LiBOG: Lifelong Learning for Black-Box Optimizer Generation

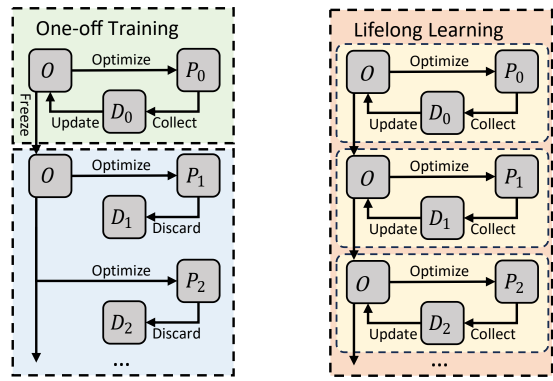
Jiyuan Pei[1] , **Yi Mei**[1*] , **Jialin Liu**[2] and **Mengjie Zhang**[1]
[1]Victoria University of Wellington
[2]Lingnan University
jiyuan.pei@vuw.ac.nz, yi.mei@ecs.vuw.ac.nz, jialin.liu@ln.edu.hk, mengjie.zhang@ecs.vuw.ac.nz

Abstract

Meta-Black-Box Optimization (MetaBBO) garners attention due to its success in automating the configuration and generation of black-box optimizers, significantly reducing the human effort required for optimizer design and discovering optimizers with higher performance than classic human-designed optimizers. However, existing MetaBBO methods conduct one-off training under the assumption that a stationary problem distribution with extensive and representative training problem samples is pre-available. This assumption is often impractical in real-world scenarios, where diverse problems following shifting distribution continually arise. Consequently, there is a pressing need for methods that can continuously learn from new problems encountered on-the-fly and progressively enhance their capabilities. In this work, we explore a novel paradigm of lifelong learning in MetaBBO and introduce LiBOG, a novel approach designed to learn from sequentially encountered problems and generate high-performance optimizers for Black-Box Optimization (BBO). LiBOG consolidates knowledge both across tasks and within tasks to mitigate catastrophic forgetting. Extensive experiments demonstrate LiBOG's effectiveness in learning to generate high-performance optimizers in a lifelong learning manner, addressing catastrophic forgetting while maintaining plasticity to learn new tasks.

1 Introduction

Black-Box Optimization (BBO) solves optimization problems by using only the output of the objective function, without requiring knowledge of its internal structure. It is commonly used in tasks with complex structures like metal manufacturing [Bi *et al.*, 2022], protein docking [Tsaban *et al.*, 2022], and hyper-parameter tuning of learning algorithms [Lechner *et al.*, 2022; Gu *et al.*, 2021]. Typically, designing and tuning BBO optimizers is labor-intensive and requires expertise. Meta-Black-Box Optimization (MetaBBO)



(a) MetaBBO (b) Lifelong Learning MetaBBO

Figure 1: Existing MetaBBO methods train the BBO optimizer O with data D_0 obtained from the problem distribution P_0 available during the training phase, then freeze the model after training to solve the newly arising problems from distributions $(P_1, P_2, \dots)$. New data obtained from subsequent problems are discarded. In contrast, lifelong learning MetaBBO utilizes the data obtained from each encountered problem to update the optimizer continually.

automates this process with machine learning, greatly reducing manual effort. By training on BBO problems sampled from a given distribution, MetaBBO learns to directly propose solutions [Chen *et al.*, 2017; TV *et al.*, 2020] to some given problems, configure optimizers with expert-derived solution update rules [Sharma *et al.*, 2019; Yi *et al.*, 2023; Lu *et al.*, 2020; Chaybouti *et al.*, 2022], or generate optimizers by automatically generating solution update rules [Chen *et al.*, 2024]. Existing MetaBBO methods rely on one-off training, where the optimizer is trained on problems sampled from a pre-defined fixed distribution, and then is directly applied to test problems (e.g., learned parameters are frozen) [Lu *et al.*, 2020; Tang and Yao, 2024], as shown in Fig 1a.

In real-world optimization scenarios, the distribution of optimization problems commonly varies over time, leading to the emergence of new problems with different but related characteristics. For example, the scheduling problem faced by a manufacturer may vary from season to season due to dynamic factors such as demands and resource supplies. Directly applying the pre-trained optimizer to solve new problems could be ineffective [Liu *et al.*, 2023; Pei *et al.*, 2024;

[*]Yi Mei is the corresponding author.

Tang and Yao, 2024; Yang *et al.*, 2025]. To address this issue, the traditional way in practice typically involves modifications of the optimizer by human experts when problem characteristics change significantly [Hoos, 2012]. However, existing MetaBBO methods encounter limitations in this adaptability. MetaBBO approaches capable of continuously learning from emerging problems to enhance their ability in a lifelong learning manner, as shown in Fig 1b, are desired.

Lifelong learning [Wang *et al.*, 2024] is a natural paradigm to continually learn/adapt BBO solvers when facing changing problem distribution over time. Although lifelong learning has achieved great success in common machine learning tasks [Kirkpatrick *et al.*, 2017; Wang *et al.*, 2020], to the best of our knowledge, no study has investigated it in the context of MetaBBO. In particular, a major challenge in lifelong learning is catastrophic forgetting [Khetarpal *et al.*, 2022; Wang *et al.*, 2024], i.e., models' performance on previous data distributions reduces significantly after training on a new distribution. Mostly studied, catastrophic forgetting usually occurs between tasks as the data from different tasks exhibit significant differences. However, catastrophic forgetting may also occur when learning a single task, especially in reinforcement learning (RL) [Igl *et al.*, 2021; Lan *et al.*, 2023; Zhang *et al.*, 2023].

