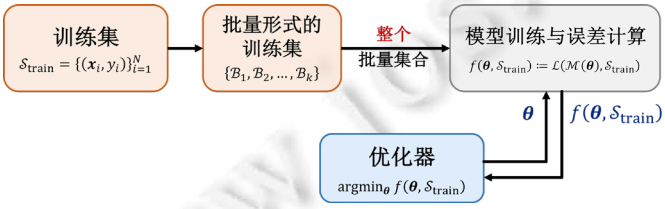
3.1 使用全量样本进行解评估的优化

在介绍本文提出的算法框架之前, 首先对要解决的问题进行形式化描述. 基于语言模型即服务的提示词黑盒微调[15,16]、模型超参数优化等任务可抽象为如下优化问题:

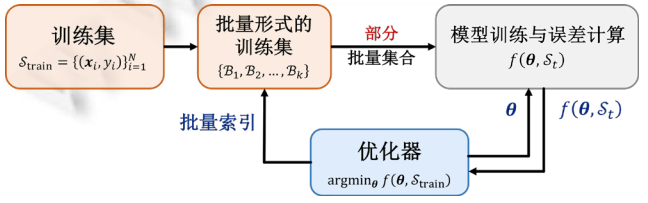
$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\Theta\subseteq\mathbb{R}^d} \mathcal{L}(\mathcal{M}(\boldsymbol{\theta}),\mathcal{S}_{train}) = \arg\min_{\boldsymbol{\theta}\in\Theta\subseteq\mathbb{R}^d} f(\boldsymbol{\theta},\mathcal{S}_{train}),$$

其中, $\mathcal{L}(\cdot)$为模型训练过程中使用的损失函数, $\mathcal{L}(\mathcal{M}(\boldsymbol{\theta}),\mathcal{S}_{train})$是解$\boldsymbol{\theta}$配置下的模型$\mathcal{M}(\boldsymbol{\theta})$在全量训练集$\mathcal{S}_{train}=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$上的损失. 简便起见, 可以将上述损失写成$f(\boldsymbol{\theta},\mathcal{S}_{train})$的形式, $f(\boldsymbol{\theta},\mathcal{S}_{train})$即是优化问题的目标函数, 优化的目标为找到能够在全量训练集$\mathcal{S}_{train}$上表现最好的解$\boldsymbol{\theta}^*$.

为了快速方便地进行评估, 通常将整个训练集划分为若干个相同大小的被称为批量(batch)的训练子集进行评估, 即: 首先, 将整个训练数据集$\mathcal{S}_{train}$按照特定批量大小 B(即每个批量的元素数量为 B)分成 K 个批量$\{\mathcal{B}_1,\mathcal{B}_2,\ldots,\mathcal{B}_K\}$, 其中, K 是满足 $K\times B$ 小于全量训练集$\mathcal{S}_{train}$元素数量的最大整数; 然后, 依次使用每个批量来评估解, 得到 K 个评估值; 最后, 将每个批量评估结果的平均值(通常是指模型训练过程中的损失或准确率)反馈给优化器, 作为当前参数下模型在整个训练数据集上的评估结果. 使用批量进行模型训练和参数优化的过程如图 2(a)所示.



(a) 使用全量样本进行解评估的优化



(b) 使用部分批量样本进行解评估的优化

图 2 全量样本进行解评估的无梯度优化 vs.部分批量样本进行解评估的无梯度优化

这种分割数据集的操作没有改变评估结果, 且可以充分利用计算机硬件的并行计算能力以加速参数优化的过程. 由于其高效和快速的特性, 已经被诸多大模型参数训练任务所采用.

3.2 使用部分批量样本进行解评估的优化

尽管将训练集分成几个批量进行评估可以更好地利用并行计算的优势, 但由于每次评估还是需要评估所有的训练集样本, 评估代价仍然很高. 相反, 若考虑使用训练集中的一小部分数据作为训练数据子集对全量数据的训练效果进行近似, 则可以大幅降低评估成本. 但使用数据子集的评估值和全量训练集评估值之间存在误差, 即:

$$f(\boldsymbol{\theta},\mathcal{S}_{train})=f(\boldsymbol{\theta},\mathcal{S}_t)+Noise,$$

其中, $f(\boldsymbol{\theta},\mathcal{S}_t)$是在第 t 次评估时, 仅在包含部分数据的数据集$\mathcal{S}_{train}$上进行评估的结果.

在仅使用样本子集进行优化时, 可以考虑基于批量进行样本选择, 即: 在将全量数据集划分成若干大小相等的批量后, 每次仅选择少量的批量进行评估. 此时, 只需用这些批量构成数据子集$\mathcal{S}_{train}$进行评估, 而不是消耗大量资源来遍历所有批量. 相较于从样本中直接选择出一部分数据, 这种选择批量的方法更加快捷方便, 且优化效果与前者类似. 这种选择批量作为样本子集的方法可以被描述为

$$\mathcal{S}_t=\bigcup\nolimits_{i\in\{i_1,i_2,...,i_{t_n}\}}\mathcal{B}_i,$$

其中, $i\in\{i_1,i_2,...,i_{t_n}\}$是第 t 次迭代中批量索引的集合, $\mathcal{B}_i$是所有批量中的第 i 个批量(即划分批量时, 编号为 i 的批量). 在这种思想的指导下, 数据子集的构建不需要精确到具体的数据, 而只需要对批量进行选择, 图 2(b)展示了这种方法的操作过程, 影响其性能的关键在于, 如何选取每次评估使用批量的索引 $i\in\{i_1,i_2,...,i_{t_n}\}$.

4 基于动态批量评估的绿色无梯度优化

4.1 GRACE框架

本文提出了一种基于动态批量评估的绿色无梯度优化方法框架(GRACE), 通过在优化过程中动态选择评估解使用的样本子集, 进而以较低的代价得到较高质量解的评估值, 有效权衡评估代价和优化性能, 实现绿色低碳且高效准确的优化效果. 具体而言, 算法以批量为单位进行评估, 并在优化过程中评估不同批量之间的相似程度, 指导算法选择差异性大、代表性强的样本批量进行评估, 实现以较低评估代价得到较高精度的解的评估的效果. 为了更好地构建批量之间的相似性关系, 本文还采用了一种特殊的线性表数据结构, 即动态评估表, 通过记录使用不同批量评估解的评估值, 指导批量关系树的构建. GRACE 框架全流程如算法 1 所示, GRACE 流程示意图如图 3 所示.

算法 1. 基于动态批量评估的绿色无梯度优化方法框架(GRACE).

输入: 黑盒优化器(optimizer), 模型训练接口(evaluate), 预算 *budget*, 更新频率 M;

1: 初始化训练批量集合$\{\mathcal{B}_1,\mathcal{B}_2,\ldots,\mathcal{B}_N\}$和动态评估表 R

2: t=0, $\mathcal{D}_0=\varnothing$;

3: **while** t<*budget* **do**

4: **if** $t\%M$=0 **then**

5: 使用算法 2, 根据动态评估表 R 构造批量相似树$\mathcal{T}$;

6: 向树$\mathcal{T}$和动态评估表 R 中添加一个新的批量;

7: **end if**

8: 基于批量相似树$\mathcal{T}$, 根据 4.3 节得到批量索引集合 $i\in\{i_1,i_2,...,i_{t_n}\}$;

9: $\boldsymbol{\theta}_t\leftarrow$优化器给出待评估的参数 $Optimizer(\mathcal{D}_t)$;

10: $\{y_{t,i}\}_{i\in\{i_1,i_2,...,i_{t_n}\}}\leftarrow$训练模型得到评估值 $Evaluate(\{(\boldsymbol{\theta}_t,\mathcal{B}_i)\}_{i\in\{i_1,i_2,...,i_{t_n}\}})$;

11: 将 $\{y_{t,i}\}_{i\in\{i_1,i_2,\ldots,i_{t_n}\}}$ 记录动态评估表 R 中，并计算出 $y_t=\frac{1}{t_n}\sum_{i\in\{i_1,i_2,\ldots,i_{t_n}\}}y_{t,i}$；

12: $\mathcal{D}_{t+1}=\mathcal{D}_t\cup\{\boldsymbol{\theta}_t,y_t\}$;

13: $t\leftarrow t+1$;

14: **end while**

15: $\boldsymbol{\theta}^*\leftarrow Optimizer(\mathcal{D}_{t+1})$;

输出：算法找到的最优解$\boldsymbol{\theta}^*$.

