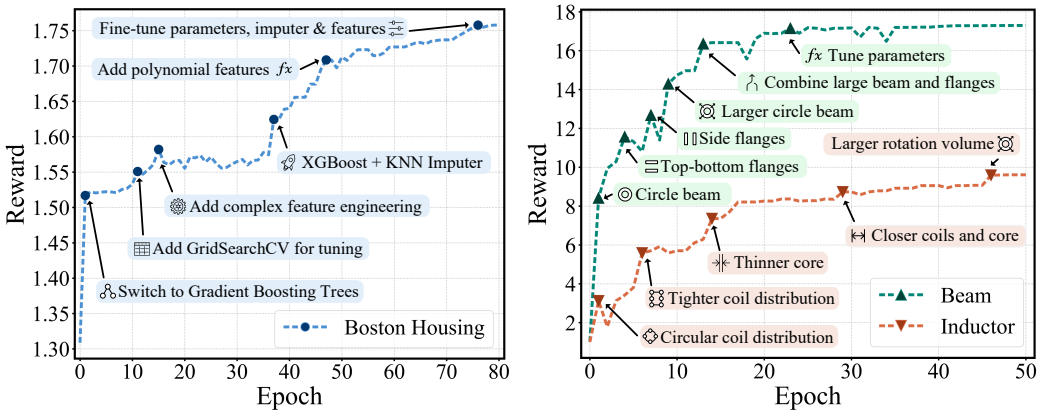
HELIX: EVOLUTIONARY REINFORCEMENT LEARNING FOR OPEN-ENDED SCIENTIFIC PROBLEM SOLVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) with reasoning abilities have demonstrated growing promise for tackling complex scientific problems. Yet such tasks are inherently domain-specific, unbounded and open-ended, demanding exploration across vast and flexible solution spaces. Existing approaches, whether purely learning-based or reliant on carefully designed workflows, often suffer from limited exploration efficiency and poor generalization. To overcome these challenges, we present **HELIX**—a **H**ierarchical **E**volutionary reinforcement **L**earning framework with **I**n-context e**X**periences. HELIX introduces two key novelties: (i) a diverse yet high-quality pool of candidate solutions that broadens exploration through in-context learning, and (ii) reinforcement learning for iterative policy refinement that progressively elevates solution quality. This synergy enables the discovery of more advanced solutions. On the circle packing task, HELIX achieves a new state-of-the-art with a sum of radii of 2.635983 using only a 14B model. Across standard machine learning benchmarks, HELIX further surpasses GPT-4o with a carefully engineered pipeline, delivering an average F1 improvement of 5.95 points on the Adult and Bank Marketing datasets and a 40.5% reduction in RMSE on Boston Housing.



(a) Housing Dataset
(Finding ML algorithms)

(b) Beam and Inductor Design
(Finding better geometric shape)

Figure 1: The figure demonstrates how our framework progressively discovers new insights and refines solutions over iterations. **(a):** Reward curve for the housing dataset optimization, where improvements are achieved through iterative adoption of better models, parameter tuning, and feature engineering, with the final reward of 1.758 corresponding to an RMSE of 1.747. **(b):** Reward curves for the beam and inductor design tasks, where the algorithm explores novel geometries and combines favorable structural features to enhance performance.

1 INTRODUCTION

Solving complex scientific problems with large language models (LLMs) is an important and increasingly active research direction (Forootani, 2025). By leveraging and enhancing their reasoning capabilities, LLMs have demonstrated promising results in tackling challenging scientific tasks, such as symbolic regression (Shojaee et al., 2024), molecular generation (Liu et al., 2024), and difficult mathematical optimization problems (Ahmed & Choudhury, 2024). Addressing such problems holds the potential to advance the boundaries of human knowledge and reshape scientific discovery.

While LLMs have shown promising applications, complex scientific problems remain particularly challenging due to three intrinsic characteristics. First, they are *domain-specific*, with unique environments and problem-specific constraints that differ across various tasks. Second, they are *open-ended*, requiring exploration of vast and flexible solution spaces. Third, they are *unbounded*, often with no known or guaranteed global optimum.

To address these challenges, we argue that a powerful LLM for solving complex scientific problems must possess three corresponding key abilities: **(1) learning from experience**, i.e., it should enable task-specific policy adaptation by incorporating feedback from previous trials, addressing the *domain-specific* nature of each problem. **(2) Balancing quality and diversity**, i.e., it should maintain a diverse population to thoroughly explore the vast and flexible solution spaces inherent in *open-ended* tasks. **(3) Exploration based on the shoulder of giants**, i.e., it should iteratively build upon existing high-quality solutions to extend the known limits of *unbounded* problems.

However, recent works largely lack the capabilities outlined above, limiting their effectiveness on complex scientific problems. Existing approaches fall into two categories. *Post-training methods* (e.g., SFT, RLVR) fine-tune LLMs on domain-specific datasets, as in AlphaCode (Li et al., 2022) and Deepseek-R1 (Ren et al., 2025), achieving strong results in code generation and mathematical reasoning. Yet such methods often suffer from entropy collapse (Cui et al., 2025) and, as shown in Yue et al. (2025), rarely move beyond the base model's capabilities. This makes it difficult to discover fundamentally new solutions, especially when sparse rewards further limit exploration. *Workflow-driven approaches* embed LLMs in predefined pipelines to improve task-specific performance. Examples include integrating genetic algorithms with LLMs for enzyme discovery (Nana Teukam et al., 2025), establishing LLM-driven evolutionary loops such as LLaMEA (van Stein & Bäck, 2024), or applying evolutionary strategies to prompts (Agrawal et al., 2025). While effective on narrow tasks, these systems are highly sensitive to workflow design and rely on static pretrained knowledge, making it hard to reuse past discoveries to guide iterative search. Both categories thus struggle to generalize in open-ended scientific domains where efficient exploration and continual refinement are essential.

To this end, we propose **HELIX**—a **H**ierarchical **E**volutionary reinforcement **L**earning framework with **I**n-context e**X**periences. Inspired by GRPO, HELIX updates the LLM policy using reward signals to progressively improve solution quality. Candidate solutions are embedded using a pretrained model, and their diversity is estimated via KNN similarity. NSGA-II is then applied to maintain a population that balances high reward and diversity. Finally, HELIX leverages the model's in-context learning ability by incorporating information and feedback from previous trials into the prompt, enabling iterative improvement of current solutions.

In experiments, we evaluated HELIX on 19 tasks across five diverse categories. Compared with strong task-specific baselines and advanced proprietary models such as GPT-4o, HELIX achieves superior performance on 16 tasks, demonstrating its ability to iteratively refine solutions and update its policy towards better results. Further analysis via ablation studies confirms that each component of HELIX contributes critically to performance. Notably, success on these unbounded and open-ended tasks suggests that iterative, diversity-aware exploration can provide useful insights for other scientific and engineering problems.

2 RELATED WORK

Reinforcement learning of LLMs. Training LLMs or LLM-based agents with reinforcement learning (RL) has recently attracted significant attention. This includes reinforcement learning from human feedback (RLHF) to align models with human preferences, as well as RL with veri-

fiable rewards (RLVR) to enhance reasoning, mathematical problem-solving, and coding capabilities. Beyond improving reasoning, RLVR-style training can also elicit new capabilities such as tool use (Feng et al., 2025) and information retrieval (Jin et al., 2025). A representative method is GRPO (Shao et al., 2024), which normalizes rewards within groups of samples. Variants such as DAPO (Yu et al., 2025) and Dr.GRPO (Liu et al., 2025) further improve GRPO through refined data sampling strategies and advantage estimation techniques. While RL can improve generalization in specific domains, the training process often suffers from decreasing entropy and diversity over time, hindering effective exploration. Some approaches, such as KL-Conv (Cui et al., 2025), attempt to address this limitation. However, for complex scientific problems, memory-less RL methods struggle to leverage solutions that have already been discovered, making it difficult to build upon prior explorations.

Evolutionary algorithms. Evolutionary algorithms are a classic approach for tackling complex optimization problems. They use "gene" to represent a solution for the problem and use random mutation to explore the whole solution space. AlphaEvolve (Novikov et al., 2025) treats code as the "gene" and applies LLM-driven mutations, successfully integrating LLM agents with evolutionary algorithms—opening the door to solving complex scientific problems. Since then, many works have adopted similar agent-based workflows to address scientific tasks such as CUDA code optimization (Lange et al., 2025), drug discovery (Gao et al., 2025), and complex scientific software usage (Fan et al., 2025; Pham et al., 2025). However, such methods typically require highly problem-specific workflow logic and prompt design, which greatly limit their effectiveness in solving more general and complex problems.

3 Proposed Method

3.1 Overview

To tackle the challenges of applying large language models (LLMs) to complex scientific discovery tasks, we propose HELIX, a hybrid framework that integrates reinforcement learning with evolutionary search. The goal is to enable LLMs to efficiently explore large and flexible solution spaces while maintaining diversity and exploiting previously discovered high-quality solutions. The framework is composed of three complementary modules: (1) **A reinforcement learning framework** that updates the policy parameters based on verifiable reward, allowing the model to *learn from experience* and progressively improve its reasoning capability. (2) **A multi-objective evolutionary mechanism** that *balancing solution quality and diversity*, ensuring that the population retains both high-performing and diverse candidates for further expansion. (3) **An in-context learning mechanism** that incorporates multiple past trials into the prompt, enabling the model to build upon previously discovered solutions and *expand its exploration on the shoulder of giants*.