In this paper, we focus on the unexplored paradigm of lifelong learning MetaBBO, with the aim of training a single model capable of generating high-performance optimizers for any problem drawn from previously learned distributions, in scenarios where problem distributions arrive sequentially over time. To achieve it, we present **LiBOG**, a novel approach with **li**felong learning for **b**lack-box **o**ptimizer **g**eneration. LiBOG takes the problems following the same distribution as a task and learns from a sequence of different tasks with lifelong RL. LiBOG features an inter-task consolidation process to preserve the learned knowledge of previous tasks. We further propose a novel intra-task consolidation method, named elite behavior consolidation, to address forgetting in one single task which could significantly impact learning performance in the context of MetaBBO,

We verify the effectiveness of LiBOG in generating high-performance optimizers for solving problems from all learned distributions through extensive experimental studies. The results demonstrate that LiBOG not only significantly mitigates catastrophic forgetting but also exhibits the capability to learn general knowledge from sequentially arriving problem distributions. Further sensitivity analysis and ablation study verify the stability of LiBOG and the contribution of each component to LiBOG's overall performance.

Our main contributions are as follows. (i) We propose the paradigm of lifelong learning for BBO optimizer generation, which is the first study in this area, to the best of our knowledge. (ii) We present elite behavior consolidation, a novel intra-task consolidation approach to address catastrophic forgetting and improve plasticity, and based on it, LiBOG, a novel method of lifelong learning to BBO optimizer generation. (iii) We verify the effectiveness of the proposed methods with extensive experimental studies, together with detailed analysis and in-depth discussion.

2 Background

2.1 MetaBBO

Black-box optimization relies exclusively on the output of the objective function to guide the optimization process [Audet and Kokkolaras, 2016]. A typical BBO optimizer operates iteratively, wherein incumbent solutions are evaluated based on their objective values, and modified by solution updating rules with the aim of achieving superior performance [Hussain *et al.*, 2018]. Examples of BBO optimizers include evolution strategy [Beyer and Schwefel, 2002], differential evolution (DE) [Storn and Price, 1997], Bayesian optimization [Wang *et al.*, 2023], and simulated annealing [Kirkpatrick *et al.*, 1983]. Typically, BBO optimizers are characterized by a wide array of tunable parameters, which significantly influence their performance. The design and configuration of these parameters are often computationally expensive and labor-intensive, requiring substantial expertise and iterative experimentation.

Meta-Black-Box Optimization (MetaBBO) is an emerging framework that leverages machine learning techniques to automate the design of black-box optimizers, significantly reducing the reliance on manual expertise and effort [Chen *et al.*, 2017; Gomes *et al.*, 2021; Ma *et al.*, 2023]. Some MetaBBO methods focus on training end-to-end models to directly generate new solutions [Chen *et al.*, 2017; TV *et al.*, 2020]. Despite their success, end-to-end methods often suffer from poor generalization and limited interpretability [Liu *et al.*, 2023]. Some other methods learn to configure human-designed optimizers, including parameter tuning and selecting solution updating rules [Sharma *et al.*, 2019; Yi *et al.*, 2023; Lu *et al.*, 2020; Chaybouti *et al.*, 2022]. However, the corresponding methods are inherently limited by the dependence of pre-existing human-crafted solution updating rules. In contrast, the recent study [Chen *et al.*, 2024] leverages symbolic equation learning and deep reinforcement learning to construct updating rules and outperforms SOTA expert-designed methods with human-crafted rules. Following the conventional machine learning paradigm, existing MetaBBO methods are typically trained on problems sampled from a pre-defined distribution, with the optimizer's parameters fixed upon completion of training. These trained models are then evaluated on separate test sets to assess their performance. To achieve good performance across a broader range of problems, current MetaBBO approaches typically involve a large number of diverse problems in the one-off training.

2.2 Lifelong Learning

Lifelong learning, also known as continual learning, requires learners to learn a sequence of different tasks to progressively enhance their capabilities on the tasks, just like the way that humans learn in their whole life [Thrun, 1998]. In lifelong learning, each task is represented by a distinct data distribution [Wang *et al.*, 2024]. Lifelong learning primarily aims to efficiently train on the data distribution of the current task while minimizing reliance on data from the distributions of previously encountered tasks. The key challenge in lifelong learning is to balance plasticity, the ability to efficiently acquire knowledge from new tasks, with stability, the capacity to preserve learned knowledge of previous tasks. The degradation of performance on learned tasks after training on new data with different distributions is known as *catastrophic forgetting*. This phenomenon is predominantly observed between tasks due to significant differences in their data.

Lifelong learning in the RL context refers to the problem of how an agent learns from a series of different environments to make good decisions on each of them [Abel *et al.*, 2018; Khetarpal *et al.*, 2022]. Each environment, commonly defined by a Markov decision process (MDP), corresponds to a task. Specifically in RL, catastrophic forgetting can occur not only between tasks but also during the learning of a single task [Igl *et al.*, 2021]. Training data, i.e., experiences, are typically generated iteratively through interactions between the policy model and the given environment. These experiences exhibit temporal correlations, and the distribution of the collected experiences changes with the policy updating, contributing to catastrophic forgetting [Igl *et al.*, 2021]. Typically, it is addressed by extensive resampling and large experience memory, significantly increasing the storage requirements and computational costs [Lan *et al.*, 2023].

3 LiBOG

A typical BBO optimizer improves the objective value by iteratively modifying incumbent solutions with solution updating rules, inherently forming a sequential decision-making process [Handoko *et al.*, 2014; Chaybouti *et al.*, 2022]. Solution updating rules play a crucial role in BBO optimizers [Sharma *et al.*, 2019; Yi *et al.*, 2023; Pei *et al.*, 2025]. Inspired by the success of recent work [Lu *et al.*, 2020; Chen *et al.*, 2024], we focus on lifelong learning for constructing solution updating rules. Specifically, we take each sequentially arrived problem distribution as a distinct learning task and train a MetaBBO model sequentially on those tasks, with the aim of automatically constructing solution updating rules by the model to effectively solve problems from any learned distribution (i.e., tasks).

To achieve this, we propose LiBOG, which consists of three levels of design. Firstly, LiBOG models the lifelong learning process of MetaBBO as a non-stationary MDP [Khetarpal *et al.*, 2022]. The non-stationary MDP is formed by a sequence of stationary MDPs, each representing a task to learn (c.f., Section 3.1). It facilitates the utilization of advanced RL methods. Based on the formulation, the second level (c.f., Section 3.2) focuses on learning to construct solution updating rules within a single task, without addressing catastrophic forgetting. We construct this level following SYMBOL [Chen *et al.*, 2024], the SOTA MetaBBO method for solution updating rule construction. Finally, at the highest level, LiBOG introduces two consolidation mechanisms to address inter-task (c.f., Section 3.3) and intra-task (c.f., Section 3.4) catastrophic forgetting, respectively. Figure 2 demonstrates the overall lifelong learning process of LiBOG. A detailed pseudocode of LiBOG's learning process can be found in the supplementary material.