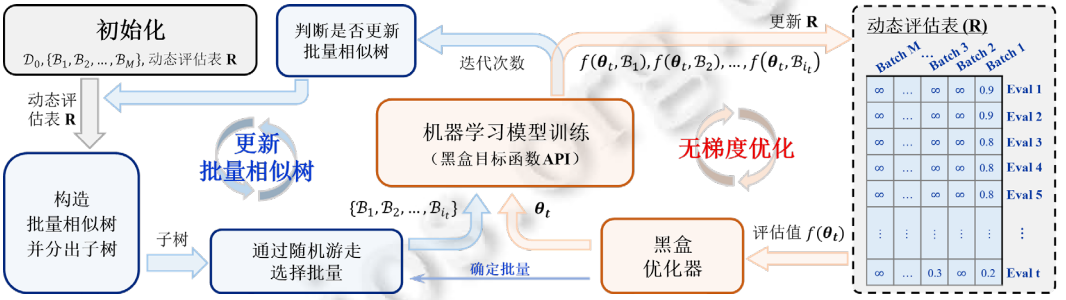


图 3 基于动态批量评估的绿色无梯度优化方法框架(GRACE)

从图 3 可以看出，算法的结构分为两个部分. 在图 3 中，右侧的红色顺时针循环代表了无梯度优化的通用框架，可以带入多种无梯度优化算法(如演化策略、贝叶斯优化等)进行实践. 左侧的蓝色逆时针循环表示算法评估期间更新批量相似树(通过批量相似树来表示不同批量之间的相似度关系)并确定评估使用的批量的过程. 黑盒 API(即优化任务的目标函数)评估后得到的解会保存在最右边的动态评估表 R 中，并将不同批量下评估结果的均值反馈给无梯度优化算法，进而完成优化器的一次无梯度优化迭代.

在图 3 右侧的红色顺时针循环表示的无梯度优化过程中，由黑盒优化器(即对应的无梯度优化方法)提供待评估的解，通过调用黑盒 API 得到解在各个批量下的评估值，将其记录在动态评估表 R 中并进行相关处理，完成黑盒优化过程. 在图 3 左侧的蓝色逆时针循环表示的更新批量相似树过程中，其核心是使用动态评估表 R 构建批量之间的相似度关系，进而指导批量样本的选择与样本子集的构建. 在这个过程中，算法会定期根据动态评估表的内容构建一棵名为批量相似树的二叉树，通过批量相似树来表征各批量之间的相似程度. 具体而言，批量相似树的每个叶节点均对应一个批量，评估值高度相似的批量叶节点会被放置在同一父节点下，表示批量内部样本的高度相似，每一个非叶节点中存储着其子节点的相似度信息. 在每次优化之前，会基于当前的批量相似树，使用批量索引选择策略选择多个批量参与到本次优化的过程(图 3 中蓝色字体确定批量的过程). 通过图 3 左侧的蓝色循环过程，动态选择出差异尽可能大的批量进行评估，以较小成本获得更准确的评估值，从而使算法能够绿色高效地完成优化任务.

算法 1 展示了 GRACE 的完整过程. 算法每消耗 M 预算时更新一次批量相似树的结构，并添加一个新的批量进行评估. 在每次优化迭代时，算法都会基于当前批量相似树的结构选择要使用的批量，并将其与黑盒优化器提供的待评估解拼接后调用黑盒评估 API，以返回每个批量在当前参数组合下的评估值. 随后，评估值会被记录在动态评估表 R 中，并根据动态评估表的记录向优化器返回最终评估值. 通过不断循环上述过程，直到消耗完昂贵优化的预算，由黑盒优化器返回最终的评估值作为优化结果，完成整个优化过程. 这里黑盒优化器可以是一般的无梯度优化方法，在本文实验中，选择了优化效果较为稳定的 CMA-ES[45]算法.

4.2 批量相似树构造算法

在本节中将介绍批量相似树的构造算法，算法首先评估不同批量之间的相似度，然后根据相似度信息建立批量之间相似关系的二叉树结构.

首先讨论动态评估表 R 的构建与动态更新的过程. 如前文所述，动态评估表 R 是用来记录解在不同批量

下评估值的动态数据结构, 其结构如图 4 所示. 动态评估表是一个二维表结构, 可以根据优化迭代次数(即不同的解的评估)和批量数的变化而改变大小. 在优化过程中, 动态评估表作为存储解在每组批量作为训练集的评估结果的数据仓库. 如果在某次优化迭代中没有在一个批量上对解进行评估, 则对应的评估值会被记录为无穷大. 如图 4 所示, 动态评估表 R 记录了各种参数在不同批量训练集下的性能, 表格的两个维度表示批量编号和优化迭代次数. 随着算法的迭代进行, 新的批量被添加到树结构中, 并且新的参数组合被评估, 动态评估表的大小也随之动态增长.

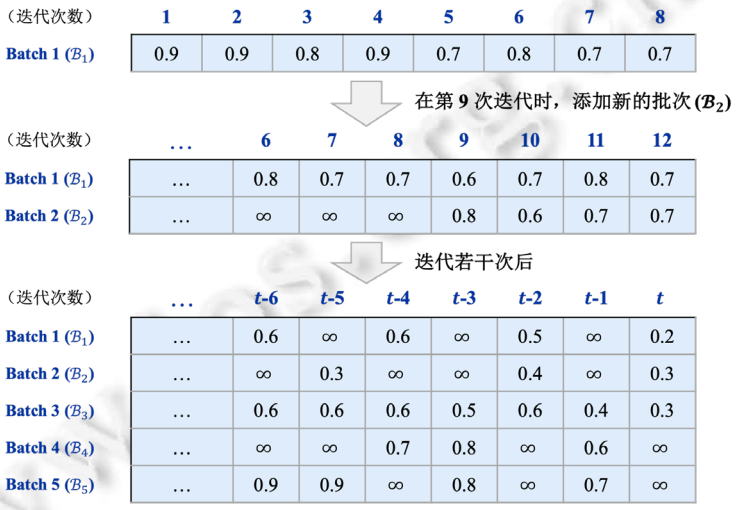


图 4 动态评估表 R 随优化迭代更新的一个示例

下面介绍批量相似树的构造算法, 如算法 2 所示, 算法旨在构建能够表示不同批量之间相似度关系的二叉树. 在计算不同批量之间的相似度时, 根据对于同一个解的评估结果差异越小的批量相似度越大的思想, 本文通过度量距离来计算相似度. 由于模型训练后期的损失或准确率通常在 10^{-2}~10^{-4} 数量级, 因此在后续的实验中, 均采用 l_1 范数来评估不同批量之间的相似度, 即: 批量$\mathcal{B}_i$ 和批量$\mathcal{B}_j$ 之间的相似度可由 $d(\mathcal{B}_i,\mathcal{B}_j)=\sum_{k\in\mathcal{I}}\|f(\theta_k,\mathcal{B}_i)-f(\theta_k,\mathcal{B}_j)\|_1$ 计算, 其中, $\mathcal{I}$为批量$\mathcal{B}_i$ 和批量$\mathcal{B}_j$ 距离当前最近 T 次评估的同时评估的迭代次数序号. 如图 4 所示: 当 T=2 时(即集合$\mathcal{I}$的大小为 2), 对于批量$\mathcal{B}_1$ 和批量$\mathcal{B}_2$, $\mathcal{I}=\{t-2,t-1,t\}$; 对于批量$\mathcal{B}_4$ 和批量$\mathcal{B}_5$, $\mathcal{I}=\{t-3,t-2,t-1\}$. 在分别计算批量两两之间的距离后, 便可以根据距离信息构建距离邻接矩阵 D. 随后, 基于距离邻接矩阵的数值, 自底而上依次合并距离最近的节点, 直至生成批量相似二叉树. 具体而言, 邻接矩阵中距离最近的节点合并到同一个父节点, 并在父节点中存储其子节点的距离. 合并节点后, 从邻接矩阵 D 中删除合并节点的信息, 并在邻接矩阵中添加新节点与剩余节点之间的距离, 更新邻接矩阵 D. 上述过程如算法 2 的第 4–9 行所示, 重复迭代直到得到根节点, 即完成一棵存储不同批量之间相似度关系的二叉树, 即批量相似树.