We consider the task as an optimization problem that has a solution space of code. Let $s \in \mathcal{S}$ denote a candidate solution, represented as code written in a domain-specific language (e.g., Python, YAML, or other DSLs). We define an objective reward function $R(\cdot)$ which only depends on the current solution (state). The optimization objective is to find a valid $s \in \mathcal{S}$ to maximize the reward:

$$\max_{s \in \mathcal{S}} R(s). \tag{1}$$

Assume the initial solution (state) is s_0. To explore and search new solutions, we use an LLM policy π_θ that takes the problem description, current solution s_t and previous solutions $\{s_k\}_{k<t}$ as input and outputs an action $a \in \mathcal{A}$, an edit or modification applied to s_t, to obtain a new solution $s_{t+1} = T(s_t, a)$, where T is the transition function. We collect states that are explored to a dataset $\mathcal{D}$ and selectively construct the population $\mathcal{P}$ which is used in evolutionary algorithms. Our goal is to improve the policy's ability to find better solutions. The objective is defined as follows,

$$\max_{\theta} \ \mathbb{E}_{s_t \sim \mathcal{P},\, a_t \sim \pi_\theta} \left[R(s_t, a_t) \right], \tag{2}$$

where $R(s_t, a_t) = R(s_{t+1})$ is the reward of the new solution. We leverage GRPO, a reinforcement learning algorithm, to update LLM policy π_θ. To address the exploration–exploitation trade-off and prevent entropy collapse in RL, we maintain a dataset $\mathcal{D} \subset \mathcal{S}$ of solutions discovered so far

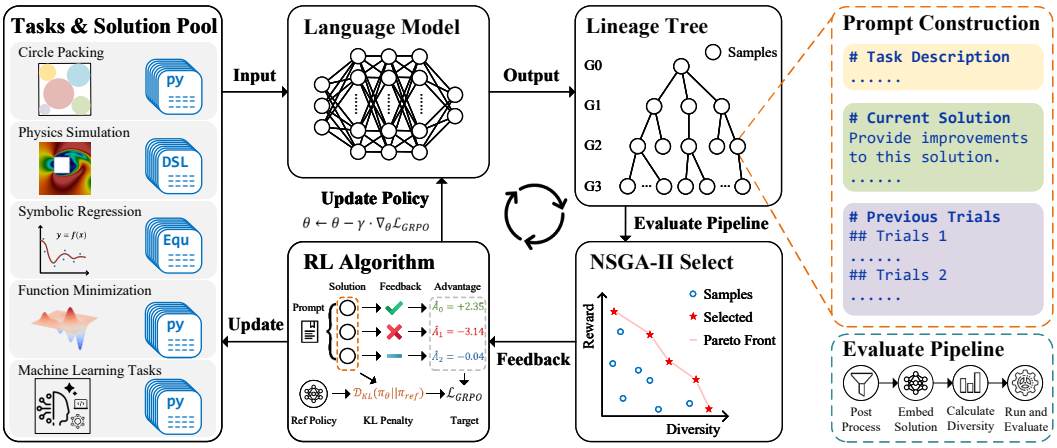


Figure 2: Illustration of HELIX framework. The workflow begins with a dataset containing task descriptions and a pool of initial solutions, which are taken by LLM as inputs. The LLM will modify and update the original solution and generate a new one, represented as descendants in lineage tree. After the evaluation pipeline, samples will be selected by NSGA-II algorithm to construct promising yet diverse candidate solutions for population evolution. The resulting reward-labeled solutions will also be used to update policy parameters via reinforcement learning.

and population $\mathcal{P} \subset \mathcal{S}$ as candidates for evolution. At each iteration, let $\{s_{t+1,i}\}$ be all solutions generated in the t-th iteration.

$$\begin{aligned} \mathcal{D} &\leftarrow \mathcal{D} \cup \{s_{t+1,i}\} \\ \mathcal{P} &\leftarrow \text{SelectTop}_{\text{NSGA-II}}(\mathcal{D}), \end{aligned} \tag{3}$$

where NSGA-II–style non-dominated sorting ensures retention of high-reward and diverse candidates. This formulation allows the model to iteratively improve its policy while exploiting previously found high-quality solutions as starting points for further exploration. Figure 2 provides a brief summary of our method.

3.2 Policy Optimization Aligned with Evolutionary Search

As the evolutionary process unfolds, updating the model parameters becomes crucial: it enables the policy to learn from both successful and failed trials, generate higher-quality solutions, and dynamically adapt to the shifting input distribution induced by the evolutionary search. Reinforcement learning is particularly suitable in this scientific setting, since open-ended scientific tasks lack standard answers and typically provide only sparse reward feedback. Motivated by the design of GRPO (Shao et al., 2024), we develop a reinforcement learning–based policy update mechanism tailored to our framework. GRPO has proven effective in enhancing LLM reasoning on mathematical and programming tasks (Guo et al., 2025), and its multi-sample generation naturally provides diverse reasoning-driven outputs that enrich the evolutionary dataset, making it a natural inspiration for our method.

Formally, given a query q and the model-generated output sequence $o_i = o_{i,1}, \ldots, o_{i,|o_i|}$ sampled from the old policy $\pi_{\theta_{\text{old}}}$, the GRPO objective is defined as:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{(q,a)\sim\mathcal{D},\ \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)}$$
$$\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{j=1}^{|o_i|}\left(\min\left(r_{i,j}(\theta)\hat{A}_{i,j}, \text{clip}(r_{i,j}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{i,j}\right) - \beta D_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})\right)\right]. \tag{4}$$

where $r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}$ is the token-level policy ratio, $\hat{A}_{i,t}$ is the token-level advantage, ϵ is the clipping parameter, and β controls the KL divergence penalty against a reference policy π_{ref}.

In our framework, each query q is constructed from the task description together with the current expanded state s_t, its ancestral states, and the corresponding rewards. Given a query q, the LLM generates G output sequences o_i, where each output corresponds to an action a_i that transforms the current state into a new state $s_{t+1,i}$. The reward signal $R(s_t, a)$ introduced in equation 2 is then assigned to the entire generated sequence and further decomposed into token-level advantages. This integration allows the policy not only to exploit feedback from rewards but also to dynamically adapt to the evolving population distribution, ensuring that the model remains effective as the search progresses across increasingly challenging solution spaces.

3.3 Evolutionary Mechanism for Balancing Quality and Diversity

In unbounded scientific research tasks, it is crucial to explore multiple promising ideas or directions. Thus, the optimization process must balance *quality*, i.e., high-reward solutions that serve as strong starting points for refinement, with *diversity*, which sustains broad exploration across the solution space. We design the evolutionary search algorithm to be a multi-objective optimization that naturally achieves a trade-off by maintaining a population that simultaneously improves in reward and preserves diverse candidates. Specifically, we innovatively adopt NSGA-II Deb et al. (2002), which is a powerful genetic algorithm, to filter high quality and diverse samples on the Pareto front of reward and diversity for subsequent expansion. To further encourage more diverse exploration and enable more accurate diversity computation, we propose to computate the diversity score based on its semantic embedding similarity using a pretrained language embedding model.

Diversity measurement. To quantify the diversity of candidate solutions, we first normalize each solution into a canonical code format and encode it into an embedding vector using a pretrained embedding model. Let $E(s) \in \mathbb{R}^d$ denote the embedding of solution $s \in \mathcal{D}$. For any solution s_i, its diversity score is computed by measuring the average similarity to its k nearest neighbors in the embedding space:

$$\text{Div}(s_i) = 1 - \frac{1}{k} \sum_{j \in \mathcal{N}_k(i)} \frac{E(s_i) \cdot E(s_j)}{\|E(s_i)\| \|E(s_j)\|}, \tag{5}$$

where $\mathcal{N}_k(i)$ denotes the indices of the k nearest neighbors of s_i in $\mathcal{D}$, measured by cosine similarity. A higher $\text{Div}(s_i)$ indicates that s_i is more distinct from other solutions, thereby contributing to population diversity.

NSGA-II based selection. Given both reward score $R(s)$ and diversity score $\text{Div}(s)$, each candidate solution can be mapped to a two-dimensional objective space. We then adopt the NSGA-II (Deb et al., 2002) algorithm to select high-quality and diverse samples. NSGA-II first applies a nondominated sorting procedure to partition solutions into multiple fronts based on Pareto dominance, where a solution s_a dominates s_b if $R(s_a) \geq R(s_b)$ and $\text{Div}(s_a) \geq \text{Div}(s_b)$ with at least one strict inequality. To further ensure diversity preservation within each front, NSGA-II computes a crowding-distance measure and selects representative samples that are well spread in the objective space.

By combining nondominated sorting with diversity preservation, the resulting population $\mathcal{P}$ retains candidates that are both high-reward and diverse. This mechanism allows the model to continuously exploit promising solutions while sustaining exploration across multiple distinct solution trajectories.

4 Experiment

In this section, we first introduce the experimental setup, including the tasks we selected for benchmarking the model's ability to solve open-ended scientific problems. Then, we present extensive experiments demonstrating that HELIX effectively enhances model capability, integrates historical experience, and balances reward with diversity, leading to significant improvements over existing baselines in solving unbounded and open-ended scientific challenges. Finally, the ablation studies reveal how different components of the framework work together in a complementary manner.