3.1 Formulation of Lifelong Learning MetaBBO

A BBO problem distribution P, i.e., a task, is represented as an MDP, represented by the tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \mathcal{H}, \rho \rangle$. $\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}$ are the optimization state space, solution updating rule space, reward function, and state transition function, respectively. $\mathcal{H}$ represents the maximal number of optimization iterations. ρ is the distribution of the initial state, defined by the initial solutions. To solve a sampled problem of the task, an optimizer first generates an initial solution set, forming the initial state following ρ. Then the optimizer iteratively observes the current state $s \in \mathcal{S}$, and constructs an updating rule a from $\mathcal{A}$ to update the incumbent solutions, which transits s to the next state $s' \in \mathcal{S}$ following $\mathcal{P}$, and provides a reward value r based on $\mathcal{R}$. The iteration will continue for $\mathcal{H}$ times, and then the best-so-far solution will be output.

Specifically, LiBOG represents a state s by a vector of fitness landscape analysis (FLA) metrics [Malan and Engelbrecht, 2013], including the distances of decision variable values and objective values of incumbent solutions. Each FLA metrics are normalized so that all tasks share the same state space. Details about state representation can be found in the supplementary material. RL-based MetaBBO methods have demonstrated that those FLA metrics are effective in representing the characteristics of optimization states [Sharma *et al.*, 2019; Lu *et al.*, 2020; Yi *et al.*, 2023]. An action a is a solution updating rule, represented by a tree-structure symbolic equation. Section 3.2 details the construction process of a solution updating rule, i.e., an action.

The reward function in [Chen *et al.*, 2024] is used:

$$\mathcal{R}(k) = \frac{f(x^{*,(k)}) - f^{opt}}{f(x^{*,(0)}) - f^{opt}} + \frac{d(X_k, X'_k)}{x_{ub} - x_{lb}}, \tag{1}$$

where $f(x^{*,(k)})$ is the best-so-far objective value within optimization iterations $[0, k]$, and f^{opt} is the optimal objective value, or the known best objective if the optimum is unknown, of the problem. X_k and X'_k are the solution population generated by the optimizer from the MetaBBO and by a given manually designed optimizer, respectively, in kth iteration, based on the same X_{k-1}. $d(X_k, X'_k) = \max_{x \in X_k} \min_{x' \in X'_k} (\|x - x'\|_2)$ is the distance measure of two solution population. x_{ub} and x_{lb} are the upper and lower bounds of variable values.

Based on the above formulation, we further model the lifelong learning process of MetaBBO as a discrete non-stationary MDP with piecewise non-stationary function [Khetarpal *et al.*, 2022], represented as $\mathbf{M} = \{\mathcal{M}_0, \mathcal{M}_1, \dots\}$. For a specific time point, only the $\mathcal{M}_i$, corresponding to the current problem distribution P_i, is available for the optimizer to interact. We assume the identities and boundaries of tasks are known, which aligns with many real-world situations. For example, managers of a manufacturing company can identify the current season in scheduling problems, where seasonal changes in material prices and product demands lead to changes in problem distribution.

Different tasks share the same state space $\mathcal{S}$ and action space $\mathcal{A}$. We assume all problems share the same optimization iteration budget, therefore, $\mathcal{H}$ is also identical for all tasks. Due to the different fitness landscapes of different problem distributions, different tasks have different state transition functions and different distributions of initial states. Besides, the reward values obtained from different tasks given the same state and action will also be different, though the formulation of the reward function is the same across tasks.

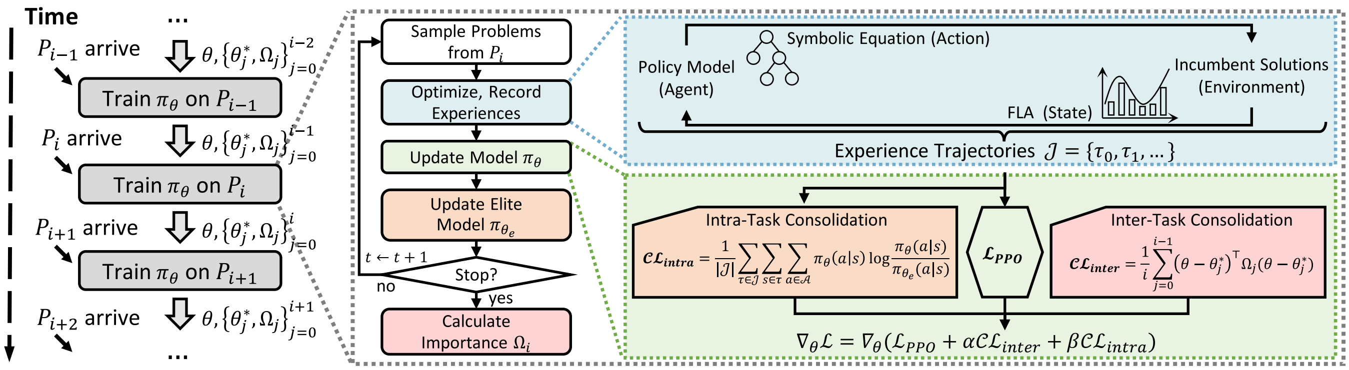


Figure 2: Illustration of LiBOG, with different problem distributions $P_{i-1}, P_i, \dots$ sequentially arrive.