算法 2. 批量相似树构造算法.

输入: 动态评估表 R, 参数 T;

1: $R[t-T,t]\leftarrow$从 R 获取最近 T+1 次迭代的结果
2: 计算距离 $d(\mathcal{B}_i,\mathcal{B}_j)$;
3: 得到距离邻接矩阵 $D=\{d_{ij}\}_{d_{ij}=d(\mathcal{B}_i,\mathcal{B}_j)}$;
4: **While** D 中的节点数>1 **do**
5: 找到 D 中距离最近的两个节点, 并记录其距离;
6: 合并上述两个节点为一个新节点;
7: 更新新节点与其他节点之间的距离;

8: 更新距离邻接矩阵 D;

9: **end while**

输出: 批量的二叉树结构$\mathcal{T}$.

图 5 展示了使用 l_1 范数度量距离在 T=2 时使用示例数据构建批量相似二叉树的过程. 算法首先从记录动态评估表 R 中提取最近的若干次评估结果, 以供后续计算距离使用(步骤 1); 然后, 使用 l_1 范数计算批量之间的相似度, 并将结果记录在距离邻接矩阵中(步骤 2); 通过不断合并最近的节点并根据距离矩阵生成新节点, 直到距离邻接矩阵中只包括一个根节点时, 便完成了批量相似二叉树的初步构建(步骤 3); 最后, 算法返回批量相似树的数据结构, 树的每个叶节点对应一个批量, 非叶节点存储其子节点之间的距离(步骤 4).

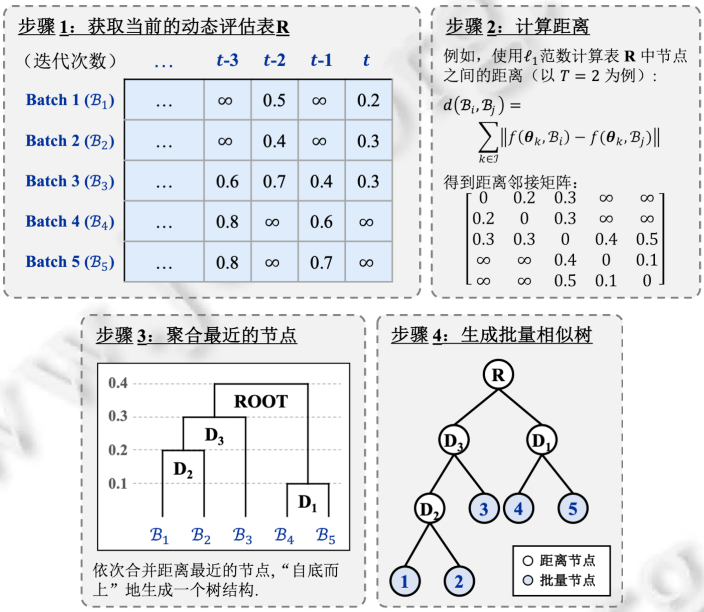


图 5 构建批量相似树的一个例子

一个值得注意的问题是: 在新节点添加到邻接矩阵时, 如何计算新节点与其他节点之间的距离. 在树的构建过程中, 当两个节点合并形成一个新节点时, 邻接矩阵会删除这两个节点并添加合并后的新节点与其他节点之间的距离信息, 此时, 通常可以使用合并前的节点的距离信息来评估新节点的距离信息, 比如取合并前两节点中较大的值、较小的值或均值作为新的距离信息等. 例如: 如果节点 A 和 B 合并形成节点 X, 并且需要计算 X 与节点 C 之间的距离, 则可以使用现有的 A, B 和 C 之间的距离信息 $d(A,C)$和 $d(B,C)$来计算 X 和 C 的距离 $d(X,C)$. 在图 6 及本文的实验部分中, 假设新节点继承合并前节点的最小距离, 即:

$$d(X,C)=\min\{d(A,C),d(B,C)\}.$$

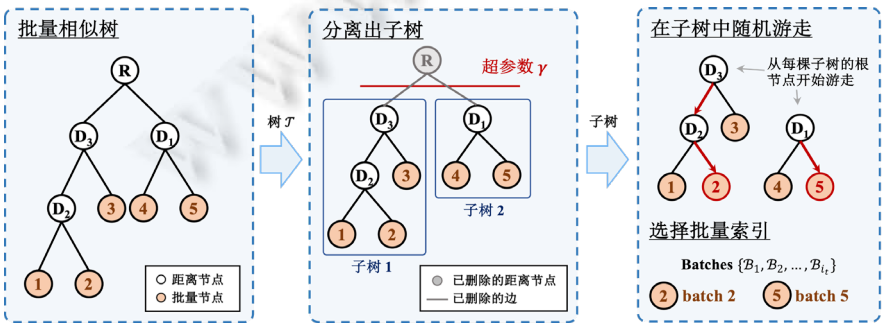


图 6 批量索引选择策略的一个例子

4.3 批量索引选择策略

在构建的批量相似树后, 本节阐述如何根据批量相似树选择优化过程中需要被评估的批量, 即批量索引选择策略. 上一节中介绍了批量相似树的两种节点: 叶节点表示批量, 非叶节点存储其两个子节点之间的相似度信息. 显然, 在批量相似树中, 如果处于同一个父节点下的两个子节点距离根节点较远, 则表明它们之间的相似度很大(如图 5 步骤 4 中的叶节点 1 和叶节点 2). 由于本文所考虑的优化任务的昂贵性, 因此评估时选取批量的原则应该是: 在实现尽可能高的评估准确性的同时, 用尽可能少的批量样本进行评估. 换句话说, 应该用尽可能少的批量来代表全体样本, 达到类似于评估全体样本的评估效果. 本文方法的基本想法是: 避免选择相似度高的批量. 相似度高的批量对解的评估值是高度相似的, 选择它们可能不能为模型提供多样的样本、探索更大的样本空间, 进而导致优化陷入局部极小值.

在选择批量时, 批量相似树被分割成多个子树, 并认为同一子树中的叶节点相似度很高, 不同子树中的叶节点表现较大的差异性. 这是因为批量相似树是根据批量的相似性构建的, 具有更高相似度的节点会较早地分配给同一个父节点. 随后, 可以从每个子树中随机选择节点, 这样便可以确保选择出具有较大差异的批量进行评估. 通过上述过程可以发现, 基于批量相似树选择优化过程中使用的批量需要解决两个问题: 首先, 需要确定如何将批量相似树分割为包含高度相似叶节点的多个子树; 其次, 需要确定从这些子树中选择出需要评估的批量方法. 这两个问题即是批量索引选择策略的关键. 对于第 1 个问题, 通过引入超参数γ来将二叉树分割成子树, 即: 如果一个节点是叶节点或其存储的距离小于γ, 而其父节点存储的距离大于γ, 则以该节点为根的子树将被视为选择的子树. 通过这种方法, 可以将批量相似树分割成多个子树, 且每个子树中的叶节点相似度很高. 对于第2个问题, 通过随机游走策略可以从每个子树中找出合适的节点来代表该子树. 随机游走策略从每个子树的根节点开始, 每次以等概率选择其左右节点之一. 这个过程重复进行, 直到到达一个叶节点, 则选择该叶节点对应的批量作为代表该子树的批量. 随机游走方法旨在增加选择树的上层节点的概率, 因为这些节点与子树中的其他节点的差异更大. 因此, 使用该策略会返回更具差异性的批量. 图6展示了使用批量索引选择策略的一个示例. 首先, 根据最左侧的批量相似树的结构, 将分离超参数γ应用于树的符合分离条件的边, 得到由多个子树组成的森林; 然后, 与原始树结构的根节点连接的子树被移除; 最后, 从每个剩余子树的根节点执行随机游走策略, 选择表示该子树的叶节点, 得到需要评估的批量索引集合.