4.1 EXPERIMENT SETTING

Tasks. To comprehensively evaluate the model's capacity for complex scientific reasoning, we design experiments on five representative categories of tasks. These tasks are particularly suited for our study because they are *unbounded*, lacking a guaranteed global optimum, *open-ended*, requiring exploration over vast and flexible solution spaces and *domain-specific*, containing unique constraints and complex background. Success in these tasks not only demonstrates the model's ability to search beyond local optima, but also provides insights that can inspire solutions in broader scientific and engineering domains.

1. **Machine Learning Tasks.** We selected three representative datasets: Adult income (Becker & Kohavi, 1996), Bank marketing (Moro et al., 2014) and Boston housing (Harrison Jr & Rubinfeld, 1978) dataset to evaluate the model's ability to solve machine learning tasks. These tasks reflects the open-ended challenge of combining ML algorithms for novel applications, with potential implications for autonomous scientific workflows.
2. **Physics Simulation Tasks.** These tasks combine geometric structures design and optimization in multi-physics environments in distinct fields. The design space of these problems has a very high degree of freedom with few global optimal solution.
3. **Circle Packing Problems.** The objective of these tasks is to maximize the sum of radii of circles packed within given shapes. It allows multiple feasible arrangements and there is no proved global optimum solution currently.
4. **Function Minimization.** It requires LLM to write a code to find the global minimum point of given functions. Agents can search freely for new mathematical optimization methods in code space.
5. **Symbolic Regression.** A benchmark (Shojaee et al., 2025) evaluates the ability of LLMs to hypothesize underlying expressions for noisy data. The model needs to search among a vast possible expression set and utilize domain specific knowledge to find solution.

Models. We selected the DeepSeek-R1-Distill-Qwen model family for our experiment due to its strong reasoning capabilities and manageable size, which is critical for performing complex scientific tasks under computational constraints. Among the model family, the 14B version offers an optimal balance between efficiency and performance, and was selected as the model in the main results. For physics simulation tasks that require strong geometric reasoning ability and physical prior knowledge, we utilize the 32B version of the model.

Baselines. We compare our approach against three key baselines:

1. **Direct Prompt**: Queries the model directly and selects the best outcome from 64 samples to establish a performance upper bound of base model.
2. **Open Evolve** (Sharma, 2025): An open-source implementation of the AlphaEvolve (Novikov et al., 2025) framework, which uses an evolutionary algorithm with multiple LLM roles (e.g., proposing code mutations, evaluating fitness) to iteratively generate, test, and evolve code or solutions across generations.
3. **Task-Specific Methods**: Represents results from established algorithms designed for each specific problem. Details of these methods can be found in Appendix C.

4.2 MAIN RESULTS

Table 1 presents the results of our methods compared to various baselines. The best results in each task are highlighted in **bold**. Since we selected multiple heterogeneous tasks, their evaluation metrics are not the same. The detailed definitions and specific evaluation criteria are deferred to Appendix B.

Across the 19 benchmark tasks, our method achieves the best performance on **16** tasks, surpassing all competing baselines. Compared under the same model settings, our framework consistently outperforms Direct Prompting across **all** benchmarks. Against OpenEvolve—the open-source version of AlphaEvolve—it achieves superior results on **18** tasks. These results clearly highlight the strength

Table 1: Results of main experiments. All values correspond to the best outcome obtained across all attempts. We use ↑ to indicate that larger values correspond to better performance, and ↓ represents the opposite. We highlighted the best results in each task in **bold**. "NA" denotes non-convergence or unsuitability for given case.

		Task Specific Methods		Direct Prompt		Open Evolve		Ours
Machine Learning	**Tasks**	LightGBM	RRL	Qwen	GPT-4o	Qwen	GPT-4o	-
	Adult Income ↑	80.36	80.72	73.72	76.91	76.90	72.27	**82.07**
	Bank Marketing ↑	75.28	76.32	0.00	76.91	75.66	78.54	**80.65**
	Boston Housing ↓	3.258	3.966	3.149	3.031	2.937	2.937	**1.747**
Physics Simulation	**Tasks**	Parameter Scan	Topology Opt	Qwen	GPT-4o	Qwen	GPT-4o	-
	Inductor ↑	6.111	6.248	2.584	0.001	1.637	1.652	**9.609**
	Beam Bending ↑	4.771	NA	5.407	4.005	10.793	6.352	**17.298**
	Magnetic Torque ↑	10.273	NA	0.323	1.201	3.488	1.607	**11.045**
	Periodic Heat ↑	1.206	NA	1.258	1.255	1.233	1.266	**1.278**
	Demultiplexer ↑	18.322	**23.555**	3.364	4.532	12.341	8.645	14.260
Circle Packing	**Tasks**	SLSQP	Genetic Algo	Qwen	GPT-4o	Qwen	GPT-4o	-
	Packing in Unit Square ↑	2.519	2.345	1.673	1.900	1.586	2.611	**2.636**
	Packing in Unit Disk ↑	4.522	3.896	4.608	3.290	4.604	3.984	**4.664**
Function Minimization	**Tasks**	SLSQP	Trust-constr	Qwen	GPT-4o	Qwen	GPT-4o	-
	Eggholder ↑	0.705	0.688	**1.000**	0.959	**1.000**	**1.000**	**1.000**
	Mishras Bird ↑	0.814	0.764	**1.000**	0.996	**1.000**	**1.000**	**1.000**
	Keanes Bump 10d ↑	0.714	0.692	0.886	0.987	**1.000**	0.997	**1.000**
	Keanes Bump 20d ↑	0.603	NA	0.794	0.657	0.596	0.983	**1.000**
	Keanes Bump 30d ↑	0.594	NA	0.923	0.625	0.677	0.668	**0.994**
Symbolic Regression	**Tasks**	LLM-SR	LaSR	Qwen	GPT-4o	Qwen	GPT-4o	-
	Chemistry ↓	4.12e-6	9.11e-5	2.66e-5	**2.44e-6**	1.59e-5	9.52e-6	7.32e-6
	Biology ↓	3.06e-6	1.53e-4	1.26e-4	7.52e-5	1.64e-4	5.31e-5	**2.98e-8**
	Physics ↓	7.62e-5	9.94e-4	2.71e-4	1.13e-4	2.76e-5	1.22e-4	**2.76e-5**
	Material Science ↓	**3.21e-9**	9.23e-6	7.14e-6	1.85e-6	6.99e-7	1.94e-6	4.46e-6

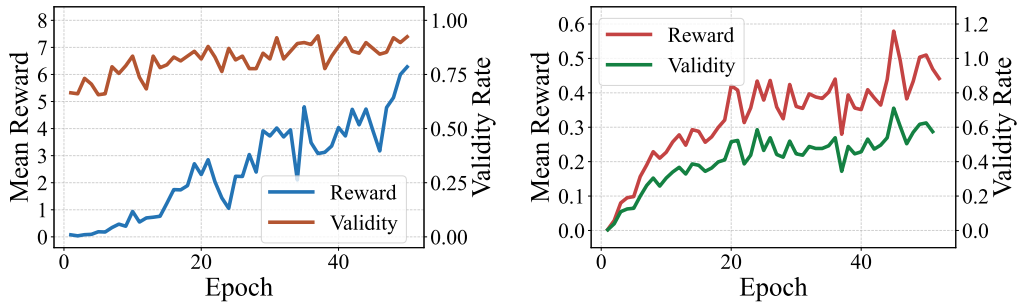
of our framework in solving open-ended scientific problems among various domains compared to other approaches.

Notably, we observe that the base Qwen models perform relatively poorly on certain tasks such as Bank Marketing and Magnetic Torque, exhibiting low rewards even in the best of 64 direct trials. However, our framework significantly improves performance in these cases by leveraging parameter updates and in-context learning to effectively incorporate feedback from the exploration process. This demonstrates that our approach can partially overcome the limitations of weaker base models by iteratively evolving toward superior solutions.

To further assess the competitiveness of our approach against state-of-the-art scientific discovery systems, we compared it with GPT-4o, one of the most advanced closed-source models. Remarkably, our method outperforms GPT-4o on **17** tasks, regardless of whether GPT-4o is equipped with multi-role collaborative reasoning frameworks. These results highlight that our framework can fully exploit the prior knowledge of smaller models through reinforcement learning, enabling cost-efficient and effective solutions to complex scientific problems.

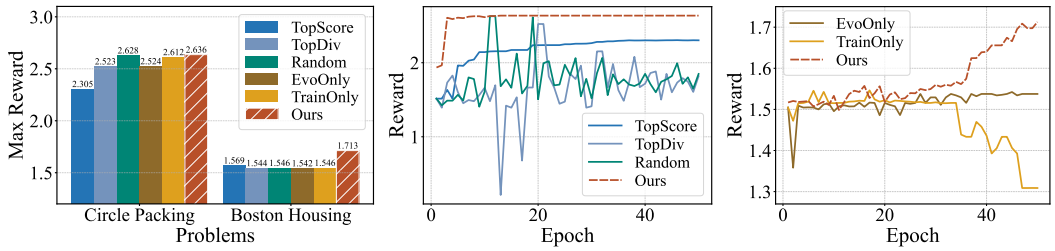
In comparison with task-specific methods, which are typically crafted by human experts for particular domains, our framework still achieves superior performance on **16** tasks. Specifically, in the circle packing task, we establish a new world record 2.635983 using only a 14B model. This highlights its ability to iteratively evolve within open-ended solution spaces and to autonomously uncover novel solutions that go beyond manually designed approaches.