3.2 Symbolic Updating Rule Construction

Solution updating rules can be formulated as equations that calculate the variable values of the new solution(s) based on the incumbent solutions (as real-value vectors). For example, the *DE/best/1* rule of DE, a classic human-designed BBO optimizer, can be formulated as $x' = x_{best} + F(x_1 - x_2)$, where x' is the new solution, x_{best} is the best incumbent solution, F is a pre-defined parameter, x_1 and x_2 are two randomly selected incumbent solutions. Symbolic equation learning facilitates the construction of rules as tree-structure symbolic equations [Zheng *et al.*, 2022; Chen *et al.*, 2024; Chen *et al.*, 2023], where terminal nodes are operands like x_{best} and internal nodes represent operators like $+$.

Following [Chen *et al.*, 2024], we apply a long short-term memory (LSTM) network as the policy model to generate solution updating rules, i.e., actions. For each state, the LSTM predicts multiple times sequentially to construct a tree. For each prediction, the model takes FLA metrics, i,e., the state, and the vectorized tree embedding of the current tree as input, and outputs the next node to be added into the tree. Detailed settings of the tree construction can be found in the supplementary material. Proximal policy optimization (PPO) [Schulman *et al.*, 2017] is used in training, given its success in training to generate BBO optimizers that outperform those crafted by human experts [Chen *et al.*, 2024].

3.3 Inter-Task Consolidation

Preventing significant deviation from the parameters optimized for earlier tasks has the potential to preserve learned knowledge. It can be achieved by applying L2 regularization on the model's parameters. However, equally regularizing all parameters leads to the loss of plasticity [Kirkpatrick *et al.*, 2017]. In contrast, LiBOG adaptively adjusts the regularization strength for each parameter based on its importance to previous tasks, based on elastic weight consolidation (EWC) method [Kirkpatrick *et al.*, 2017]. Parameters deemed more critical for earlier tasks are preserved to a greater extent, while less important parameters are allowed larger updates, providing more flexibility for learning new tasks. The importance of parameters for the ith task is calculated as:

$$\Omega_i^\theta = \frac{1}{|\mathcal{J}|}\sum_{\tau\in\mathcal{J}}\left(\frac{1}{|\tau|}\sum_{k=0}^{|\tau|-1}\nabla_\theta \ell_{\theta,k}\nabla_\theta \ell_{\theta,k}^\top\right), \tag{2}$$

where $\mathcal{J} = \{\tau_0, \tau_1, \dots\}$ is a given set of experience trajectories, $\tau = \{(s_k, a_k, r_k)\}_{k=0}^{\mathcal{H}}$ is a trajectory obtained by model π_θ from ith task, i.e., records of all decisions in an optimization process, and $\ell_{\theta,k} = \log \pi_\theta(a_k|s_k)$ corresponds the action probability generated by model π_θ given $(s_k, a_k) \in \tau$.

Specifically, after completing training on each task, LiBOG records the parameter values θ^* and calculates the importance Ω to this task with the trajectories obtained for the last training epoch. During training on a new task, an inter-task consolidation term $\mathcal{CL}_{inter}$ is incorporated into the loss function, calculated as:

$$\mathcal{CL}_{inter}(\theta) = \frac{1}{i}\sum_{j=0}^{i-1}(\theta - \theta_j^*)^\top \Omega_j (\theta - \theta_j^*). \tag{3}$$

A smaller $\mathcal{CL}_{inter}$ indicates the parameters important to previous tasks are similar to the best value obtained on those tasks. Consolidation in parameter level is relatively storage-efficient, as it only requires saving the parameters and importance with the space complexity as $O(|\theta| \cdot I)$ rather than retaining any task-specific experiences, where $|\theta|$ is the number of model parameters and I is the total number of tasks.

3.4 Intra-Task Consolidation

Typically, the distribution of obtained experience in RL could shift due to changes in policy caused by the model parameter updating. This results in intra-task catastrophic forgetting, which can hinder model convergence and reduce generalizability [Igl *et al.*, 2021]. Common approaches in RL, such as extensive resampling and large experience memory, address this issue but significantly increase storage requirements and computational costs [Lan *et al.*, 2023]. Given the inherently computationally expensive nature of BBO, such brute-force methods become less practical.

Existing studies about intra-task forgetting focus on single-task scenarios [Ghiassian *et al.*, 2020; Pan *et al.*, 2021; Zhang *et al.*, 2023], with limited attention to its effects in multi-task settings. We focus on lifelong learning with multiple tasks. Inter-task and intra-task forgetting could interact, leading to more pronounced effects on the training process. Additionally, the substantial stochasticity inherent in the BBO process introduces significant fluctuations in obtained experiences, potentially exacerbating the impacts.

To address the challenges, we propose the elite behavior consolidation (EBC) method. EBC maintains an elite model, parameterized by θ_e, as a reference and updates it as a copy of the new model whenever the new model has better performance than the elite model. In the context of MetaBBO, the model that obtain a better average final objective value on the current tasks is defined as better. During each model update, EBC regularizes the current model's behavior to align with the elite model. Specifically, EBC introduces the KL divergence between the action probability distributions of the current model and the elite model for given states as a loss term $\mathcal{CL}_{intra}$, calculated as:

$$\mathcal{CL}_{intra}(\theta) = \frac{1}{|\mathcal{J}|} \sum_{\tau \in \mathcal{J}} \sum_{s \in \tau} \sum_{a \in \mathcal{A}} \pi_\theta(a|s) \log \frac{\pi_\theta(a|s)}{\pi_{\theta_e}(a|s)}. \quad (4)$$

When training on one task is finished, the record of θ_e will be discarded. Compared to regularizing model parameters, EBC directly regularizes model behavior, effectively addressing catastrophic forgetting caused by changes in behavioral patterns. It may also better accommodate exploration within the model parameter space, as different sets of model parameters can correspond to similar, near-optimal policy behaviors.

In summary, to maximize the overall reward $\mathcal{R}$ while addressing both inter-task and intra-task forgetting, LiBOG updates a model on each task with the following loss function.

$$\mathcal{L}(\cdot) = \mathcal{L}_{PPO}(\cdot) + \alpha \cdot \mathcal{CL}_{inter}(\cdot) + \beta \cdot \mathcal{CL}_{intra}(\cdot), \quad (5)$$

where $\mathcal{L}_{PPO}$ is the loss function of PPO algorithm to maximize $\mathcal{R}$, α and β are two pre-defined hyper-parameters balancing the two consolidation terms.