5 实验分析

5.1 实验设置

实验部分, 本文分别在基于语言模型即服务的黑盒微调(BBT)任务(使用 RoBERTa LARGE 作为基础大模型, 该模型的参数量为 355 M)[16]和机器学习 LightGBM 模型[21]超参数调优任务上进行了对比实验、消融实验、超参数分析等实验, 并对算法运行过程和实验结果进行分析.

模型与任务的选择以及相关的参数设置见表 1.

表 1 任务、模型、数据集等基本情况

任务与模型	优化任务的维度	数据集名称	数据集大小(K)	批量大小	评估总量预算	数据集特点
BBT[16] (RoBERTa LARGE)	500 和 50	SST-2	67	8 (4×2)	6 000	语言情感(二分类)
		AG's News	120	16 (4×4)	6 000	新闻话题(四分类)
LightGBM[21]	11	Digits	1.8	50, 100	300	手写数字识别(多分类)
		Housing	20.6	100	500	房价回归
		CovType	581	100	500	植被类型识别(多分类)

对于 NLP 大模型 BBT 任务, 本文选择在二分类数据集 SST-2 和四分类数据集 AG's News 上进行实验. 由于基于贝叶斯优化的部分对比算法具有高维瓶颈, 为了更好地对各种方法进行对比, 对比实验部分采用了 50 (低维)和 500(高维)两种模型参数量进行实验(在所有对比方法进行参数量 50 的实验, 在非贝叶斯优化的对比

方法上进行参数量 500 的实验; 原始任务[16]中的参数量是 500). 对于批量大小的设置, 本文按照 Few-shot 的有关设定, 设计批量大小与分类任务的类别数成比例(即在 SST-2 上的批量大小为 8, 在 AG's News 上的批量大小为 16). 类似地, 在 LightGBM 模型上进行了两个多分类任务和一个回归任务, 数据集大小涉及 3 个不同的数量级.

在基于语言模型即服务的黑盒微调(BBT)任务上, 使用预训练的 RoBERTa LARGE 模型[16], 通过优化器对维度为 500 或 50 的提示词进行微调优化, 进而完成一系列自然语言处理任务(见表 1). 在机器学习 LightGBM 模型[21]超参数调优任务上, 通过优化如表 2 所示的 11 个模型超参数, 进而使得 LightGBM 模型能在各任务上得到较好的效果(表中连续类型的超参数, 优化时保留 4 位小数; 对数连续类型的超参数, 在对数空间上优化, 优化时保留 5 位小数).

表 2 LightGBM 超参数优化任务的解空间表

LightGBM 超参数	含义	数值类型	解空间(优化时使用的范围)
learning_rate	学习率	连续	[0.05,0.55]
n_estimators	boosting 迭代次数	离散	在区间[50,350]内的整数
min_split_gain	执行节点分裂的最小增益	连续	[0,1]
min_child_sample	一个叶子上的最小数据量	离散	在区间[5,105]内的整数
min_child_weight	一个叶子上的最小海森和	对数连续	$[1\times10^{-4},1\times10^{-1}]$
max_depth	树的最大深度	离散	{3,4,5,6}
num_leaves	树上的叶子节点个数	离散	在区间[5,30]内的整数
subsample	随机特征参数, 缓解过拟合	连续	[0.8,1]
colsample_bytree	随机特征参数, 缓解过拟合	连续	[0.8,1]
reg_alpha	L_1 正则化超参数	对数连续	$[1\times10^{-2},1\times10^{3}]$
reg_lambda	L_2 正则化超参数	对数连续	$[1\times10^{-2},1\times10^{3}]$

在所有对比实验中, 超参数γ的取值均为 5.0. 实验中, 对于基于演化策略的算法, 种群大小分别设置为 20(BBT 实验)或 5(LightGBM 实验); 对于 GRACE 算法, 每隔 600 次(BBT 实验)或 25 次(LightBGM 实验)更新一次批量相似树并向树中添加一个新批量, 在评估距离时, 参数 T 取值为 100(BBT 实验)或 10(LightBGM 实验). 对于所有的对比实验和超参数实验, 均使用随机种子 21~30 进行 10 次随机实验, 记录随机实验的均值和方差信息. 算法运行过程中, 我们记录并绘制各算法找到的当前最优解对应的结果. 对于 BBT 任务, 评价指标均与原文[16]保持一致, 使用默认的损失函数和准确率; 对于 LightGBM 超参数优化任务, 均使用 Sklearn 自带的评估函数进行评估, 其中, 回归数据集 Housing 采用 R^2 指标(R^2 score)进行评估, 分类数据集 CovType 和 Digits 采用准确率(accuracy)进行评估. 在 BBT 系列实验中, 预算选取为 6 000, 每消耗 600 预算时, 在全量数据集和测试集上测试一次性能; 在 LightGBM 系列实验中, 预算选取为 500, 每消耗 50 预算时, 在全量数据集上测试一次性能.

在对比实验中, 本文将 GRACE 与以下 7 种算法进行了比较.

(1) BOHB[35]: 一种基于 Hyperband 调度算法[57,58]的多保真度优化算法, 使用基于 TPE 的代理模型有效地分配采样配置;

(2) ASHA[36]: 一种基于 Successive Halving 资源分配算法的多保真度优化算法, 利用早停机制将更多资源分配给有前途的配置;

(3) Batch-BO[59]: 一种并行的高保真度贝叶斯优化方法;

(4) Few-shot[16,60]: 在小样本的设定下, 仅使用少量训练数据或固定的批量优化模型. 具体而言, 在分类任务中, 小样本训练集中包含等量的各类别样本; 在回归任务中, 从全量训练集中随机抽取少量样本构成小样本训练集;

(5) Average[41,61]: 一种噪声优化算法, 通过重新采样并取多个样本评估的平均值来减少评估误差的影响. 在实验中, 重复采样了 3 个批量;

(6) Threshold[32]: 一种噪声优化方法, 通过指定阈值来控制无梯度优化算法的解的更新. 即: 在优化迭代过程中, 若当前解比上次迭代解的性能提升超过阈值, 则使用这组解更新无梯度优化器内部的模

型. 本文的实验结果为阈值取 0.5 时的结果(在复现对比方法时, 尝试过各种阈值选择, 阈值为 0.5 时总体性能最好);

(7) Stochastic: 无放回的随机选择批量进行评估.

在所有对比方法中, BOHB、ASHA、Batch-BO 为常见的多保真度背景下的超参数优化算法, Few-shot 为小样本学习下的常用设定, Average、Threshold 为两种常见的噪声优化方法, Stochastic 为随机使用批量的对比方法.

