ICLR

华南理工大学

South China University of Technology

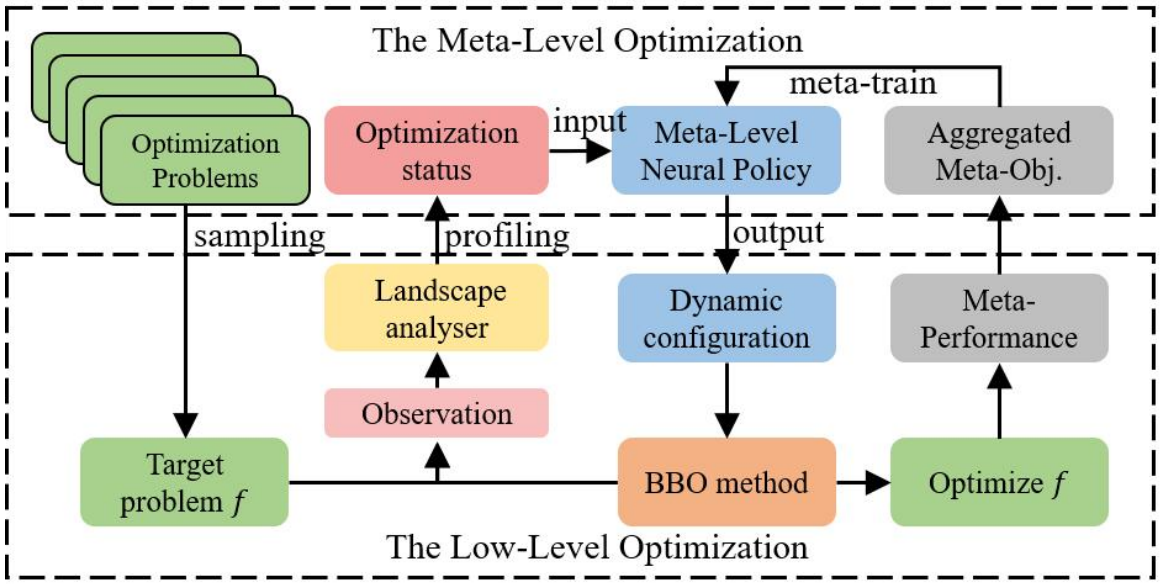
Neural Exploratory Landscape Analysis for Meta-Black-Box-Optimization

By: Yue-Jiao Gong

ICLR

South China University of Technology

Part I: What is Meta-Black-Box-Optimization (MetaBBO)?



MetaBBO leverages the generalization strength of Meta-learning to enhance the optimization performance of BBO algorithms in the minimal expertise cost [1] [2].

Bi-level Paradigm:

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

$$\omega_i^t = \pi_\theta(s_i^t), \quad s_i^t = \mathrm{sf}(\mathcal{A}, f_i, t)$$

Meta-level Algorithm Design Task:
The policy π_θ is trained to dictates desired algorithm design ω by conditioning the optimization state feature s.

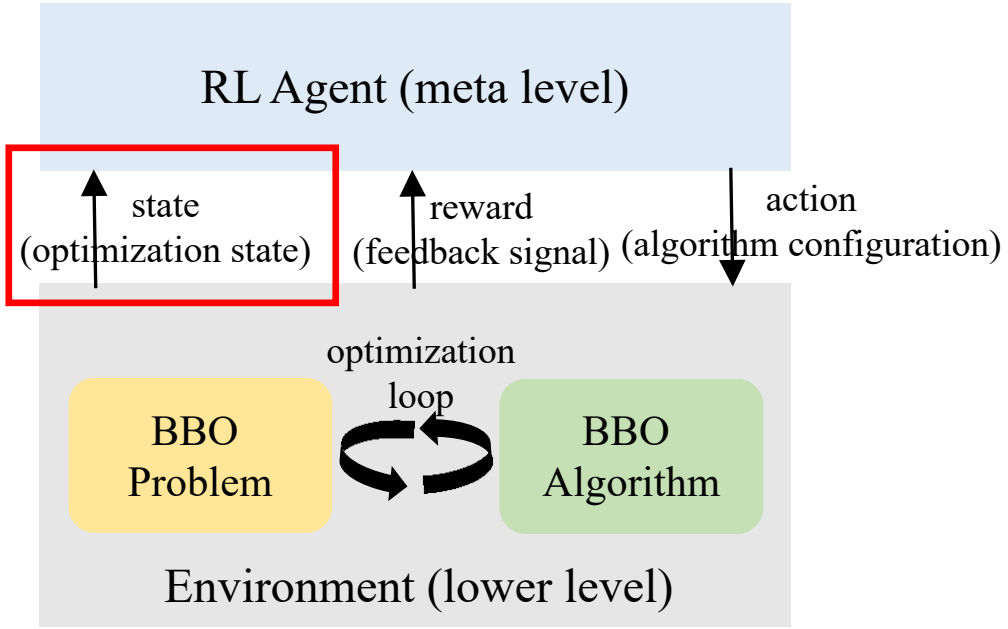
Low-level Optimization Task:
The algorithm A adopts the dictated design to optimize a distribution of problems P, providing feedback signal $perf(\cdot)$ for training the policy.