4 Experiments

We focus on the MetaBBO scenario where various problem distributions sequentially arise. Through experimental studies[1], we aim to answer the following research questions.

- **Optimization effectiveness:** Does LiBOG demonstrate superior optimization ability on learned problems compared to SOTA human-designed optimizers and MetaBBO methods without lifelong learning?
- **Addressing catastrophic forgetting:** What are the effects of catastrophic forgetting in such scenarios, and are the two consolidation mechanisms in LiBOG effective in mitigating this issue?
- **Hyper-parameter sensitivity:** How sensitive is LiBOG to the weights of consolidation terms, i.e., α and β?

Problem Distributions. The training dataset is constructed from the widely studied IEEE CEC Numerical Optimization Competition Benchmark [Mohamed *et al.*, 2021]. The benchmark is known for its extensive use in the optimization research community, serving for comparative studies and fostering advancements in BBO methods. Four function categories, namely, uni-modal, basic, hybrid, and composition, are provided. Different categories own different properties and landscape features, such as uni-modal and multi-modal landscapes, (non-)separability, and (a)symmetry. Each function can be configured by the dimension, searching space range, function offset, and function rotation, forming a specific problem. We set the dimension and searching space to 10 and $[-100, 100]^{10}$ for all the functions. Then, we form each category as a task by introducing a distribution of offset $z \sim U[-80, 80]^{10}$ and a distribution of rotation uniformly distributed in $\mathbb{R}^{10\times 10}$ for each function of the category. When sampling a problem from a task, all functions of the corresponding category are selected with the same probability. In summary, four tasks $\{P^U, P^B, P^H, P^C\}$ are constructed, corresponding to uni-modal, basic, hybrid, and composition categories, respectively. We randomly generated three different task orders aiming to eliminate the influence of specific task orders on the experimental results. Details about the training dataset and the task orders can be found in the supplementary material.

Baselines. To the best of our knowledge, there is no existing work on lifelong learning MetaBBO, as current MetaBBO methods focus on learning for a single task. We compare LiBOG with the SOTA human-designed BBO optimizer MadDE [Biswas *et al.*, 2021], which is one of the winners of the CEC competition, and SYMBOL [Chen *et al.*, 2024], a SOTA MetaBBO method for single-task learning. For SYMBOL, we applied two strategies to learn across multiple consecutive tasks: (i) randomly generating a model each time a new task appears and then training the model on the new task, denoted as *restart*, and (ii) directly updating the obtained model on a new task, denoted as *fine-tuning*. Additionally, we compared with a baseline method assuming all functions are available at once and sampling problems from all of them to train SYMBOL in each epoch, denoted as *all-task*.

Hyper-parameter Setting and Performance Evaluation. For LiBOG, *restart* and *fine-tuning*, the models are trained on each task for 100 epochs equally. Following [Chen *et al.*, 2024], *all-task* trains the model on all functions simultaneously for 100 epochs. Both SYMBOL and LiBOG use MadDE as the guide optimizer for reward calculation. For LiBOG, the values of α and β are set to 1 based on the rule of thumb. To test the optimization performance of an optimizer on a task, we sample 32 problems with the corresponding distribution. We run the optimizer to solve each of the problems, and take the average output objective value over the problems as the test performance on the task. The objective values are normalized between 0 and 1. A larger objective value indicates better optimization performance. More settings about hyper-parameters and the normalization method can be found in the supplementary material.

4.1 Effectiveness Evaluation

To verify the effectiveness of LiBOG in generating high-performance optimizers, we tested and compared the performance of LiBOG and all baseline methods on each task. For LiBOG and *fine-tuning*, the model obtained after training on the final task was used for testing. For *restart*, the model trained on a task is used for testing on that task. For *all-task*, the model obtained after the one-off training was tested. MadDE does not involve a learning process and is directly

[1] Our code is available in https://github.com/PeiJY/LiBOG.

Method	Order 0					Order 1					Order 2				
	P_0 (P^U)	P_1 (P^B)	P_2 (P^H)	P_3 (P^C)	avg.	P_0 (P^C)	P_1 (P^U)	P_2 (P^B)	P_3 (P^H)	avg.	P_0 (P^U)	P_1 (P^C)	P_2 (P^H)	P_3 (P^B)	avg.
LiBOG	**1**	**1**	**1**	**1**	**1**	**1**	**1**	2	2	**1.5**	2	**1**	**1**	2	**1.5**
restart	4	4	2	2	3	4	4	**1**	**1**	2.5	4	5	2	**1**	3
fine-tuning	2	3	3	3	2.75	3	3	4	4	3.5	3	3	3	4	3.25
all-task	5	2	5	5	4.25	5	5	3	5	4.5	5	4	5	3	4.25
MadDE	3	5	4	4	4	2	2	5	3	3	**1**	2	4	5	3

Table 1: The rank of test results on each task. Smaller ranks indicate better performance.

tested on all functions. Each learning method underwent 10 independent learning runs for each task order. The model obtained in each run is tested, and the average test results over the 10 runs are compared. Table 1 presents the rank of each method. More detailed results can be found in the supplementary material.

LiBOG outperforms all baseline methods for each of the three task orders. Specifically, LiBOG's superior performance compared to *fine-tuning* demonstrates the effectiveness of LiBOG's two consolidation methods in addressing catastrophic forgetting and maintaining good plasticity. *Restart* achieved the best performance in some cases, as expected due to its task-specific training and lack of influence from task distribution differences. However, LiBOG is ranked in first place for eight of the twelve cases over the three orders, indicating that, in general, LiBOG effectively transfers knowledge of previously learned tasks to enhance the learning of subsequent tasks without forgetting, though the problem distributions are different.

The rank stability of LiBOG across different task orders (average ranks of 1, 1.5, and 1.5) demonstrates its robustness to changes in task orders.