上述对比方法中, Few-shot、Average、Stochastic 可以视作对 GRACE 算法的消融. 对于 Few-shot, 这种方法是大规模语言模型微调任务的常用设置, 即固定使用 1 个批量进行评估; 对于 Average, 该方法是经典的噪声优化方法之一, 其每次优化时随机使用多个批量样本(实验中使用 3 个), 取均值作为最终的评估结果; 对于 Stochastic, 这种方法类似于梯度优化算法中的随机梯度下降法, 每次优化随机使用一个批量进行评估. 由于 Few-shot、Average、Stochastic 缺少动态确定待评估的批量数量和动态选择有代表性的批量进行评估的过程, 可以视为简化的批量处理策略, 因此这几种算法可视为消融了 GRACE 算法关键部件的退化版算法. 此外, 由于与绿色低碳目标的抵触, 本文没有设计使用全量数据集的优化方法进行实验. 以 CovType 任务为例, 其使用全量数据集优化一次约需要消耗 5 000 次评估预算(如表 1 所示, 其数据集大小为 581K, 批量大小为 100, 在全量数据集上评估即在约 5 810 个批量上评估, 至少需要 5 810 预算); 而在绿色低碳背景下, 该实验的预算只有 500 次, 无法进行一次完整的优化.

5.2 结果分析

下面结合对比实验结果(如图 7、表 3 所示(表中平均评估时间是指完成一次无梯度优化的迭代过程消耗时间的均值; 在 SST-2 和 AG's News 上, 高维指优化任务的维度为 500, 其中, 记录为"–"的数据表示在 36 小时无法得到实验结果; 低维指优化任务的维度为 50, 后同))、超参数实验结果(如图 8(d)–图 8(h)所示)、算法行为分析图(如图 8(a)–图(c)所示), 对实验结果分别从绿色低碳、准确性与高效性、可理解性、稳健性这 4 个方面进行分析.

(1) 绿色低碳.

本文算法和实验的绿色低碳性能主要体现在 3 个方面——任务背景与实验设定绿色低碳、算法运行时间短、实验不需要大量显卡.

- 在任务背景与实验设定方面, 算法基于批量评估, 相较于评估全量数据, 评估能耗约降低 1 000 倍; 此外, BBT 是大模型提示词微调任务, 大模型内部参数保持不动, 优化时仅微调提示词参数(参数量小), 符合绿色要求理念;
- 在资源消耗方面, 所有 BBT 单次实验(预算 6 000)的运行时间均不超过 30 分钟, 所有 LightGBM 单次实验(预算 500)的运行时间均不超过 5 分钟, 算法能在保证精度的同时降低能耗, 减少资源消耗;
- 在实验设备方面, 本文在 BBT 上的全部实验仅使用一张 3080 显卡, 在 LightGBM 上的全部实验均未使用显卡(仅 8 核 CPU 的 Apple M1 Pro 芯片), 上述较低能耗的实验设定, 符合绿色低碳理念.

(2) 准确性与高效性.

通过在 BBT 和 LightGBM 上的一系列对比实验, 本文从解的质量、收敛效果、运行时间等多个角度说明 GRACE 算法的准确性和高效性.

- 在解的质量方面, 本文提出的 GRACE 算法可以在多个任务上搜索到质量最好的解. 如表 3 所示, 表中统计了在不同任务中, 不同对比算法寻得的最优解以及平均评估时间. 在 LightGBM 的 4 个数据集上, GRACE 算法均能寻得性能最好的解; 在 BBT 系列实验中, GRACE 算法也都可以找到高质量的解, 其中, 低维 SST-2 任务虽然只取得第二名的优化性能, 但却取得了最好的泛化性能. 由于为了适应基于贝叶斯优化的系列算法, 实验中设计了 BBT 的低维实验, 这可能是 AG's News 低维只取得次优性能的原因, 但在原始的 AG's News 任务上(高维), GRACE 仍能取得最好的实验性能;
- 在收敛效果方面, GRACE 算法均展现出较快的收敛速度和较好的收敛效果. 如图 7 所示, 图中展示了

各种对比算法在不同任务上的收敛效果以及 BBT 系列任务的泛化性能(泛化性能只在测试集上测试当前最优解的性能, 测试集并不会参与到训练过程). 从图 8 中可以看出, GRACE 算法在各任务上均能够较快地收敛; 从表 3 中展示的各种算法的最终收敛情况, 可以看出 GRACE 可以收敛到高质量解;

- 在运行时间方面, 从表 3 中的平均评估时间可以看出, GRACE 算法能够很好地权衡优化时间和解的性能, 在使用远少于多保真度优化方法(BOHA, ASHA)运行时间的情况下, 实现高质量的收敛.

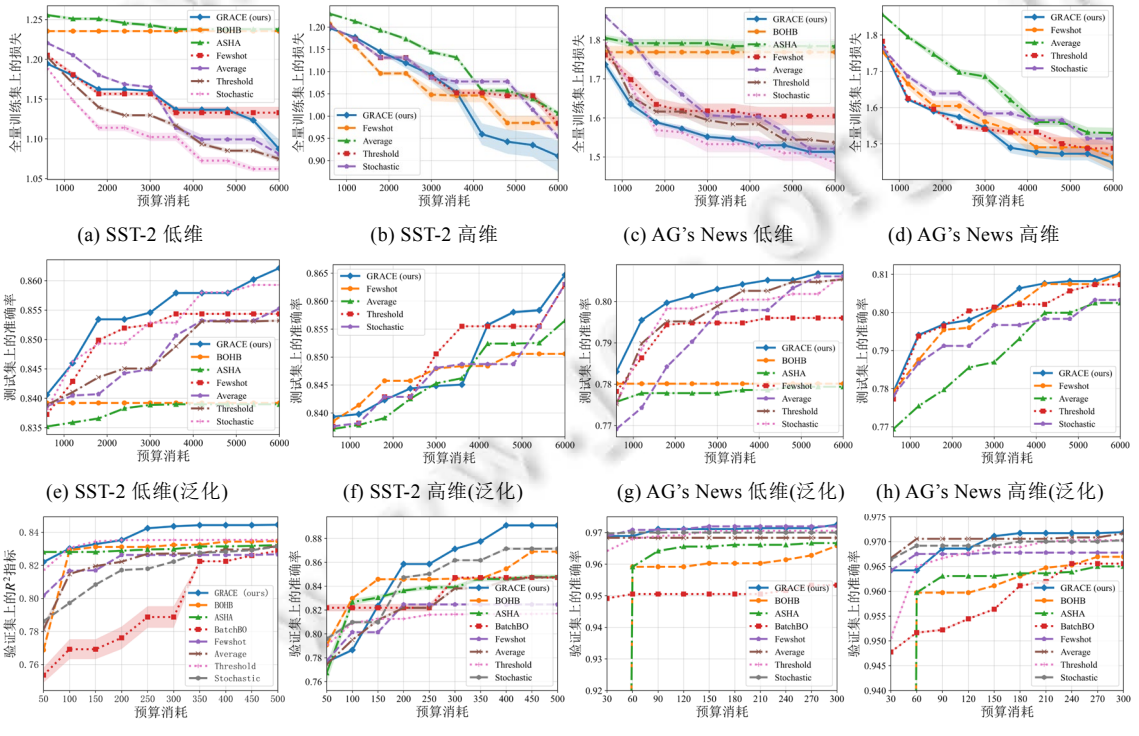
(3) 稳健性.

我们设计了针对 GRACE 超参数γ的实验, 通过不同超参数配置下的实验结果说明 GRACE 算法的超参数稳健性. 超参数γ控制着对批量相似树的分割, 同时间接控制着每次评估使用的批量数. 在算法超参数稳健性分析部分, 本文通过选取γ的不同取值{1.0,2.0,4.0,5.0,6.0,8.0,10.0,12.0}, 观察 GRACE 的性能变化. 在 SST-2 数据集和 Housing 数据集上的超参数实验结果分别如图 8(e)、图 8(g)所示. 从图中可以看出: 除了较为极端的(1.0 和 12.0), 其他超参数取值均能取得较好的收敛效果, 这体现了 GRACE 算法的稳健性. 同时可以发现: 尽管在一定范围的超参数配置下, GRACE 算法在各任务上均能取得较好的收敛效果, 但对于不同的任务, 超参数γ的最优取值可能有所不同, 这也是未来工作可进一步探索的问题, 即自适应调节γ.