To provide further evidence that our framework effectively integrates reinforcement learning and evolutionary algorithms, we analyze its convergence behavior on two representative cases: inductor design and adult income prediction. Figure 3 plots the average reward and validity of model outputs during training. Both metrics exhibit a clear upward trend: the validity rate rises steadily, showing that the model increasingly generates outputs that satisfy task constraints, while the average reward improves, reflecting higher-quality solutions. This dual improvement demonstrates that reinforcement learning progressively strengthens the model's intrinsic reasoning ability. It also indicates that



(a) Inductor design (b) Adult income prediction

Figure 3: Convergence analysis on the Inductor and Adult tasks. The curves show the progressive improvement of average reward and validity during training, demonstrating that our framework effectively leverages reinforcement learning feedback and evolutionary dynamics to produce increasingly valid and high-quality solutions.



(a) Max reward of varients (b) Circle Packing (c) Boston Housing

Figure 4: Ablation analysis of framework components. **(a):** Maximum reward achieved by different ablation variants. **(b):** Curve of epoch-wise maximum reward on the Circle Packing task, highlighting the critical role of balancing diversity and quality for stable optimization. **(c):** Curve of epoch-wise maximum reward on the Boston Housing task, showing the necessity of combining reinforcement learning with evolutionary guidance.

the quality of the evolving population keeps improving, enabling the model to leverage in-context feedback as well as intuitions from high-reward solutions to generate better outputs.

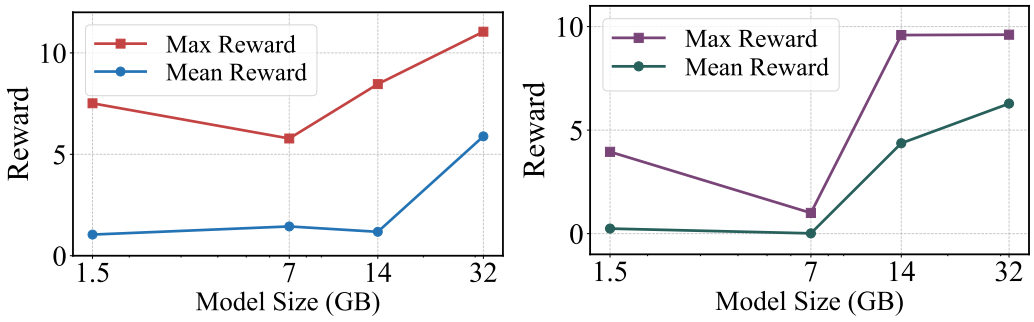
4.3 Ablation Study

4.3.1 Effectiveness of Framework Components

To better understand the contribution of each component in our framework, we conduct ablation studies on the Boston Housing and Circle Packing tasks. We design several controlled variants by selectively disabling or simplifying parts of the algorithm: **TopScore**, where only the highest-reward candidate in the dataset is selected for further evolution; **TopDiv**, where selection relies solely on diversity without considering reward; **Random**, where candidates are sampled randomly from the population; **EvoOnly**, where the model parameters are kept fixed and only the evolutionary pipeline is applied; and **TrainOnly**, which removes the evolutionary mechanism and in-context prompting, reducing the framework to pure GRPO reinforcement learning. These variants allow us to disentangle the relative importance of reward-driven selection, diversity maintenance, evolutionary population updates, and reinforcement learning in driving overall performance.

Figure 4a reports the maximum reward achieved under different ablation settings. Across both tasks, all variants perform worse than our full framework, confirming the necessity of each component. We next analyze the results task by task.

For the **Circle Packing** problem, high-quality solutions rely on diverse initial starting points for optimization algorithms. As shown in Figure 4b, eliminating diversity (TopScore) significantly re-



(a) Magnetic torque maximization

(b) Inductor design

Figure 5: Scaling analysis of model parameter scale on **(a):** magnetic torque maximization and **(b):** inductor design tasks.

duces reward, since the search quickly collapses into narrow solution modes. In contrast, Random and TopDiv maintain higher diversity, enabling the model to extend from a richer set of initial states. However, focusing solely on diversity also leads to instability—visible in the large variance of TopDiv and Random—whereas TopScore and our full method (Ours) remain relatively stable. This instability disrupts training and prevents the model from finding strong solutions in later epochs. These results highlight that balancing diversity and solution quality is critical for solving such problems.

For the **Boston Housing** task, strong performance requires careful parameter tuning and complex feature engineering, which typically emerge from iteratively learning from past experience. As shown in Figure 4c, disabling either reinforcement learning or evolution severely limits performance. With EvoOnly, the model remains bounded by its initial capacity and fails to break through training bottlenecks. Conversely, with TrainOnly, the model cannot effectively accumulate knowledge in context and collapses during training. These results demonstrate that both parameter updates and in-context evolutionary guidance are indispensable for helping the model accumulate expertise and progressively refine its solutions.

4.3.2 Scaling Experiments

Here, we discuss the impact of base model size on task performance. We evaluate our framework on two representative tasks, Magnetic Torque Maximization and Inductor Design, using the DeepSeek-R1-Distill-Qwen model family with 1.5B, 7B, 14B, and 32B parameters. As shown in Figure 5, for the magnetic torque task, the reward steadily increases with model size, indicating stronger reasoning ability and more effective exploration. For the inductor design task, we observe a reward plateau around 9.6. However, the mean reward continues to grow as model size increases, suggesting that larger models generate more valid and higher-quality candidates. These results demonstrate that our framework exhibits scaling property: as the underlying LLM grows, the system can push the boundaries of scientific discovery by enabling more efficient and higher-quality exploration.

5 Conclusion

In this work, we proposed HELIX, a hierarchical evolutionary reinforcement learning framework with in-context experiences. By integrating reinforcement learning, evolutionary selection, and in-context trial incorporation, HELIX effectively balances exploration and exploitation, enables task-specific adaptation, and iteratively refines solutions. Extensive experiments across 19 tasks in five diverse categories demonstrate that HELIX consistently outperforms strong task-specific baselines and advanced proprietary models. Overall, HELIX shows strong potential for advancing open-ended scientific discovery by enabling iterative, diversity-aware exploration. Looking ahead, it could provide a foundation for broader applications in engineering, optimization, and autonomous research systems.

ETHICS STATEMENT

This work focuses on developing a hybrid reinforcement learning and evolutionary framework for solving complex scientific problems. It does not involve human subjects, sensitive personal data, or proprietary datasets, and thus raises no direct ethical or privacy concerns. All datasets used are publicly available and widely adopted in prior research. All authors have reviewed and agree to abide by the ICLR Code of Ethics as linked above, and affirm that this submission complies with the principles of honesty, transparency, fairness, and responsible conduct.

REPRODUCIBILITY STATEMENT

Detailed descriptions of the experimental setup, task definitions, and evaluation metrics are provided in Appendix A and Appendix B.

Source code will be available at `https://anonymous.4open.science/r/HELIX-1829/`.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, et al. Gepa: Reflective prompt evolution can outperform reinforcement learning. *arXiv preprint arXiv:2507.19457*, 2025.

Tasnim Ahmed and Salimur Choudhury. Lm4opt: Unveiling the potential of large language models in formulating mathematical optimization problems. *INFOR: Information Systems and Operational Research*, 62(4):559–572, 2024.

Oliver A Bauchau and James I Craig. Euler-bernoulli beam theory. In *Structural analysis*, pp. 173–221. Springer, 2009.

Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

COMSOL AB. Comsol multiphysics®, 2024. URL `www.comsol.com`.

Andrew R Conn, Nicholas IM Gould, and Philippe L Toint. *Trust region methods*. SIAM, 2000.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, et al. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2): 182–197, 2002.

E Fan, Weizong Wang, and Tianhan Zhang. Chatcfd: an end-to-end cfd agent with domain-specific structured thinking. *arXiv preprint arXiv:2506.02019*, 2025.

Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. Retool: Reinforcement learning for strategic tool use in llms. *arXiv preprint arXiv:2504.11536*, 2025.

Ali Forootani. A survey on mathematical reasoning and optimization with large language models. *arXiv preprint arXiv:2503.17726*, 2025.

Bowen Gao, Yanwen Huang, Yiqiao Liu, Wenxuan Xie, Wei-Ying Ma, Ya-Qin Zhang, and Yanyan Lan. Pharmagents: Building a virtual pharma with large language model agents. *arXiv preprint arXiv:2503.22164*, 2025.

Arya Grayeli, Atharva Sehgal, Omar Costilla Reyes, Miles Cranmer, and Swarat Chaudhuri. Symbolic regression with a learned concept library. *Advances in Neural Information Processing Systems*, 37:44678–44709, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

David Harrison Jr and Daniel L Rubinfeld. Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management*, 5(1):81–102, 1978.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Robert Tjarko Lange, Aaditya Prasad, Qi Sun, Maxence Faldor, Yujin Tang, and David Ha. The ai cuda engineer: Agentic cuda kernel discovery, optimization and composition. Technical report, Technical report, Sakana AI, 02 2025, 2025.

Charles L Lawson and Richard J Hanson. *Solving least squares problems*. SIAM, 1995.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

Xianggen Liu, Yan Guo, Haoran Li, Jin Liu, Shudong Huang, Bowen Ke, and Jiancheng Lv. Drugllm: Open large language model for few-shot molecule generation. *arXiv preprint arXiv:2405.06690*, 2024.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

S. Moro, P. Rita, and P. Cortez. Bank Marketing. UCI Machine Learning Repository, 2014. DOI: https://doi.org/10.24432/C5K306.