[1] Ma Z et al. MetaBox: A Benchmark Platform for Meta-Black-Box Optimization with Reinforcement Learning. NeurIPS 2023.

[2] Ma Z et al. Toward Automated Algorithm Design: A Survey and Practical Guide to Meta-Black-Box-Optimization. arXiv preprint arXiv:2411.00625, 2024.

Part II: Motivation of this paper?

The learning effectiveness of MetaBBO relies on an informative optimization state feature design, which reflects the BBO problem properties and timely profiles the current optimization progress.



The optimization state design can be defined as:

$$\lambda: (X, f(X)) \rightarrow s$$

where X is the sample points, f() is the objective function.

Exploratory Landscape Analysis [1] is a common practice in existing MetaBBO works for state feature design, which is handcrafted by human experts for decribing a BBO problem.

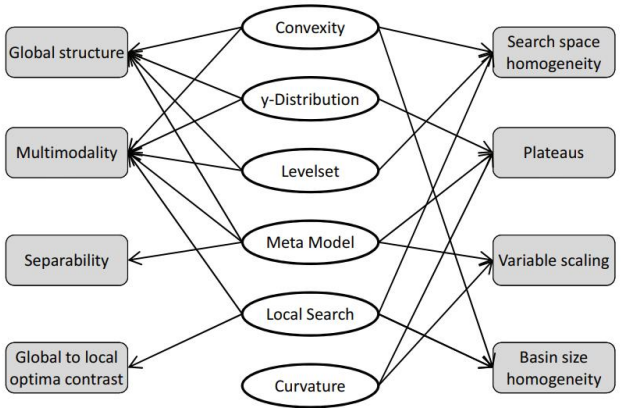
[1] Mersmann O et al. Exploratory landscape analysis. Proceedings of the 13th annual conference on Genetic and evolutionary computation. 2011: 829-836.

ICLR

South China University of Technology

Part II: Motivation of this paper?

Traditional ELA [1]



not tailored for MetaBBO ✗

crafted by human experts ✗

high comp. complexity ✗

MetaBBO Customization [2][3]

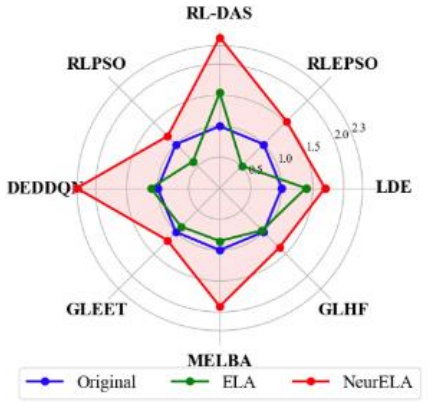
		States	Notes
Population	s_1	$mean_{i,j}\|\|x_i^{(t)} - x_j^{(t)}\|\|_2$	Average distance between any pair of individuals in current population.
	s_2	$mean_i\|\|x_i^{(t)} - x^{*,(t)}\|\|_2$	Average distance between each individual and the best individual in t^{th} generation.
	s_3	$mean_i\|\|x_i^{(t)} - x^{*}\|\|_2$	Average distance between each individual and the best-so-far solution.
Objective	s_4	$mean_i(f(x_i^{(t)}) - f(x^*))$	Average objective value gap between each individual and the best-so-far solution.
	s_5	$mean_i(f(x_i^{(t)}) - f(x^{*,(t)}))$	Average objective value gap between each individual and the best individual in t^{th} generation.
	s_6	$std_i(f(x_i^{(t)}))$	Standard deviation of the objective values of population in t^{th} generation, a value equals 0 denotes converged.
Time Stamp	s_7	$(T-t)/T$	The potion of remaining generations, T denotes maximum generations for one run.
	s_8	st/T	st denotes how many generations the optimizer stagnates improving.
	s_9	$\begin{cases}1 & \text{if } f(x^{*,(t)}) < f(x^*) \\ 0 & \text{otherwise}\end{cases}$	Whether the optimizer finds better individual than the best-so-far solution.