(4) 可理解性.

通过展示 GRACE 算法在优化过程中使用批量数目的动态变化, 我们可以加强对算法行为的理解, 进而分析算法的行为特征. 同时, 针对不同的批量大小进行实验, 观察算法的性能变化.

通过动态展示优化过程中评估使用的批量数, 进而对算法的内部运行过程进行深入的分析. 在这部分实验中, 选取了分类任务 SST-2、AG's News 和回归任务 Housing 进行进一步的分析, 实验结果如图 8(a)~图(c)所示. 同时, 在超参数实验中, 本文也绘制了 GRACE 在超参数γ不同取值下的评估批量数动态变化图, 如图 8(f)和图 8(h)所示.



(a) SST-2 低维 (b) SST-2 高维 (c) AG's News 低维 (d) AG's News 高维

(e) SST-2 低维(泛化) (f) SST-2 高维(泛化) (g) AG's News 低维(泛化) (h) AG's News 高维(泛化)

(i) Housing (j) CovType (k) Digits (50) (l) Digits (100)

图 7 对比实验结果图

表 3 对比实验结果表

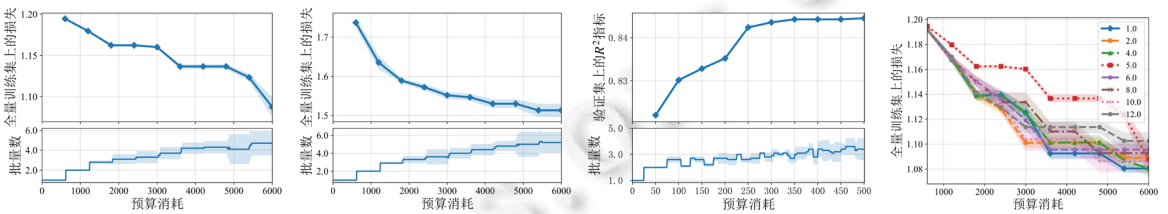
数据集	算法	全量训练集上的损失	测试集上的准确率	平均评估时间(s)
SST-2(低维)	BOHA	1.2355±0.0013	0.8392±0.0000	10.1394
	ASHA	1.2379±0.0024	0.8390±0.0000	8.9035
	BatchBO	–	–	–
	Fewshot	1.1329±0.0073	0.8544±0.0001	0.1360
	Average	1.0809±0.0096	0.8553±0.0002	0.4095
	Threshold	1.0748±0.0034	0.8532±0.0002	0.5516
	Stochastic	**1.0622±0.0034**	0.8593±0.0001	0.1403
	GRACE (ours)	1.0874±0.0118	**0.8622±0.0000**	0.3114
SST-2(高维)	BOHA	–	–	–
	ASHA	–	–	–
	BatchBO	–	–	–
	Fewshot	0.9847±0.0163	0.8506±0.0002	0.1472
	Average	1.0067±0.0049	0.8565±0.0000	0.4162
	Threshold	0.9839±0.0125	0.8630±0.0002	0.6302
	Stochastic	0.9541±0.0075	0.8631±0.0001	0.1451
	GRACE (ours)	**0.9102±0.0351**	**0.8647±0.0004**	0.3016
AG's News(低维)	BOHA	1.7687±0.0161	0.7801±0.0002	16.5034
	ASHA	1.7843±0.0134	0.7794±0.0002	14.2637
	BatchBO	–	–	–
	Fewshot	1.6048±0.0236	0.7960±0.0002	0.6175
	Average	1.5213±0.0135	0.8061±0.0002	1.8549
	Threshold	1.5368±0.0239	0.8054±0.0003	1.5093
	Stochastic	**1.4852±0.0240**	0.8066±0.0002	0.6597
	GRACE (ours)	1.5132±0.0166	**0.8068±0.0002**	1.6364
AG's News(高维)	BOHA	–	–	–
	ASHA	–	–	–
	BatchBO	–	–	–
	Fewshot	1.4641±0.0362	0.8098±0.0003	0.4268
	Average	1.5299±0.0148	0.8025±0.0001	1.2690
	Threshold	1.4877±0.0242	0.8073±0.0003	1.5411
	Stochastic	1.5146±0.0168	0.8033±0.0002	0.4588
	GRACE (ours)	**1.4478±0.0246**	**0.8102±0.0002**	1.0812

表 3 对比实验结果表(续 1)

数据集	算法	验证集上的 R^2 指标或准确率	平均评估时间(s)
Housing	BOHA	0.8347±0.0000	40.9581
	ASHA	0.8323±0.0001	77.8718
	BatchBO	0.8288±0.0001	16.4311
	Fewshot	0.8267±0.0001	1.2787
	Average	0.8317±0.0000	2.7311
	Threshold	0.8354±0.0001	0.9612
	Stochastic	0.8314±0.0000	0.8966
	GRACE (ours)	**0.8446±0.0001**	3.6170
CovType	BOHA	0.8690±0.0001	64.6879
	ASHA	0.8478±0.0019	73.1404
	BatchBO	0.8472±0.0007	43.9048
	Fewshot	0.8247±0.0010	0.3832
	Average	0.8475±0.0011	19.5978
	Threshold	0.8170±0.0006	6.2001
	Stochastic	0.8714±0.0011	1.5143
	GRACE (ours)	**0.8910±0.0006**	8.2153
Digits(批量大小=50)	BOHA	0.9658±0.0000	12.3648
	ASHA	0.9667±0.0000	13.3524
	BatchBO	0.9533±0.0001	6.9905
	Fewshot	0.9717±0.0001	0.4019
	Average	0.9683±0.0001	1.0083
	Threshold	0.9706±0.0000	0.8316
	Stochastic	0.9700±0.0001	0.3711
	GRACE (ours)	**0.9725±0.0000**	1.0268

表 3 对比实验结果表(续 2)

数据集	算法	验证集上的 R^2 指标或准确率	平均评估时间(s)
Digits(批量大小=100)	BOHA	0.9669±0.0001	21.4368
	ASHA	0.9650±0.0000	20.1456
	BatchBO	0.9656±0.0001	11.1024
	Fewshot	0.6978±0.0001	0.4851
	Average	0.9712±0.0000	1.4598
	Threshold	0.9703±0.0000	1.7442
	Stochastic	0.9703±0.0001	0.4407
	GRACE (ours)	**0.9713±0.0001**	1.1679



(a) SST-2 上算法分析 (b) AG's News 上算法分析 (c) Housing 上算法分析 (d) SST-2 上超参数γ分析

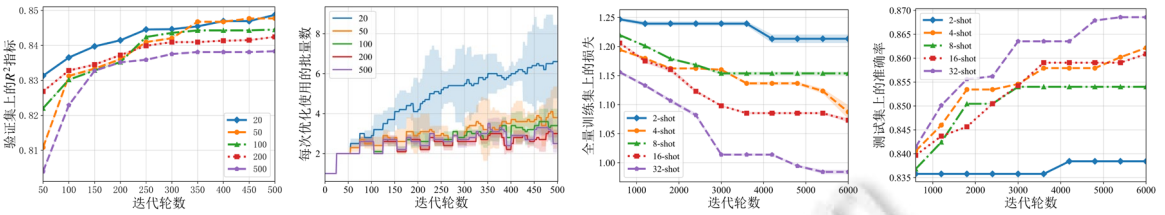
(e) SST-2 上超参数γ泛化分析 (f) SST-2 上超参数γ分析 (g) Housing 上超参数γ泛化分析 (h) Housing 上超参数γ分析

图 8 算法行为分析与超参数分析实验结果图

从图 8(a)–(c)中可以看出: GRACE 算法能够动态控制评估中使用的批量数目, 具体体现在优化前期批量数的迅速增加以及优化后期的逐渐稳定; 同时, 随着优化后期逐渐出现新的更有代表性的批量, 批量数会缓慢增加. 这个过程符合算法设计时对批量评估的预期, 即: 既不浪费预算、评估过多重复的区别较小的批量, 又能尽可能多的评估区别较大的批量. 同时, 从图 8(f)、图 8(h)可以看出: 当 GRACE 算法的γ取较为极端的情况时, 批量数的变化与理想的变化方式稍有出入, 这也是超参数实验中γ取 1.0 或 12.0 时算法的性能会出现轻微下降的一种解释.