4.2 Addressing Forgetting

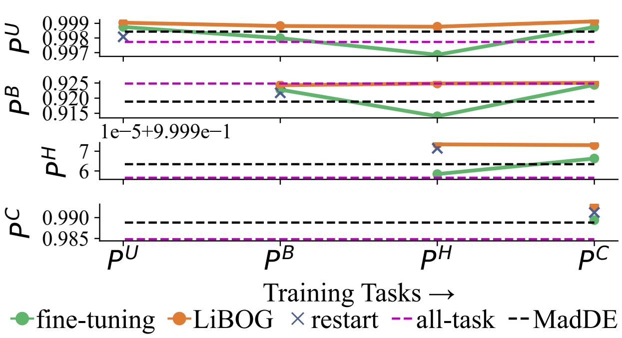


Figure 3: Test performance on each learned task during the lifelong learning process of task order 0. In *fine-tuning*, catastrophic forgetting of previous tasks is significant, but is mild in LiBOG.

We analyzed the impact of catastrophic forgetting in the studied lifelong learning MetaBBO scenarios, and the effectiveness of LiBOG in addressing catastrophic forgetting. Figure 3 shows the test performance on previously learned tasks after the training of each new task of one task order. Though there is no lifelong learning process in MadDE, *all-task* and *restart*, we add them for reference. Results of other orders demonstrate a similar pattern, and can be found in the supplementary material.

Fine-tuning shows a significant performance drop on the first task P_0 (i.e., uni-modal functions P^U) and the second task P_1 (i.e., basic functions P^B) after training on the subsequent two and one tasks, respectively, performing worse than *all-task* and MadDE, indicating a significant catastrophic forgetting. In contrast, LiBOG experiences very small performance degradation after learning new tasks, suggesting that forgetting in LiBOG is mild. In some cases, training the model on a task (e.g., P_3 (P_C) in Figure 3) was observed to enhance its performance on previously learned tasks. This is perhaps due to the inherent function similarity between tasks. This explains why LiBOG outperforms others, as it effectively utilizes similar tasks to better learn shared knowledge while retaining knowledge from dissimilar tasks without forgetting.

To further analyze the training process within each task, we recorded the models obtained after each training epoch throughout the lifelong learning process and tested them on the current and previous tasks. Figure 4 shows the results in task order 0. The results of the other two orders can be found in the supplementary material.

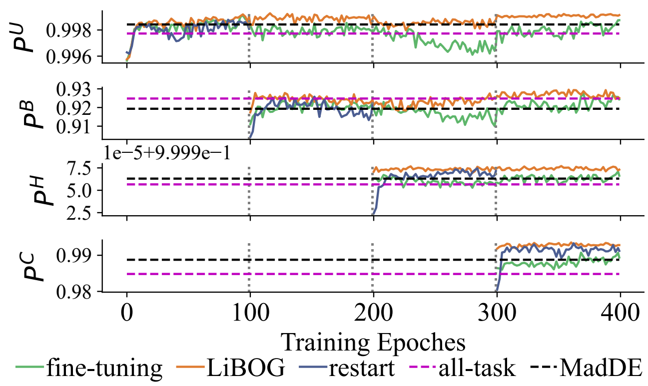


Figure 4: Test performance on each task of models obtained during the lifelong learning process, under task order 0. Vertical gray lines indicate the time of task changes.

The performance of both *fine-tuning* and LiBOG on P_1 (P^B), P_2 (P^H), and P_3 (P^C) is significantly better than *restart* on epochs 100, 200, and 300, respectively. It indicates that using the model trained on previous tasks as the initial

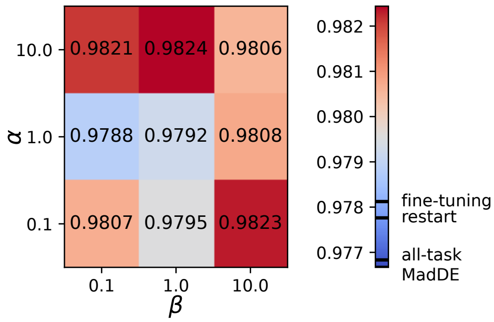


Figure 5: Performance of LiBOG trained with different values of α and β. Under all settings, LiBOG outperforms baselines.

model leads to better initial performance, compared to a randomly initialized model used in *restart*. Helpful knowledge is transferred with the trained model.

In some cases, during training on a new task (e.g, training on P_2 (P^H) during epochs 200-299 in Figure 4), *fine-tuning* gradually loses performance on previous tasks (e.g., performance reducing on P_0 (P^U) and P_1 (P^B)), demonstrating significant inter-task forgetting. Moreover, *fine-tuning* fails to effectively learn and improve performance on the new task. Significant shifts in experience distribution, combined with the inherent stochasticity of the BBO process, could be the reason, leading to intra-task forgetting and poor plasticity.

LiBOG obtained good performance during learning on the earlier tasks (i.e., on P_0 (P^U) and P_1 (P^B) during epochs 0-199 in Figure 4). When subsequent tasks are introduced, LiBOG demonstrates promising performance on the new tasks even before dedicated training (i.e., on P_2 (P^H) and P_3 (P^C) at epochs 200 and 300, respectively), while maintaining this performance stably without forgetting. It indicates that the two consolidation mechanisms work together to facilitate both the retention of old knowledge and the stable learning of new knowledge.

4.3 Sensitivity Analysis and Ablation Study

We evaluated the performance of LiBOG under different hyper-parameter settings, and conducted an ablation study to provide a more comprehensive and reliable analysis.

Consolidation weights. To address catastrophic forgetting, LiBOG incorporates two consolidation mechanisms, each introducing a term to the loss function, controlled by two weights α and β. Under task order 0, we trained LiBOG with different weight settings and recorded its average test performance across all tasks after learning all tasks. The tested candidate values are $\{0.1, 1, 10\}$ for both α and β. Each setting underwent five lifelong learning runs, and Figure 5 demonstrates the average of the five runs. Although the weights affect the performance, the performance of LiBOG is relatively stable, and outperforms baselines for all settings.

Ablation study. We conducted an ablation study to evaluate the contribution of the two consolidation mechanisms. By individually removing the inter-task consolidation and intra-task consolidation mechanisms, we created two ablation versions: LiBOG with only intra-task consolidation (denoted as *only-intra*) and LiBOG with only inter-task consolidation (denoted as *only-inter*). For each ablation version, we performed lifelong learning five repeat runs on task order 0 and tested the final model's performance across all tasks. The best-performance weight values above are used. Table 2 shows the average of the five runs. LiBOG outperforms both ablation versions, indicating that both consolidation mechanisms are critical to LiBOG's effectiveness, as removing either significantly reduces its performance.