tailored for MetaBBO ✓

crafted by human experts ✗

high comp. complexity ✗

NeurELA (this paper)



universal for MetaBBOs ✓

Automated learning ✓

Efficient computation ✓

[1] Mersmann O et al. Exploratory landscape analysis. Proceedings of the 13th annual conference on Genetic and evolutionary computation. 2011: 829-836.

[2] Ma et al. Auto-configuring exploration-exploitation tradeoff in evolutionary computation via deep reinforcement learning. ACM GECCO 2024.

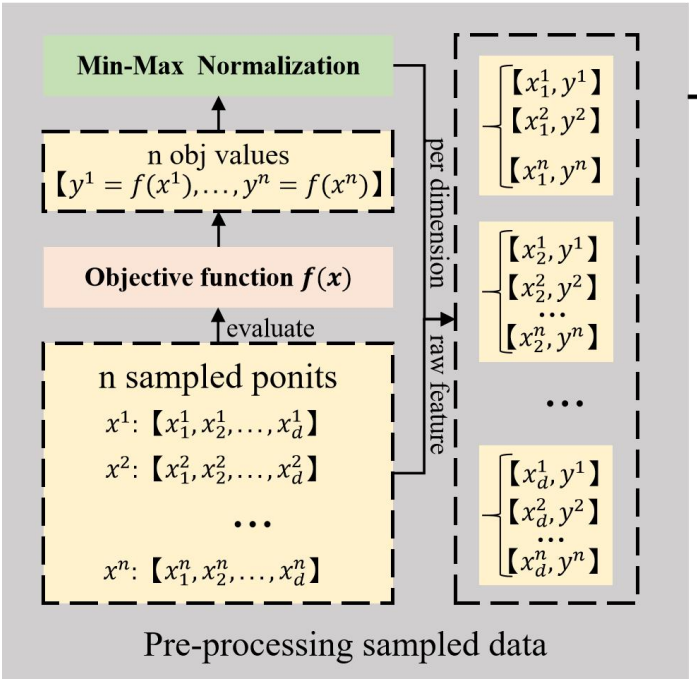
[3] Chen et al. SYMBOL: Generating Flexible Black-Box Optimizers through Symbolic Equation Learning. ICLR 2025.