Yves Gaetan Nana Teukam, Federico Zipoli, Teodoro Laino, Emanuele Criscuolo, Francesca Grisoni, and Matteo Manica. Integrating genetic algorithms and language models for enhanced enzyme design. *Briefings in bioinformatics*, 26(1):bbae675, 2025.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Thang D Pham, Aditya Tanikanti, and Murat Keçeli. Chemgraph: An agentic framework for computational chemistry workflows. *arXiv preprint arXiv:2506.06363*, 2025.

ZZ Ren, Zhihong Shao, Junxiao Song, Huajian Xin, Haocheng Wang, Wanjia Zhao, Liyue Zhang, Zhe Fu, Qihao Zhu, Dejian Yang, et al. Deepseek-prover-v2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. *arXiv preprint arXiv:2504.21801*, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Asankhaya Sharma. Openevolve: an open-source evolutionary coding agent, 2025. URL `https://github.com/codelion/openevolve`.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan K Reddy. Llm-sr: Scientific equation discovery via programming with large language models. *arXiv preprint arXiv:2404.18400*, 2024.

Parshin Shojaee, Ngoc-Hieu Nguyen, Kazem Meidani, Amir Barati Farimani, Khoa D Doan, and Chandan K Reddy. Llm-srbench: A new benchmark for scientific equation discovery with large language models. *arXiv preprint arXiv:2504.10415*, 2025.

Niki van Stein and Thomas Bäck. Llamea: A large language model evolutionary algorithm for automatically generating metaheuristics. *IEEE Transactions on Evolutionary Computation*, 2024.

Zhuo Wang, Wei Zhang, Ning Liu, and Jianyong Wang. Scalable rule-based representation learning for interpretable classification. *Advances in Neural Information Processing Systems*, 34:30479–30491, 2021.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

A Training and Evaluation Details

Training. We primarily use DeepSeek-Distill-Qwen-14B and 32B as the backbone models in our experiments. The models are fine-tuned with the VERL framework (Sheng et al., 2024) under the GRPO algorithm. Each model is trained for 80 epochs with a fixed learning rate of 1×10^{-6}, updating all parameters. We set the KL coefficient in GRPO to 1×10^{-3} and the number of rollouts to 16. The rollouts are generated via VLLM (Kwon et al., 2023) backend with temperature equals to 1.0 and top_p equals to 0.95. Training was conducted using eight A100 GPUs for 14B models and sixteen H100 GPUs for 32B models. For training efficiency, we use Pytorch FSDP (Zhao et al., 2023) with parameter offload and optimizer offload. Gradient checkpoint and Flash-Attention (Dao, 2024) are used by default.

Evaluation. The evaluation is performed on a Slurm Workload Manager system. For each job, we allocate 4 Intel(R) Xeon(R) Platinum 8168 CPUs for execution and impose time limits for each task: five minutes for physics simulation, two minutes for machine learning and function minimization, and one minute for circle packing and symbolic regression. The execution time includes the time for task-dependent evaluators to calculate reward. For the detailed evaluate metric and reward calculation, please refer to Appendix B.

B Definition and Evaluation of Problems

In this section we explain the detailed problem definition and evaluation metrics of all the tasks used in the experiment.

B.1 Physics Simulation

To test the model's capacity for geometric reasoning and ability to utilize physics prior knowledge to discover better designs, we proposed the following physics simulation tasks. These tasks mainly require the model to generate a yaml representation of a complex geometry under certain constraints to maximize the reward. We utilize COMSOL Multiphysics® (COMSOL AB, 2024), a commercial FEA software for industrial multiphysics simulations, for the evaluation backend.

B.1.1 Acoustic Demultiplexer

This task aims to design an acoustic demultiplexer. The demultiplexer is a data distributing device which takes acoustic energy from the input port and distributes different frequency bands to the specific output port. The model is asked to propose the cavity geometry within a circular domain as seen in Fig. 6 to maximize the acoustic pressure at output port 2 while minimizing the pressure at output port 3. The input acoustic pressure level is set to 1 Pa at port 1, and the frequency level is set to 7500 Hz.

The model is guided by the following reward R where P_i is the power output at port i, p_{rms} is the Root Mean Square (RMS) pressure field, ρ is fluid density, and c is sound speed.

$$\begin{aligned} P &= \int_{port} \frac{p_{rms}^2}{\rho c} dl \\ R &= \frac{log_{10}(P_2) - log_{10}(P_3)}{0.292} \end{aligned} \tag{6}$$

We use a value of 0.292 on the denominator of Eq. 6 to normalize the reward. And Fig. 6 shows a symmetric design with 7 circular cavities in the computation domain, producing equal acoustic pressure at the two output ports and thus $R = 0$. Notice that LLM is not limited by the circular cavity pattern, and is prompted to freely explore any viable cavity geometries within the computation domain.

B.1.2 Magnetic Torque

This task aims to design the geometry of an iron core that generates large torque when subjected to a uniform magnetic field. Fig. 7 shows the problem setting, an example iron core geometry

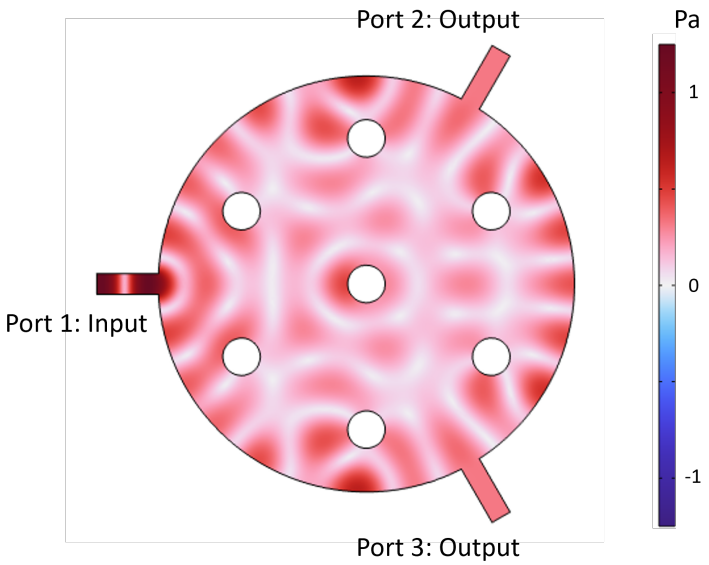


Figure 6: The RMS pressure field of an acoustic demultiplexer at frequency level 7500 Hz. The RMS pressure field in log scale is proportional to the acoustic power.

and the corresponding magnetic flux density norm field. A uniform magnetic field intensity of $\mathbf{H} = [0, 1e5]$ A/m is applied to the circular boundary. The iron core possesses a large permeability $\mu \gg \mu_0$ distorts the magnetic flux density field $\mathbf{B}$ within the circular air domain. The distorted $\mathbf{B}$ thus applies a torque on the iron core, which can be obtained from Comsol by solving the static Maxwell's equations.

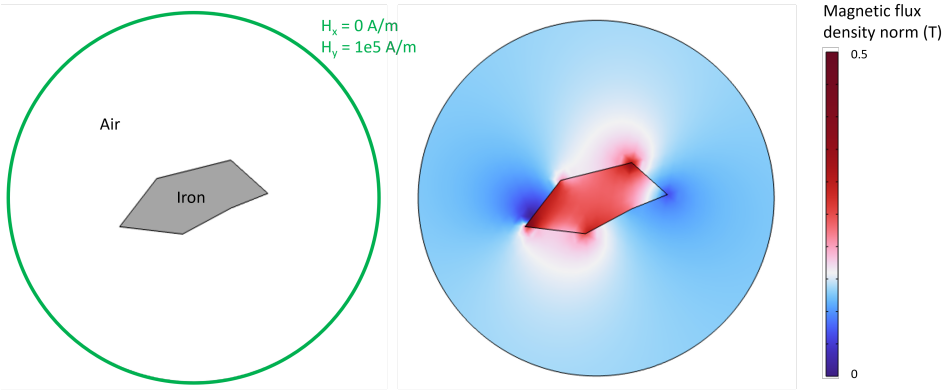


Figure 7: The magnetic flux density field generated by an iron core subject to a uniform magnetic field boundary condition. The distorted magnetic flux density field then applies a torque on the iron core.

To guide the model reinforcement learning and evolutionary search, the following reward R is computed as below where $\mathbf{T}$ is Maxwell stress tensor, $\mathbf{r}$ is position vector, and $\boldsymbol{\tau}$ represents magnetic torque which is simplified to τ_z in 2D simulations:

$$\begin{aligned} \mathbf{T} &= \frac{1}{\mu_0}(\mathbf{BB} - \frac{1}{2}B^2\mathbf{I}) \\ R = \frac{||\boldsymbol{\tau}||}{9241.99 \cdot A} &= \frac{1}{9241.99 \cdot A}||\int_S \mathbf{r} \times (\mathbf{T} \cdot \hat{\mathbf{n}})dA|| \end{aligned} \tag{7}$$

We use a value of 9241.99 on the denominator of Eq. 7 to normalize the reward. Notice that a perfectly symmetric iron core (for instance a circle) would have $\tau_z = 0$. Therefore, we expect to train and evolve the LLM to produce a highly irregular iron core geometry to generate large magnetic torque values. We set a minimum area of $2e^{-4}$ m^2 to avoid naive designs.