此外, 图 9 展示了不同批量大小的各任务实验结果, 并展示了不同批量大小设置下的批量数动态变化示意图. 在 SST-2 数据集中, 我们选取了 Few-shot 为{2,4,8,16,32}(对应批量大小为{4,8,16,32,64})的设定; 在 Housing 数据集中, 我们选取了批量大小为{20,50,100,200,500}的设定. 超参数γ均设定为 5.0. 通过实验结果可以看出: 在大多情况下, 随着批量大小的增大, 算法的性能越来越好. 但实验中也有一些小批量效果好于大批量的现象, 造成这种现象的原因如下.

- 首先, 不同曲线的优化次数与预算消耗之间的关系并不相同(因此, 实验结果图的横轴并不是"预算消耗"而是"迭代轮数"). 例如: 假设批量大小为 100 时优化一次的预算消耗为 1, 则批量大小为 200 时优化一次的预算消耗应该为 2, 这意味着完成一次优化迭代时, 不同的批量大小的预算消耗是不同的;
- 其次, 批量的构造过程是完全随机的, 由于训练数据的数量是有限的, 因此, 不同批量的数据质量是无法保证的. 而上述实验结果为重复实验 10 次的平均结果, 因此实验结果具有一定的随机性;
- 同时, 出于绿色低碳的考虑, 在前文的对比实验中, 本文并未选取性能较好的大批量, 而是使用了小批量(批量大小如表 1 所示)完成优化算法, 并针对小批量给出了超参数γ的推荐值, 既能实现较高的准确率, 又降低了能耗, 以尽可能小的代价最大程度地完成优化目标.



(a) Housing (b) Housing (分析) (c) SST-2 低维 (d) SST-2 低维(泛化)

(e) SST-2 低维(分析) (f) SST-2 高维 (g) SST-2 高维(泛化) (h) SST-2 高维(分析)

图 9 不同批量大小在各任务上的实验结果图

6 总 结

本文提出了基于动态批量评估的绿色无梯度优化方法框架GRACE, 基于训练子集的相似性, 通过自适应动态调整解的评估使用的训练数据子集大小, 在保证评估准确性与稳定性的同时, 有效地降低了评估代价, 实现了绿色低碳且高效准确的优化. 在实际任务中, 如语言模型即服务提示词黑盒微调和模型超参数优化, 通过实施对比实验、消融实验等, 展示了 GRACE 算法在时间效率、优化性能、资源效益等方面的优越性. 通过在超参数的不同取值下实验, 并分析算法运行过程中批量数的动态变化情况, 说明了算法的稳健性和可靠性. 未来, 我们将进一步设计超参数γ的自适应调节解决方案, 以实现更佳的性能. 此外, 我们将探索 GRACE 在体量更大的数据集以及模型上的应用, 并提出针对无梯度优化任务的绿色评价新指标.

References:

[1] Janiesch C, Zschech P, Heinrich K. Machine learning and deep learning. Electronic Markets, 2021, 31(3): 685−695.

[2] Sarker IH. Machine learning: Algorithms, real-world applications and research directions. SN Computer Science, 2021, 2(3): 160.

[3] Shih FY. Image Processing and Pattern Recognition: Fundamentals and Techniques. John Wiley & Sons, 2010.

[4] Lu D, Weng Q. A survey of image classification methods and techniques for improving classification performance. Int'l Journal of Remote Sensing, 2007, 28(5): 823−870.

[5] Shen Z, Cui CR, Dong GX, *et al*. Unified image aesthetic and emotional prediction based on deep multi-task learning. Ruan Jian Xue Bao/Journal of Software, 2023, 34(5): 2494−2506 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6487.htm [doi: 10.13328/j.cnki.jos.006487]

[6] Yu D, Deng L. Automatic Speech Recognition. Berlin: Springer, 2016.

[7] Gaikwad SK, Gawali BW, Yannawar P. A review on speech recognition technique. Int'l Journal of Computer Applications, 2010, 10(3): 16−24.

[8] Wolf T, Debut L, Sanh V, *et al*. Transformers: State-of-the-art natural language processing. In: Proc. of the 2020 Conf. on Empirical Methods in Natural Language Processing. 2020. 38−45.

[9] Chowdhary K. Natural Language Processing. Springer, 2020.

[10] Galassi A, Lippi M, Torroni P. Attention in natural language processing. IEEE Trans. on Neural Networks and Learning Systems, 2020, 32(10): 4291−4308.

[11] Wang NY, Ye YX, Liu L, *et al*. Language models based on deep learning: a review. Ruan Jian Xue Bao/Journal of Software, 2021, 32(4): 1082−1115 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6169.htm [doi: 10.13328/j.cnki.jos.006169]

[12] Rishabh A, Schuurmans D, Norouzi M. An optimistic perspective on offline reinforcement learning. In: Proc. of the 37th Int'l Conf. on Machine Learning. 2020. 104−114.

[13] Botvinick M, Ritter S, Wang J, *et al*. Reinforcement learning, fast and slow. Trends in Cognitive Sciences, 2019, 23(5): 408−422.

[14] Huang ZG, Liu Q, Zhang LH, *et al*. Research and development on deep hierarchical reinforcement learning. Ruan Jian Xue Bao/Journal of Software, 2023, 34(2): 733−760 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6706.htm [doi: 10.13328/j.cnki.jos.006706]

[15] Sun T, He Z, Qian H, *et al*. BBTv2: Towards a gradient-free future with large language models. In: Proc. of the 2022 Conf. on Empirical Methods in Natural Language Processing. 2022. 3916−3930.

[16] Sun T, Shao Y, Qian H, *et al*. Black-box tuning for language-model-as-a-service. In: Proc. of the 39th Int'l Conf. on Machine Learning. 2022. 20841−20855.

[17] Sanderson K. GPT-4 is here: What scientists think. Nature, 2023, 615(7954): 773.

[18] Liu P, Yuan W, Fu J, *et al*. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. ACM Computing Surveys, 2023, 55(9): 195,1−195,35.

[19] Lester B, Al-Rfou R, Constant N. The power of scale for parameter-efficient prompt tuning. In: Proc. of the 2021 Conf. on Empirical Methods in Natural Language Processing. 2021. 3045−3059.

[20] Hu S, Ding N, Wang H, *et al*. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. In: Proc. of the 60th Annual Meeting of the Association for Computational Linguistics. 2022. 2225−2240.

[21] Ke G, Meng Q, Finley T, *et al*. LightGBM: A highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems 30. 2017. 3146−3154.

[22] Conn AR, Scheinberg K, Vicente L. Introduction to derivative-free optimization. Philadelphia: society for industrial and applied mathematics, 2009. https://epubs.siam.org/terms-privacy

[23] Rios L, Sahinidis N. Derivative-free optimization: A review of algorithms and comparison of software implementations. Journal of Global Optimization, 2013, 56(3): 1247−1293.

[24] Zhou ZH, Yu Y, Qian C. Evolutionary Learning: Advances in Theories and Algorithms. Springer, 2019.

[25] Li S, Kirby R, Zhe S. Batch multi-fidelity Bayesian optimization with deep auto-regressive networks. In: Advances in Neural Information Processing Systems 34. 2021. 25463−25475.

[26] Poloczek M, Wang J, Frazier P. Multi-information source optimization. In: Advances in Neural Information Processing Systems 30. 2017. 4288−4298.