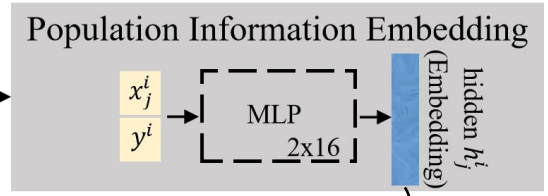
ICLR

South China University of Technology

Part III: Methodology of NeurELA

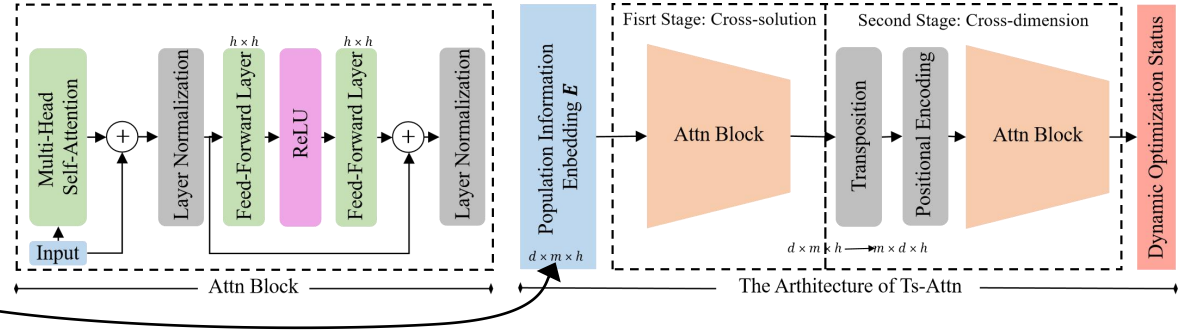
III.A Neural Network Design and Computational Workflow





Min-Max Norm.

normalization on the sample points (by ub/lb) and objective values (within batch) facilitates generalization across various optimization problems.



Cross-solution Attention: same dimension, different samples

Cross-dimension Attention: same sample, different dimensions

Advantages
i) fully-informed feature extraction
ii) attention mechanism ensures model scalability

Pre-processing stage provides a universal way to process sample points with various dimensions/scales

Feature extraction stage provides a comprehensive data-mining through two attention-based blocks

Part III: Methodology of NeurELA

III.B Training Objective of NeurELA

If we aims at learning a universal landscape analyser for various MetaBBOs, we need a *multi-task operating space*:

$$\Omega = \{\mathbb{T}_k = (\mathbb{A}_k, \mathbb{D}_k) \mid k = 1, 2, \ldots, K\}$$

For each MetaBBO task T_k, it comprises a MetaBBO method A_k and dataset D_k (BBO instance collection).
Then, a *relative performance metric* is defined as:

$$\Upsilon(\Lambda_\theta \mid \mathbb{T}_k) = \frac{1}{P \times Q} \sum_{p=1}^{P} \sum_{q=1}^{Q} Z_{p,q}, \text{ where } Z_{p,q} = -\frac{f^*_{p,q} - \mu_p}{\sigma_p}$$

It measures the performance improvement when using NeurELA to replace the original feature extraction design in A_k.
Finally, the *training objective* of NeurELA (Λ_θ) is formulated as:

$$\max_\theta F_{\text{metaELA}} = \mathbb{E}_{\mathbb{T}_k \sim \Omega}[\Upsilon(\Lambda_\theta \mid \mathbb{T}_k)] = \frac{1}{K} \sum_{k=1}^{K} \Upsilon(\Lambda_\theta \mid \mathbb{T}_k)$$

Note: the training objective above is not differentiable along the neural network parameter θ of NeurELA. This implicitely indicates that we have to find an alternative way beyond the gradient descent methods to train NeurELA !!!