B.1.3 Beam Bending

This task aims to design the cross section geometry of a cantilever beam subject to a superposed loading of bending moments M_x and M_y, shear forces T_x and T_y along the two in-plane directions, and twisting moment T_z along the out-of-plane direction. The cantilever beam is assumed to be linear elastic with Young's modulus 1 GPa and Poisson's ratio 0.3. Fig. 8 shows an example beam cross section design and the von Mises stress distribution as calculated from Eq. 8, solved using the Beam Cross Section module in Comsol. As the cross section stays in the x-y plane, σ_{xx}, σ_{yy}, and τ_{xy} take 0 values.

$$\sigma_{vm} = \sqrt{\frac{1}{2}[(\sigma_{xx} - \sigma_{yy})^2 + (\sigma_{xx} - \sigma_{zz})^2 + (\sigma_{yy} - \sigma_{zz})^2] + 3 \cdot (\tau_{xy}^2 + \tau_{xz}^2 + \tau_{yz}^2)} \tag{8}$$

The reward is set to be $R = \frac{I_1^{0.8} \cdot I_2^{0.2}}{1.32e^{-3} \cdot A}$ where A is the cross section area, I_1 is the largest second moment of inertia, I_2 is the smallest second moment of inertia. We use a value of $1.32e^{-3}$ on the denominator to normalize the reward. I_1 and I_2 represent the beam's largest and smallest resistance over different bending loading directions, and can be calculated from the stress field following the classical beam bending theory (Bauchau & Craig, 2009). We set a minimum area of $2e^{-3}$ m^2 to avoid naive designs.

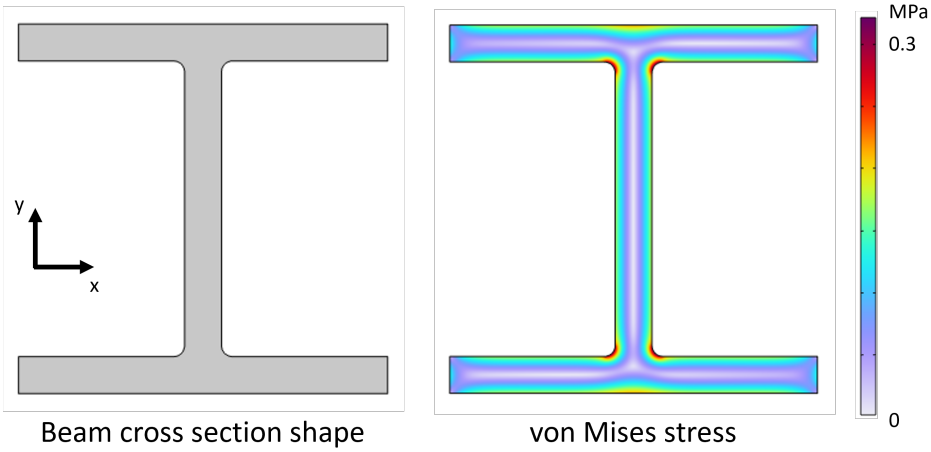


Figure 8: The von Mises stress field generated by applying bending moment, shear force, and twisting moment on a cantilever beam cross section design.

B.1.4 Periodic Heat

This task aims to design the unit cell geometry of a periodic meta-material for best effective thermal conductivity. The base material is assumed to be aluminum with density 2700 kg/m^3 and thermal conductivity 238 W/mK. Fig. 9 shows an example 2D unit cell geometry which will be extruded in the z direction to form the 3D unit cell. The resultant temperature distribution and effective properties are solved using Comsol based on the homogenization theory. The results are calculated from a 1 K temperature difference boundary conditions along x, y, and z directions.

$$R = \frac{trace(\mathbf{k}_{eff})}{0.178 \cdot \rho_{eff}} \tag{9}$$

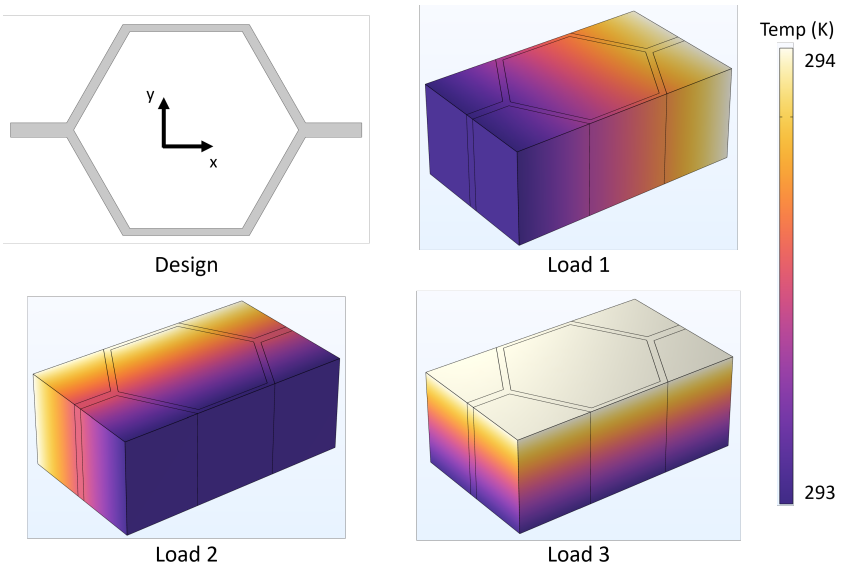


Figure 9: Temperature distribution of the meta-material under three loading conditions. The effective properties are calculated based on temperature distributions according to the homogenization theory.

where $\mathbf{k}_{eff}$ is the homogenized effective thermal conductivity matrix, and ρ_{eff} is the effective density, which simply equals to the percentage of volume filled by aluminum. We use a value of 0.178 on the denominator to normalize the reward. This objective function targets to maximize the thermal conductivity along x, y, and z directions under limited material usage. We set a maximum effective density $\rho_{eff} \leq 2000$ kg/m^3 to avoid naive designs.

B.1.5 Inductor

This task aims to design an inductor which is a critical component in power electronics. Fig. 10 shows an example inductor consisting of an iron core and coil windings in a cylindrical coordinate. A sinusoidal current excitation is supplied to the coils at a frequency of 1000 Hz and magnitude 500 A. The iron core possesses a nonlinear magnetization curve with an initial permeability of 663 H/m and saturates at 5 T. The resultant magnetic field is calculated using Comsol by solving the Maxwell's equations in frequency domain. The model is asked to propose the optimal iron core geometry as well as the placement of the coil windings (coil shapes are fixed) to produce the maximum inductance with limited material usage.

$$R = \frac{L}{43.11 \cdot V} = \frac{0.5 \cdot \int_\Omega (B_r \cdot \overline{H_r} + B_\phi \cdot \overline{H_\phi} + B_z \cdot \overline{H_z}) dV}{43.11 \cdot V} \tag{10}$$

The reward calculation is shown in Eq. 10 where B_r, B_ϕ, and B_z are cylindrical components of magnetic flux density field, and H_r, H_ϕ, H_z are components of magnetic intensity field. Both fields take complex values for frequency domain response. We use a value of 43.11 on the denominator to normalize the reward. The numerator stands for the inductance which is a volume integral of magnetic energy. We set a minimum iron core volume of $1e^{-3}$ m^3 to avoid naive designs.

B.2 Circle Packing

The objective of these tasks is to pack a fixed number of circles in a specific domain and maximize the sum of the radii of these circles. The circles cannot overlap with each other or exceed the domain boundary. All the centers and radii can change as long as the constraints are satisfied.

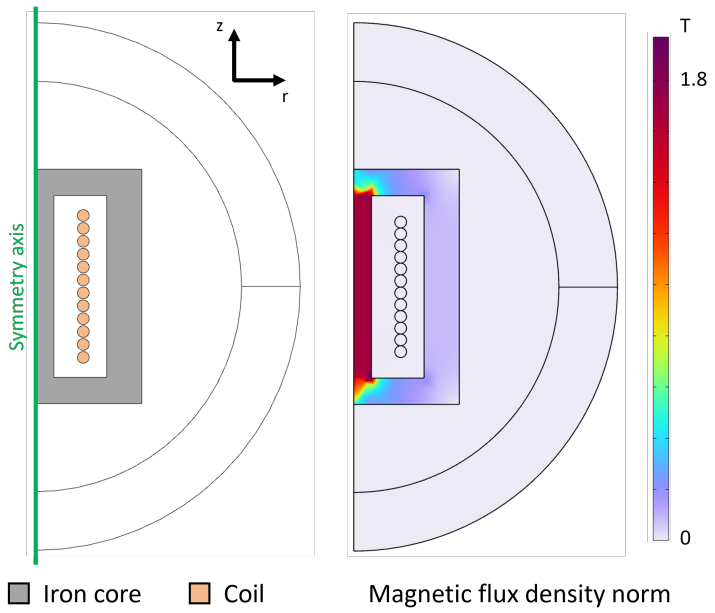


Figure 10: The magnetic flux density norm field generated by an inductor. The copper coils are excited by a 500 A, 1000 Hz sinusoidal current.