[27] Hu YQ, Yu Y, Tu W, *et al*. Multi-fidelity automatic hyper-parameter tuning via transfer series expansion. In: Proc. of the 33rd AAAI Conf. on Artificial Intelligence. 2019. 3846−3853.

[28] Nemirovski A, Juditsky A, Lan G, *et al*. Robust stochastic approximation approach to stochastic programming. SIAM Journal on Optimization, 2009, 19(4): 1574−1609.

[29] Bottou L. Large-scale machine learning with stochastic gradient descent. In: Proc. of the 19th Int'l Conf. on Computational Statistics. 2010. 177−186.

[30] James B, Bardenet R, Bengio Y, *et al*. Algorithms for hyper-parameter optimization. In: Advances in Neural Information Processing Systems 24. 2011. 2546−2554.

[31] Wu J, Toscano-Palmerin S, Frazier PI, *et al*. Practical multi-fidelity bayesian optimization for hyperparameter tuning. In: Proc. of the 35th Conf. on Uncertainty in Artificial Intelligence. 2019. 788−798.

[32] Qian C, Yu Y, Zhou ZH. Analyzing evolutionary optimization in noisy environments. Evolutionary Computation, 2018, 26(1): 1−41.

[33] Qian C, Bian C, Yu Y, *et al*. Analysis of noisy evolutionary optimization when sampling fails. Algorithmica, 2021, 83(4): 940−975.

[34] Kandasamy K, Dasarathy G, Schneider J, *et al*. Multi-fidelity Bayesian optimisation with continuous approximations. In: Proc. of the 34th Int'l Conf. on Machine Learning. 2017. 1799−1808.

[35] Falkner S, Klein A, Hutter F. BOHB: Robust and efficient hyperparameter optimization at scale. In: Proc. of the 35th Int'l Conf. on Machine Learning. 2018. 1436−1445.

[36] Li L, Jamieson K, Rostamizadeh A, *et al*. A system for massively parallel hyperparameter tuning. In: Proc. of the Machine Learning and Systems 2020. 2020.

[37] Li S, Xing W, Kirby R, *et al*. Multi-fidelity Bayesian optimization via deep neural networks. In: Advances in Neural Information Processing Systems 33. 2020.

[38] Belakaria S, Deshwal A, Doppa J. Multi-fidelity multi-objective Bayesian optimization: An output space entropy search approach. In: Proc. of the 34th AAAI Conf. on Artificial Intelligence. 2020. 10035−10043.

[39] Takeno S, Fukuoka H, Tsukada Y, *et al*. Takeuchi I, Karasuyama M. Multi-fidelity Bayesian optimization with max-value entropy search and its parallelization. In: Proc. of the 37th Int'l Conf. on Machine Learning, 2020. 9334−9345.

[40] Roux N, Schmidt M, Bach F. A stochastic gradient method with an exponential convergence rate for finite training sets. In: Advances in Neural Information Processing Systems 25. 2012. 2672−2680.

[41] Qian C. Distributed Pareto optimization for large-scale noisy subset selection. IEEE Trans. on Evolutionary Computation, 2020, 24(4): 694−707.

[42] Liu Y, Hu YQ, Qian H, *et al*. ZOOpt: A toolbox for derivative-free optimization. Science China Information Sciences, 2022, 65(10).

[43] Johnson R, Zhang T. Accelerating stochastic gradient descent using predictive variance reduction. In: Advances in Neural Information Processing Systems 26. 2013. 315−323.

[44] Defazio A, Bach F, Lacoste-Julien S. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In: Advances in Neural Information Processing Systems 27. 2014. 1646−1654.

[45] Hansen N, Müller S, Koumoutsakos P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evolutionary Computation, 2003, 11(1): 1−18.

[46] Bäck T, Schwefel H. An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation, 1993, 1(1): 1−23.

[47] Jin Y, Wang H, Chugh T, *et al*. Data-driven evolutionary optimization: An overview and case studies. IEEE Trans. on Evolutionary Computation, 2018, 23(2): 442−458.

[48] Shahriari B, Swersky K, Wang Z, *et al*. Taking the human out of the loop: A review of Bayesian optimization. Proc. of the IEEE, 2016, 104(1): 148−175.

[49] Garnett R. Bayesian Optimization. Cambridge University Press, 2023.

[50] Bartlett P, Gabillon V, Valko M. A simple parameter-free and adaptive approach to optimization under a minimal local smoothness assumption. In: Proc. of the 2019 Algorithmic Learning Theory. 2019. 184−206.

[51] Munos R. From bandits to Monte-Carlo tree search: The optimistic principle applied to optimization and planning. Foundations and Trends in Machine Learning, 2014, 7(1): 1−129.

[52] Valko M, Carpentier A, Munos R. Stochastic simultaneous optimistic optimization. In: Proc. of the 30th Int'l Conf. on Machine Learning 2013. 2013. 19−27.

[53] Jones D, Perttunen C, Stuckman B. Lipschitzian optimization without the Lipschitz constant. Journal of Optimization Theory and Applications, 1993, 79: 157−181.

[54] Boer P, Kroese D, Mannor S, *et al*. A tutorial on the cross-entropy method. Annals of Operations Research, 2005, 134(1).

[55] Hansen N, Ostermeier A. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In: Proc. of the 1996 IEEE Int'l Conf. on Evolutionary Computation. Nayoya University, 1996. 312−317.

[56] Hansen N, Ostermeier A. Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation, 2001, 9(2): 159−195.

[57] Li L, Jamieson K, DeSalvo G, *et al*. Hyperband: A novel bandit-based approach to hyperparameter optimization. Journal of Machine Learning Research, 2017, 18(185): 1−52.

[58] Lindauer M, Eggensperger K, Feurer M, *et al*. SMAC3: A versatile Bayesian optimization package for hyperparameter optimization. Journal of Machine Learning Research, 2022, 23(54): 1−9.

[59] González J, Dai Z, Hennig P, *et al*. Batch Bayesian optimization via local penalization. In: Proc. of the 19th Int'l Conf. on Artificial Intelligence and Statistics. 2016. 648−657.

[60] Brown T, Mann B, Ryder N, *et al*. Language models are few-shot learners. In: Advances in Neural Information Processing Systems 33. 2020.

[61] Qian C, Yu Y, Tang K, *et al*. On the effectiveness of sampling for evolutionary optimization in noisy environments. Evolutionary Computation, 2018, 26(2): 237–267.

附中文参考文献:

[5] 申朕, 崔超然, 董桂鑫, 等. 基于深度多任务学习的图像美感与情感联合预测研究. 软件学报, 2023, 34(5): 2494–2506. http://www.jos.org.cn/1000-9825/6487.htm [doi: 10.13328/j.cnki.jos.006487]

[11] 王乃钰, 叶育鑫, 刘露, 等. 基于深度学习的语言模型研究进展. 软件学报, 2021, 32(4): 1082–1115. http://www.jos.org.cn/1000-9825/6169.htm [doi: 10. 13328/j.cnki.jos.006169]

[14] 黄志刚, 刘全, 张立华, 等. 深度分层强化学习研究与发展. 软件学报, 2023, 34(2): 733–760. http://www.jos.org.cn/1000-9825/6706.htm [doi: 10.13328/j.cnki.jos.006706]

钱鸿(1991－), 男, 博士, 副教授, CCF 专业会员, 主要研究领域为机器学习, 无梯度优化, 演化学习, 智能教育.

舒翔(2000－), 男, 硕士生, CCF 学生会员, 主要研究领域为机器学习, 无梯度优化及其在大语言模型中的应用.

孙天祥(1997－), 男, 博士生, 主要研究领域为自然语言处理.

邱锡鹏(1983－), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为自然语言处理.

周爱民(1978－), 男, 博士, 教授, 博士生导师, CCF 专业会员, 主要研究领域为演化优化与学习, 智能教育.