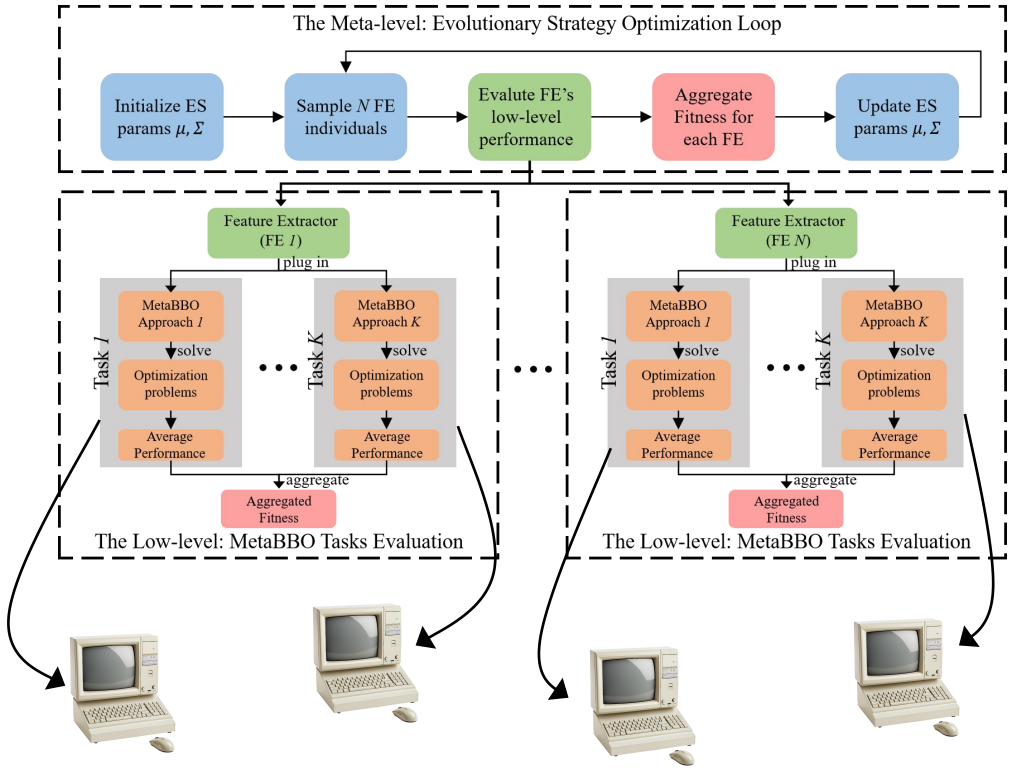
Part III: Methodology of NeurELA

III.C Training Algorithm for NeurELA

Algorithm 1 Pseudo code of training NeurELA

Input: The MetaBBO task space Ω, Evolution Strategy ES, optimization horizon $maxGen$.
Output: Optimal neural landscape analyser Λ_{θ^*}
Initialize a group of $\{\Lambda_{\theta_i}\}_{i=1}^{N}$ by $ES.init()$;
for $generation = 1$ **to** $maxGen$ **do**
  for each Λ_{θ_i} **do**
    for each MetaBBO task $\mathbb{T}_k \in \Omega$ **do**
      Construct $\mathbb{A}_k(\Pi_\phi, \Gamma, \Lambda_0)$ and $\mathbb{A}'_k(\Pi_\phi, \Gamma, \Lambda_\theta)$;
      Meta-train $\mathbb{A}_k$ and $\mathbb{A}'_k$ on $\mathbb{D}_{train}$;
      Test meta-trained $\mathbb{A}_k$ and $\mathbb{A}'_k$ on $\mathbb{D}_{\text{test}}$;
    end for
    Evaluate the fitness f_i of Λ_{θ_i} by F_{metaELA} in Eq. (4);
  end for
  Set Λ_{θ^*} as the one with the highest fitness so far;
  Update the distributional parameters in ES by $ES.update(\{\Lambda_{\theta_i}\}_{i=1}^{N}, \{f_i\}_{i=1}^{N})$;
  Sample a new generation of $\{\Lambda_{\theta_i}\}_{i=1}^{N}$ by $ES.sample()$;
end for

Instead of using gradient descent, we adopt Neuroevolution [1] for training NeurELA. It is a paradigm where gradient information is not required and the parameters are trained in a black-box way. We adopt a poulation-based optimizer Fast CMA-ES [2] to evolve a population of NeurELA networks.



Ray [3] is used in our training algorithm implementation to distribute time-comsuming MetaBBO inner-loop.

[1] Miikkulainen R et al. Evolving deep neural networks. Artificial intelligence in the age of neural networks and brain computing. 2024.

[2] Li et al. Fast covariance matrix adaptation for large-scale black-box optimization. IEEE Transactions on Cybernetics. 2018.

[3] Philipp M et al. Ray: A distributed framework for emerging ai applications. Symposium on Operating Systems Design and Implementation. 2018.