Method	Performance	Method	Performance
LiBOG	**0.982440**	fine-tuning	0.978123
only-intra	0.982306	restart	0.977762
only-inter	0.975054	all-task	0.976833
		MadDE	0.976669

Table 2: Ablation study results of inter-consolidation and intra-consolidation mechanisms in LiBOG.

5 Conclusions

In real-world scenarios, diverse optimization problems often arise sequentially, with the problem distribution changing over time. Focusing on these scenarios, this paper studies the unexplored paradigm of lifelong learning for BBO optimizer generation. We propose LiBOG, a novel lifelong learning-based MetaBBO approach. In LiBOG, the optimization process of MetaBBO is formulated as a non-stationary MDP, where the state transition, reward distribution, and initial state distribution vary across tasks. LiBOG employs elastic weight consolidation (EWC) for inter-task consolidation, mitigating catastrophic forgetting caused by differences in problem distributions between tasks. Additionally, we propose elite behavior consolidation (EBC), a novel method that aligns model behavior with elite models obtained within a single task for intra-task consolidation. Experiments on various task orders and hyper-parameter settings demonstrate the effectiveness and robustness of LiBOG in transferring knowledge for enhanced learning on new tasks and addressing catastrophic forgetting. The ablation study verifies the contribution of each consolidation mechanism.

Despite its promising performance, LiBOG has some potential limitations. It cannot be directly applied to scenarios with continuously changing problem distribution or the distribution changing time points are unknown (i.e., unknown task boundaries), as EWC requires explicitly storing a model and parameter importance matrix for each task. Furthermore, EBC relies on constraining model updates based on elite models, which may limit performance when the performance landscape is highly complex and requires extensive exploration. Addressing these limitations will have the potential to enhance LiBOG's expertise in broader applications.

Acknowledgments

This work was supported by the National Key R&D Program of China (Grant No. 2023YFE0106300) and Lingnan University (SUG-008/2425).

References

[Abel *et al.*, 2018] David Abel, Yuu Jinnai, Sophie Yue Guo, George Konidaris, and Michael Littman. Policy and value transfer in lifelong reinforcement learning. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 20–29. PMLR, 10–15 Jul 2018.

[Audet and Kokkolaras, 2016] Charles Audet and Michael Kokkolaras. Blackbox and derivative-free optimization: theory, algorithms and applications. *Optimization and Engineering*, 17(1):1–2, February 2016.

[Beyer and Schwefel, 2002] Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies – A comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

[Bi *et al.*, 2022] Sirui Bi, Benjamin Stump, Jiaxin Zhang, Yousub Lee, John Coleman, Matt Bement, and Guannan Zhang. Blackbox optimization for approximating high-fidelity heat transfer calculations in metal additive manufacturing. *Results in Materials*, 13:100258, 2022.

[Biswas *et al.*, 2021] Subhodip Biswas, Debanjan Saha, Shuvodeep De, Adam D Cobb, Swagatam Das, and Brian A Jalaian. Improving differential evolution through Bayesian hyperparameter optimization. In *IEEE Congress on Evolutionary Computation*, pages 832–840, 2021.

[Chaybouti *et al.*, 2022] Sofian Chaybouti, Ludovic Dos Santos, Cedric Malherbe, and Aladin Virmaux. Meta-learning of black-box solvers using deep reinforcement learning. In *Sixth Workshop on Meta-Learning at the Conference on Neural Information Processing Systems*, 2022.

[Chen *et al.*, 2017] Yutian Chen, Matthew W. Hoffman, Sergio Gómez Colmenarejo, Misha Denil, Timothy P. Lillicrap, Matt Botvinick, and Nando de Freitas. Learning to learn without gradient descent by gradient descent. In *International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 748–756. PMLR, 06–11 Aug 2017.

[Chen *et al.*, 2023] Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V Le. Symbolic discovery of optimization algorithms. In *Advances in Neural Information Processing Systems*, volume 36, pages 49205–49233. Curran Associates, Inc., 2023.

[Chen *et al.*, 2024] Jiacheng Chen, Zeyuan Ma, Hongshu Guo, Yining Ma, Jie Zhang, and Yue-Jiao Gong. SYMBOL: Generating flexible black-box optimizers through symbolic equation learning. In *International Conference on Learning Representations*, 2024.

[Ghiassian *et al.*, 2020] Sina Ghiassian, Banafsheh Rafiee, Yat Long Lo, and Adam White. Improving performance in reinforcement learning by breaking generalization in neural networks. In *International Conference on Autonomous Agents and MultiAgent Systems*, page 438–446, Richland, SC, 2020. International Foundation for Autonomous Agents and Multiagent Systems.

[Gomes *et al.*, 2021] Hugo Siqueira Gomes, Benjamin Léger, and Christian Gagné. Meta learning black-box population-based optimizers, 2021.

[Gu *et al.*, 2021] Bin Gu, Guodong Liu, Yanfu Zhang, Xiang Geng, and Heng Huang. Optimizing large-scale hyperparameters via automated learning algorithm. *arXiv, 2102.09026*, 2021.

[Handoko *et al.*, 2014] Stephanus Daniel Handoko, Duc Thien Nguyen, Zhi Yuan, and Hoong Chuin Lau. Reinforcement learning for adaptive operator selection in memetic search applied to quadratic assignment problem. In *Companion Publication of Conference on Genetic and Evolutionary Computation*, page 193–194. ACM, 2014.

[Hoos, 2012] Holger H Hoos. Automated algorithm configuration and parameter tuning. In *Autonomous search*, pages 37–71. Springer, 2012.

[Hussain *et al.*, 2018] Kashif Hussain, Mohd Najib Mohd Salleh, Shi Cheng, and Yuhui Shi. Metaheuristic research: A comprehensive survey. *Artificial Intelligence Review*, 52(4):2191–2233, 2018.