Formally, let $n = 26$ be the number of circles, $\{x_i\}_{i\leq n}, \{y_i\}_{i\leq n}$ be the coordinates of centers and $\{r_i\}_{i\leq n}$ be the radii. The objective can be written as:

$$R = \sum_{i=1}^{n} r_i, \tag{11}$$

while the constraint is

$$\begin{aligned} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} &\geq r_i + r_j, \quad \forall 1 \leq i < j \leq n \\ x_i - r_i &\geq 0, \quad \forall 1 \leq i \leq n \\ x_i + r_i &\leq 1, \quad \forall 1 \leq i \leq n \\ y_i - r_i &\geq 0, \quad \forall 1 \leq i \leq n \\ y_i + r_i &\leq 1, \quad \forall 1 \leq i \leq n, \end{aligned} \tag{12}$$

for the packing in a unit square, and

$$\begin{aligned} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} &\geq r_i + r_j, \quad \forall 1 \leq i < j \leq n \\ \sqrt{x_i^2 + y_i^2} + r_i &\leq 1, \quad \forall 1 \leq i \leq n, \end{aligned} \tag{13}$$

for the packing in a unit disk.

B.3 Function Minimization

These tasks require the model to find an effective algorithm to locate the global minimum of a complex function with various local minima. For a given function $f(\mathbf{x}^*)$ and the model's prediction $\hat{\mathbf{x}}^*$, The evaluation metric is defined as:

$$R = \frac{|f(\mathbf{x}^*)|}{|f(\mathbf{x}^*)| + |f(\hat{\mathbf{x}}^*) - f(\mathbf{x}^*)|}. \tag{14}$$

This metric is suitable for distinct functions with varying scales of $|f(\mathbf{x}^*)|$. It satisfies $0 \leq R \leq 1$ and if the model successfully finds the global minimum, the reward will be $R = 1.0$.

B.3.1 EGGHOLDER FUNCTION

The Eggholder function is a classical task for evaluating evolutionary optimization algorithms with various local minima. It can be defined as:

$$f(\mathbf{x}) = -(x_2 + 47)\sin(\sqrt{|(x_2 + 47) + \frac{x_1}{2}|}) - x_1 \sin(\sqrt{|x_1 - (x_2 + 47)|}), \tag{15}$$

with constraint $-512 \leq x_1, x_2 \leq 512$ and a global minimum $f((512, 404.2319)) \approx -959.6407$ under such constraint.

Figure 11 illustrates the landscape and the global minimum point of the Eggholder function.

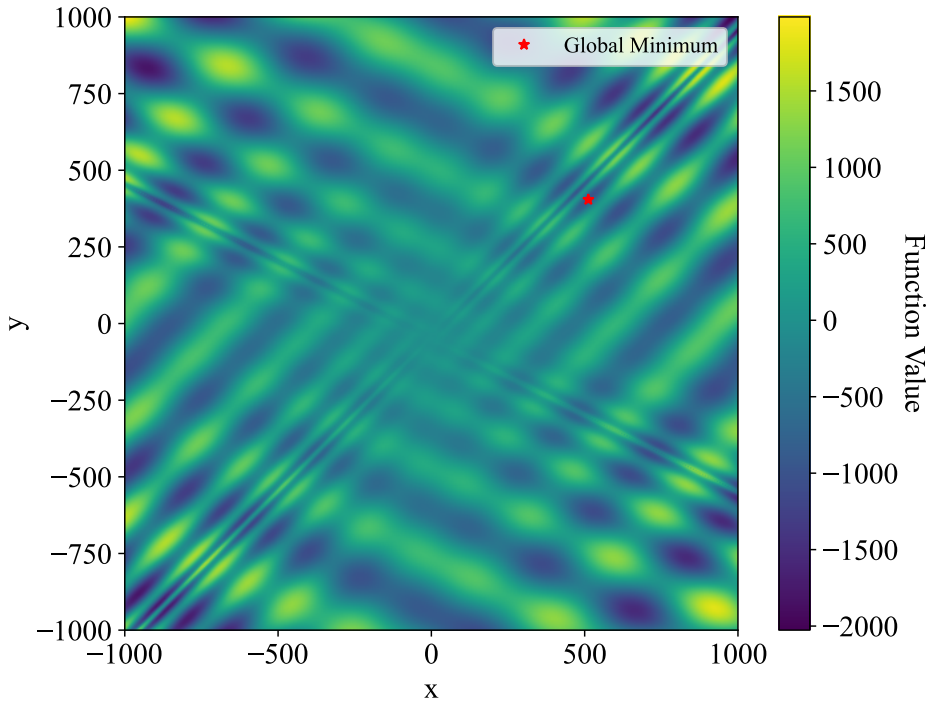


Figure 11: The landscape and global minimum point of Eggholder function with constraints $-512 \leq x, y \leq 512$

B.3.2 MISHRA'S BIRD FUNCTION

The Mishra's Bird function is a classic test function used in optimization to evaluate the performance of algorithms. It is known for having a unique "bird-shaped" landscape with multiple local minima and a single global minimum. It's often used to test an algorithm's ability to avoid getting stuck in suboptimal solutions.

The function is defined as:

$$f(\mathbf{x}) = \sin(x_2)e^{(1-\cos(x_1))^2} + \cos(x_1)e^{(1-\sin(x_2))^2} + (x_1 - x_2)^2 \tag{16}$$

with the constraints:

$$\begin{aligned} -10 \leq x_1 \leq 0 \\ -6.5 \leq x_2 \leq 0 \\ x_1^2 + x_2^2 \geq 25. \end{aligned} \tag{17}$$

The global minimum is $f((-3.1302, -1.5822)) \approx -106.7645$.

Figure 12 shows the landscape of the Mishra's Bird function and its global minimum point.

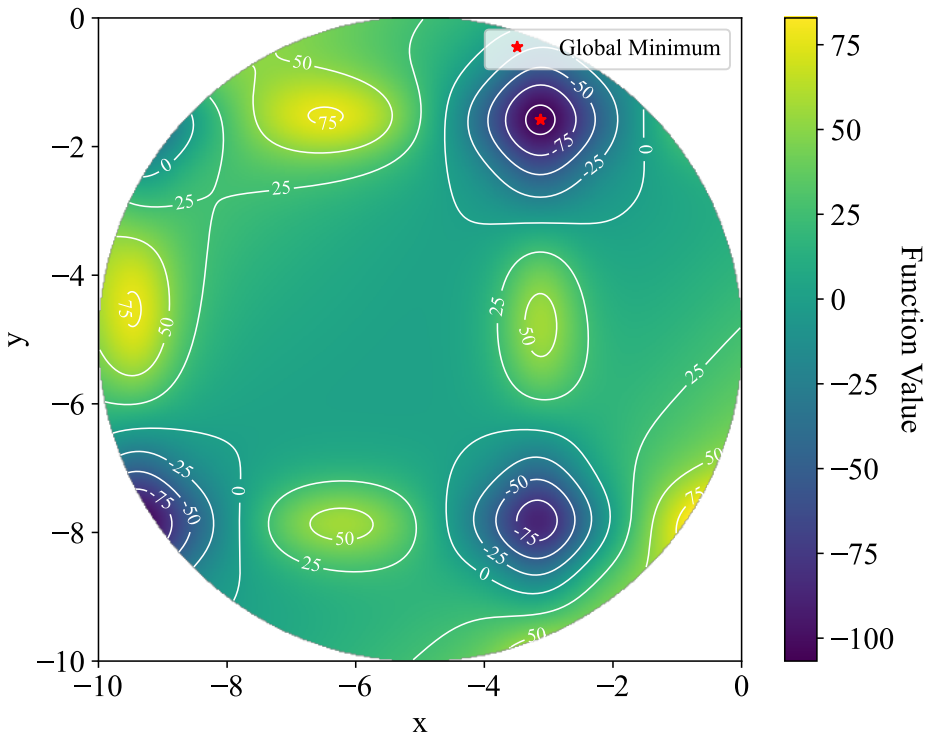


Figure 12: The landscape and global minimum point of Mishra's Bird function with constraints $x_1 \in [-10, 0]$, $x_2 \in [-6.5, 0]$ and $x_1^2 + x_2^2 \geq 25$.

B.3.3 KEANES BUMP FUNCTION

The Keanes Bump function is a challenging, non-convex test function commonly used to evaluate the performance of optimization algorithms in handling high-dimensional problems with complex constraints. The function's landscape is highly irregular, containing numerous local minima, and its feasible region is a small, irregular subset of the search space.

Let d be the dimension of variables and $f : \mathbb{R}^d \to \mathbb{R}$, the function is defined as:

$$f(\mathbf{x}) = \frac{-|\sum_{i=1}^{d} \cos^4(x_i) - 2\prod_{i=1}^{d} \cos^2(x_i)|}{\sqrt{\sum_{i=1}^{d} i x_i^2}} \tag{18}$$

with the following constraints:

$$\begin{aligned} &0 < x_i \leq 10, \qquad \forall 1 \leq i \leq d \\ &\sum_{i=1}^{d} x_i \leq 7.5d \\ &\prod_{i=1}^{d} x_i \geq 0.75. \end{aligned} \tag{19}$$

The global minimum is located within the feasible region, which is a small, bounded area defined by these constraints. The image in this document, Figure 13, shows a two-dimensional visualization of the function's landscape. However, for our experiments, we tested the function in its 10-D, 20-D, and 30-D versions, where the complexity increases significantly. The global minima and their corresponding function values for these dimensions are listed below.

- **10-D Version**: The global minimum value is approximately -0.747310362.