ICLR

South China University of Technology

Part IV: Empirical Validation and In-depth Analysis

IV.A Training setup

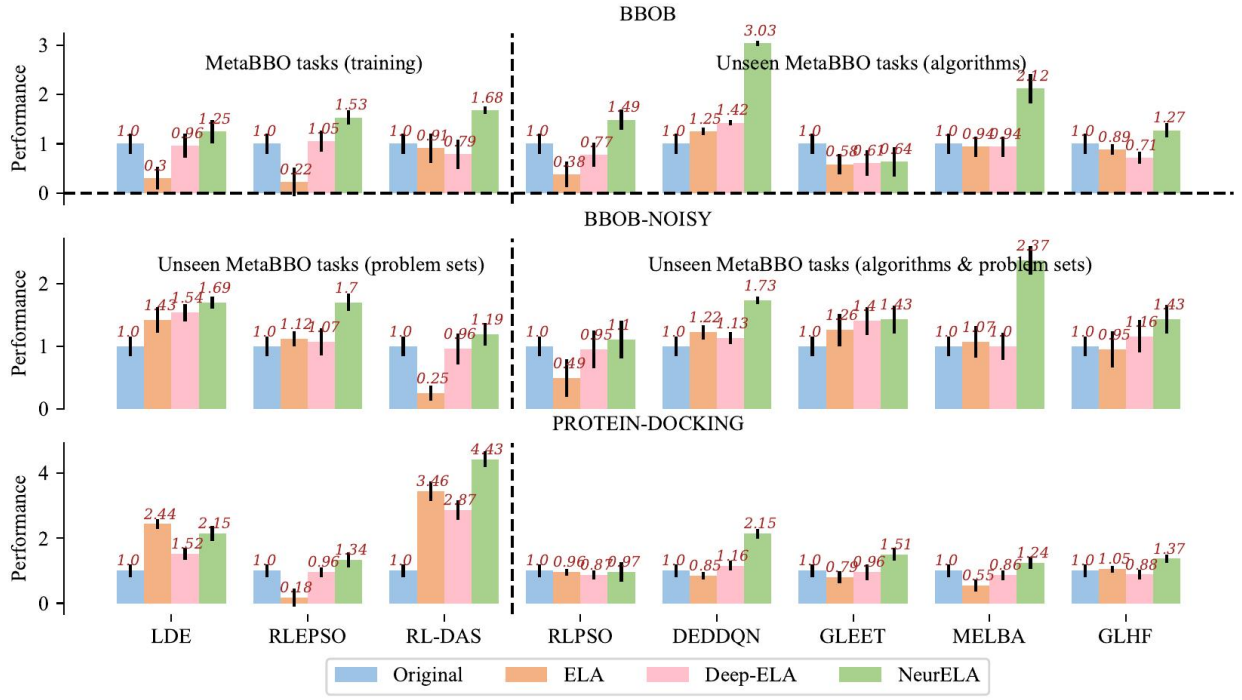
1. Training task space:
-MetaBBO algs: LDE [1]
RLEPSO [2]
RL-DAS [3]
-BBO ins. set: CoCo-BBOB [4]

2. Hyper-param. of Fast CMA-ES:
-pop_size: 10
-max_gens: 50
-init_sigma: 0.3
-lr_c: 6.06 e-4

3. Baselines:
-Original: those in MetaBBOs
-ELA [5]: traditional ELA
-DeepELA [6]: learning-based

4. Test scenarios:
-five unseen MetaBBOs
-two novel BBO ins. sets
-both the unseen MetaBBOs and the novel BBO ins. sets

A node of a Slurm cluster, with 2 Sapphire Rapids 6458Q CPUs and 256 GB memories.