[Igl *et al.*, 2021] Maximilian Igl, Gregory Farquhar, Jelena Luketina, Wendelin Boehmer, and Shimon Whiteson. Transient non-stationarity and generalisation in deep reinforcement learning. In *International Conference on Learning Representations*, 2021.

[Khetarpal *et al.*, 2022] Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, December 2022.

[Kirkpatrick *et al.*, 1983] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[Kirkpatrick *et al.*, 2017] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[Lan *et al.*, 2023] Qingfeng Lan, Yangchen Pan, Jun Luo, and A. Rupam Mahmood. Memory-efficient reinforcement learning with value-based knowledge consolidation. *Transactions on Machine Learning Research*, 2023.

[Lechner *et al.*, 2022] Mathias Lechner, Ramin Hasani, Philipp Neubauer, Sophie Neubauer, and Daniela Rus. Pyhopper – Hyperparameter optimization. *arXiv, 2210.04728*, 2022.

[Liu *et al.*, 2023] Shengcai Liu, Yu Zhang, Ke Tang, and Xin Yao. How good is neural combinatorial optimization? A systematic evaluation on the traveling salesman problem. *IEEE Computational Intelligence Magazine*, 18(3):14–28, 2023.

[Lu *et al.*, 2020] Hao Lu, Xingwen Zhang, and Shuang Yang. A learning-based iterative method for solving vehicle routing problems. In *International Conference on Learning Representations*, 2020.

[Ma *et al.*, 2023] Zeyuan Ma, Hongshu Guo, Jiacheng Chen, Zhenrui Li, Guojun Peng, Yue-Jiao Gong, Yining Ma, and Zhiguang Cao. Metabox: A benchmark platform for meta-black-box optimization with reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pages 10775–10795. Curran Associates, Inc., 2023.

[Malan and Engelbrecht, 2013] Katherine M. Malan and Andries P. Engelbrecht. A survey of techniques for characterising fitness landscapes and some possible ways forward. *Information Sciences*, 241:148–163, 2013.

[Mohamed *et al.*, 2021] Ali Wagdy Mohamed, Anas A Hadi, Ali Khater Mohamed, Prachi Agrawal, Abhishek Kumar, and PN Suganthan. Problem definitions and evaluation criteria for the CEC 2021 on single objective bound constrained numerical optimization. In *IEEE Congress on Evolutionary Computation*, 2021.

[Pan *et al.*, 2021] Yangchen Pan, Kirby Banman, and Martha White. Fuzzy tiling activations: A simple approach to learning sparse representations online. In *International Conference on Learning Representations*, 2021.

[Pei *et al.*, 2024] Jiyuan Pei, Jialin Liu, and Yi Mei. Learning from offline and online experiences: A hybrid adaptive operator selection framework. In *Proceedings of the Genetic and Evolutionary Computation Conference*, page 1017–1025, New York, NY, USA, 2024. ACM.

[Pei *et al.*, 2025] Jiyuan Pei, Yi Mei, Jialin Liu, Mengjie Zhang, and Xin Yao. Adaptive operator selection for metaheuristics: A survey. *IEEE Transactions on Artificial Intelligence*, pages 1–21, 2025.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv, 1707.06347*, 2017.

[Sharma *et al.*, 2019] Mudita Sharma, Alexandros Komninos, Manuel López-Ibáñez, and Dimitar Kazakov. Deep reinforcement learning based parameter control in differential evolution. In *Proceedings of the Genetic and Evolutionary Computation Conference*, page 709–717, New York, NY, USA, 2019. Association for Computing Machinery.

[Storn and Price, 1997] Rainer Storn and Kenneth Price. Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997.

[Tang and Yao, 2024] Ke Tang and Xin Yao. Learn to optimize – A brief overview. *National Science Review*, 11(8):nwae132, 04 2024.

[Thrun, 1998] Sebastian Thrun. *Lifelong Learning Algorithms*, pages 181–209. Springer US, Boston, MA, 1998.

[Tsaban *et al.*, 2022] Tomer Tsaban, Julia K. Varga, Orly Avraham, Ziv Ben-Aharon, Alisa Khramushin, and Ora Schueler-Furman. Harnessing protein folding neural networks for peptide–protein docking. *Nature Communications*, 13(1), January 2022.

[TV *et al.*, 2020] Vishnu TV, Pankaj Malhotra, Jyoti Narwariya, Lovekesh Vig, and Gautam Shroff. Meta-learning for black-box optimization. In *Machine Learning and Knowledge Discovery in Databases*, pages 366–381. Springer International Publishing, 2020.

[Wang *et al.*, 2020] Zhi Wang, Han-Xiong Li, and Chunlin Chen. Incremental reinforcement learning in continuous spaces via policy relaxation and importance weighting. *IEEE Transactions on Neural Networks and Learning Systems*, 31(6):1870–1883, 2020.

[Wang *et al.*, 2023] Xilu Wang, Yaochu Jin, Sebastian Schmitt, and Markus Olhofer. Recent advances in bayesian optimization. *ACM Comput. Surv.*, 55(13s), July 2023.

[Wang *et al.*, 2024] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5362–5383, 2024.

[Yang *et al.*, 2025] Yifan Yang, Gang Chen, Hui Ma, Cong Zhang, Zhiguang Cao, and Mengjie Zhang. Graph assisted offline-online deep reinforcement learning for dynamic workflow scheduling. In *International Conference on Learning Representations*, 2025.

[Yi *et al.*, 2023] Wenjie Yi, Rong Qu, Licheng Jiao, and Ben Niu. Automated design of metaheuristics using reinforcement learning within a novel general search framework. *IEEE Transactions on Evolutionary Computation*, 27(4):1072–1084, 2023.

[Zhang *et al.*, 2023] Tiantian Zhang, Xueqian Wang, Bin Liang, and Bo Yuan. Catastrophic interference in reinforcement learning: A solution based on context division and knowledge distillation. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):9925–9939, 2023.

[Zheng *et al.*, 2022] Wenqing Zheng, Tianlong Chen, Ting-Kuei Hu, and Zhangyang Wang. Symbolic learning to optimize: Towards interpretability and scalability. *arXiv, 2203.0657*, 2022.