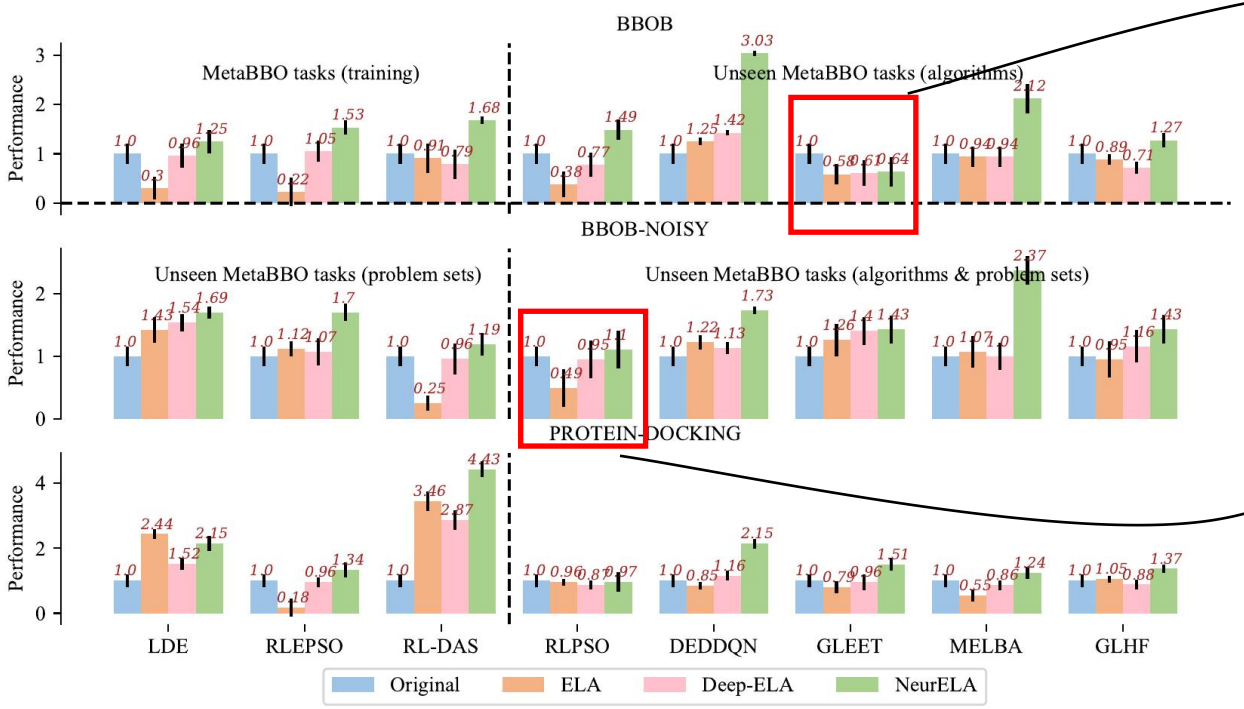


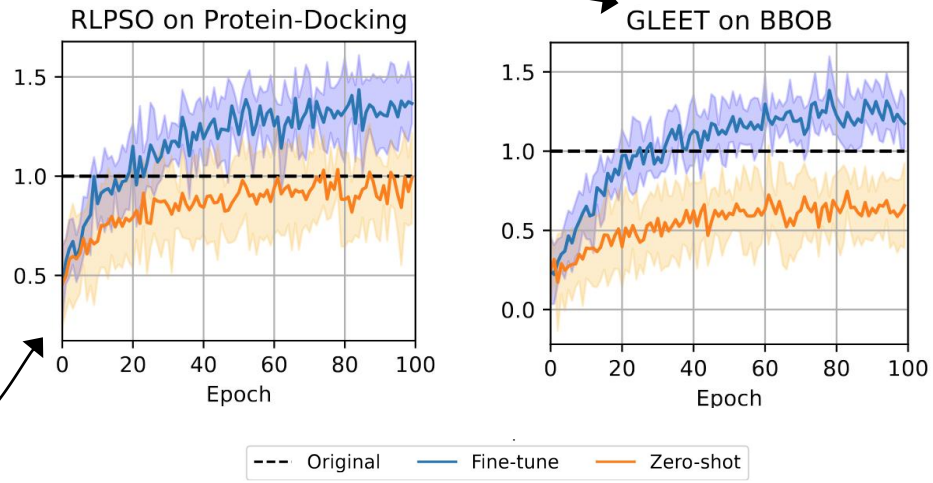
Zero-shot Perforamence

[1] Sun et al. MetaBox: earning adaptive differential evolution algorithm from optimization experiences by policy gradient. IEEE TEVC. 2021.
[2] Yin et al. Rlepso: Reinforcement learning based ensemble particle swarm optimize. ICACAI, 2021.
[3] Guo et al. Deep reinforcement learning for dynamic algorithm selection: IEEE TSMC, 2024.
[4] Hansen et al. Coco: A platform for comparing continuous optimizers in a black-box setting. Optimization Methods and Software, 2021.
[5] Kerschke et al. omprehensive feature-based landscape analysis of continuous and constrained optimization problems using the r-package flacco, 2019.
[6] Seiler et al. Deep-ela: Deep exploratory landscape analysis with self-supervised pretrained transformers for single-and multi-objective continuous optimization problems, 2024.

Part IV: Empirical Validation and In-depth Analysis

IV.B Generalization Performance of NeurELA





Direct Zero-shot performance of NeurELA is generally good, which validates the effectiveness of the training of NeurELA. However, in some cases, integrating NeurELA into might not achieves ideal performance, how do we do?

In case of NeurELA can not boost the target MetaBBO task, it supports fine-tuning by co-trained with MetaBBO

The MetaBBO training curves above clearly demonstrate that NeurELA could be efficiently fine-tuned.

ICLR

paper link

Our team page

South China University of Technology

Part IV: Empirical Validation and In-depth Analysis

IV.C Computational Complexity

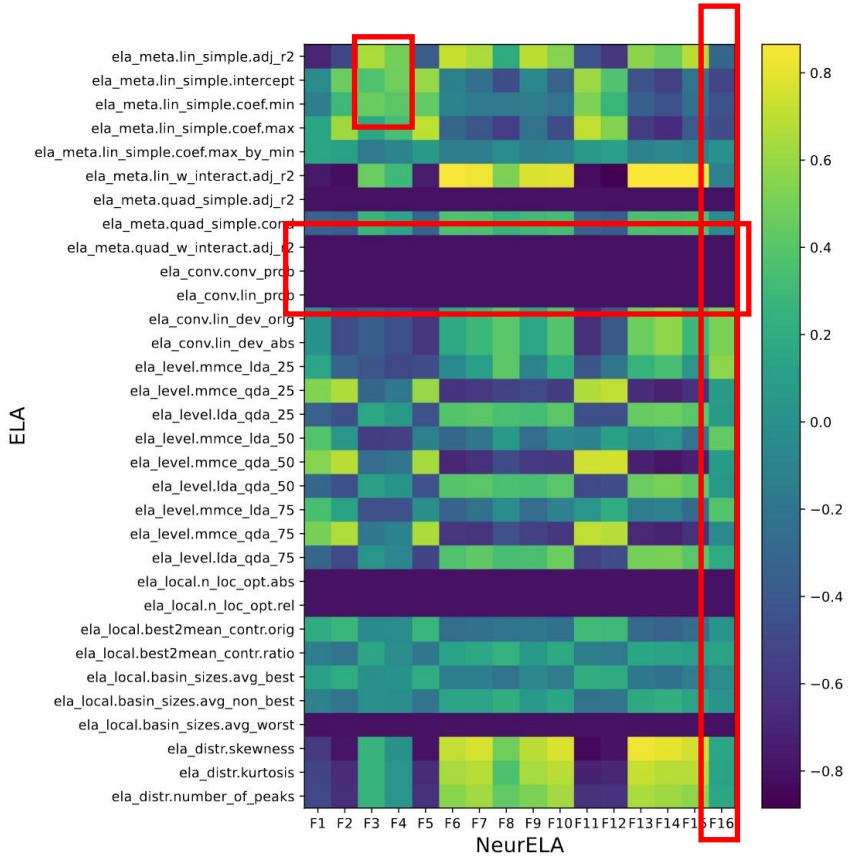
	$m = 100$		
	$d = 10$	$d = 100$	$d = 1000$
Original	**4.99E-03**	**5.61E-03**	**6.38E-03**
ELA	1.49E-01	1.48E+01	3.00E+02
NeurELA	5.42E-03	6.24E-03	7.11E-03
	$d = 10$		
	$m = 100$	$m = 500$	$m = 1000$
Original	**4.99E-03**	2.53E-02	5.16E-02
ELA	1.49E-01	1.69E-01	2.02E-01
NeurELA	5.42E-03	**7.97E-03**	**9.38E-03**

computational wall-time when problem dimension (d) and sample size (m) vary

NeurELA would not increase the complexity compared to the original designs in MetaBBOs

ELA indeed present exponentially increasing computational cost

IV.D What has NeurELA learned?



correlation with ELA features

NeurELA has discovered some novel features that show weak correlation with ELA, e,g, F16

some NeurELA features (F3, F4) correlate with meta_model feature group of ELA.

the convexity feature group in ELA show no correlation with NeurELA features.