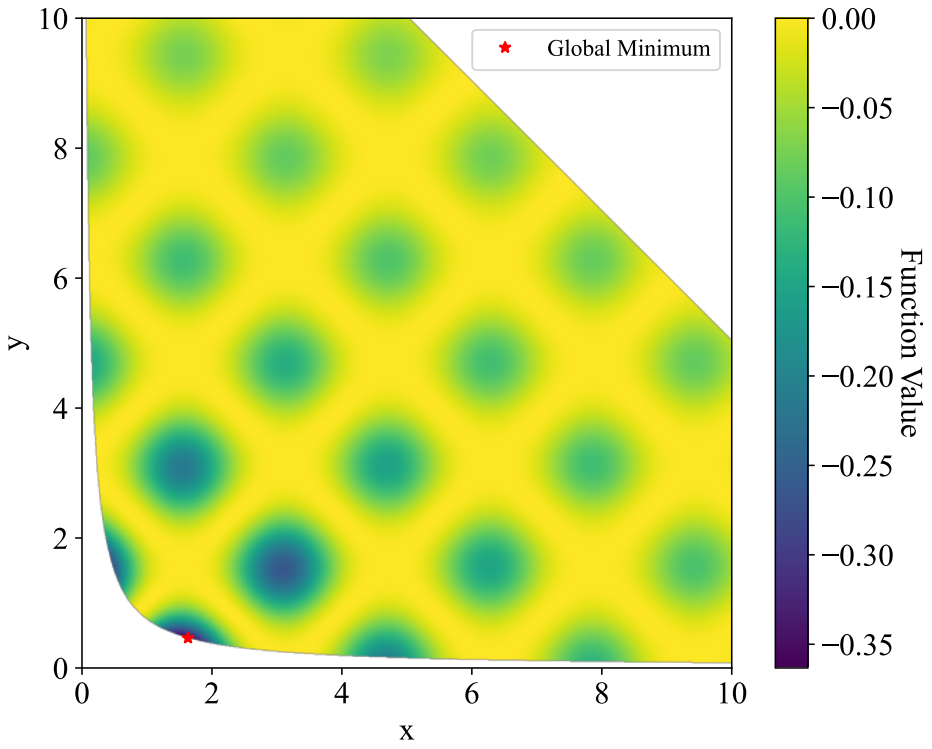


Figure 13: The landscape and global minimum point of the 2-D Keanes Bump function. The feasible region, a small part of the search space, is the only area with finite function values.

- **20-D Version**: The global minimum value is approximately -0.803619104.
- **30-D Version**: The global minimum value is approximately -0.818056222.

B.4 Symbolic Regression

In this task, the model has to uncover symbolic mathematical expressions from observational data. The benchmark and baselines are provided by Shojaee et al. (2025), which includes equations and data in chemistry, biology, physics and material science domains. In each category, several cases are created, each containing its own train and test sets generated by the same underlying equation. The model trained on the train set has to propose an expression to minimize the normalized mean square error (NMSE) on the test set, which is defined as:

$$\text{NMSE} = \frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}, \tag{20}$$

where N is the number of observations in the test set.

To ensure a fair and robust comparison with the benchmark paper's results, we use the median of the NMSE calculated across all tasks within the same category c:

$$\text{NMSE}_c = \text{median}(\text{NMSE}_{c,1}, \text{NMSE}_{c,2}, \ldots, \text{NMSE}_{c,n}). \tag{21}$$

The reward we used for reinforcement learning for category c is then set to:

$$R_c = -\log_{10}(\text{NMSE}_c). \tag{22}$$

In the benchmark, all the methods have a limit of 1000 trials for each single case, and we obey the same rule in our experiments, adjusting the number of training steps accordingly.

B.5 MACHINE LEARNING

We selected 3 classic machine learning datasets, and the model has to write Python code to maximize the F1 score for classification tasks and minimize the rooted mean square error (RMSE) for regression tasks. The details are described below.

B.5.1 ADULT INCOME

The Adult income dataset (Becker & Kohavi, 1996) is a well-known binary classification task. The goal is to predict whether a person's income exceeds \$50,000 per year based on various demographic features such as age, education, marital status, and occupation. The dataset is sourced from the 1994 U.S. Census and contains both categorical and numerical features, with some missing values.

The dataset itself contains a separate train and test split. We then load the train set for model's training and evaluate its result on the test set. The reward is the Macro F1 score, defined as:

$$R = \frac{1}{C} \sum_{c=1}^{C} \frac{2 \cdot P_c \cdot R_c}{P_c + R_c} \tag{23}$$

where P_c, R_c are the precision and recall for class c, and $C = 2$ is the total number of classes.

B.5.2 BANK MARKETING

The Bank marketing dataset (Moro et al., 2014) is another binary classification problem. It includes data from a Portuguese bank's direct marketing campaigns, where the objective is to predict whether a client will subscribe to a term deposit. This dataset is characterized by a high number of categorical features and a significant class imbalance, making it a good benchmark for evaluating model performance under challenging real-world conditions.

To ensure a robust evaluation, we use a 5-fold cross-validation strategy with StratifiedKFold in sklearn to handle the class imbalance. The data is randomly split into five folds, maintaining the same class distribution in each fold as in the original dataset. The model is trained and evaluated five times, with each fold serving as the test set once. The final reward is the average of the Macro F1 scores obtained from all five folds. If a task fails to produce a result in any fold, its reward is considered to be 0 for that fold. The final result is the Macro F1 score, as defined in equation 23.

B.5.3 BOSTON HOUSING

The Boston housing dataset (Harrison Jr & Rubinfeld, 1978) is a classic regression problem. The task is to predict the median value of owner-occupied homes in Boston suburbs, based on 13 features. These features include per capita crime rate, a number of rooms per dwelling, and the proportion of non-retail business acres. While the original dataset is no longer widely used for research due to ethical concerns, it remains a common benchmark for teaching and evaluating regression models.

To evaluate model performance, we use a 5-fold cross-validation strategy with KFold, splitting the data into five folds. The model is trained and evaluated five times, with each fold serving as the test set once. The final reward for this task is the average of the scores from all five folds. The reward is calculated using the following formula:

$$R = 2 - \log_{10}(\text{RMSE} + 10^{-10}), \tag{24}$$

where:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \tag{25}$$

This reward metric is designed to penalize larger RMSE values while rewarding smaller ones. If a task fails in any fold, its reward is considered to be 0 for that fold.

C DESCRIPTION OF TASK SPECIFIC BASELINES

In this section, we introduce the task-specific baseline methods and describe their implementation details.

Physics Simulation. For physics-related optimization problems, we use two widely adopted modules in COMSOL Multiphysics: parameter search and topology optimization. For parameter search, we first parameterize the geometry based on initial solutions provided by human experts, and then optimize within the search space defined by these parameters. For topology optimization, human experts specify deformable geometric regions, while COMSOL applies its built-in topology optimization solvers to iteratively refine the structure.

Circle Packing. We consider two strong baselines: Sequential Least Squares Programming (SLSQP) (Lawson & Hanson, 1995) and a Genetic Algorithm (GA). SLSQP formulates circle packing as a constrained optimization problem, maximizing the sum of radii subject to boundary and non-overlap constraints. The GA baseline encodes circle positions and radii, evolves a feasible population with selection, crossover, and mutation, and evaluates fitness by the total radii.

Function Minimization. We adopt two standard constrained optimization solvers from `scipy.optimize`: Sequential Least Squares Programming (SLSQP) and the trust-constr method (Conn et al., 2000). Both are widely used gradient-based methods that provide strong task-specific baselines for function minimization.

Symbolic Regression. For symbolic regression tasks, we directly use results reported in LLM-SRBench (Shojaee et al., 2025), obtained by GPT-4o-mini running two recent methods: LaSR (Grayeli et al., 2024), which enhances evolutionary search with LLM-guided concept discovery, and LLM-SR (Shojaee et al., 2024), which combines LLM scientific priors with evolutionary equation search. These represent competitive state-of-the-art baselines for symbolic regression.

Machine Learning. For machine learning benchmarks, we evaluate two interpretable yet competitive models: LightGBM (Ke et al., 2017), a gradient boosting framework widely adopted in practice, and Rule-based Representation Learner (RRL) (Wang et al., 2021), which learns discrete non-fuzzy rules via gradient grafting to achieve both scalability and interpretability.

D Example of model output

We present examples of the best solutions found by our framework across different task categories. These visualizations highlight how HELIX generates high-quality and interpretable outputs in diverse scientific domains.

D.1 Physics simulation tasks

Acoustic demultiplexer. Figure 14 displays the acoustic pressure field of our best-performing demultiplexer, which achieves a reward of 14.260.

Iron core torque optimization. The best iron core design is shown in Figure 15, where the magnetic flux density norm reaches a reward of 11.045.

Beam design. Figure 16 illustrates the von Mises stress pattern of the best beam structure discovered, which achieves a reward of 17.298.

Meta-material optimization. The temperature distributions of the optimized meta-material under two loading conditions are presented in Figure 17, yielding a reward of 1.278.

Inductor design. The optimized inductor is visualized in Figure 18, with a magnetic flux density norm field corresponding to a reward of 9.609.

D.2 Circle packing tasks

Packing in square. As shown in Figure 19, our framework successfully packs 26 circles inside a square, achieving a sum of radii of 2.635983 and surpassing the previous world record.

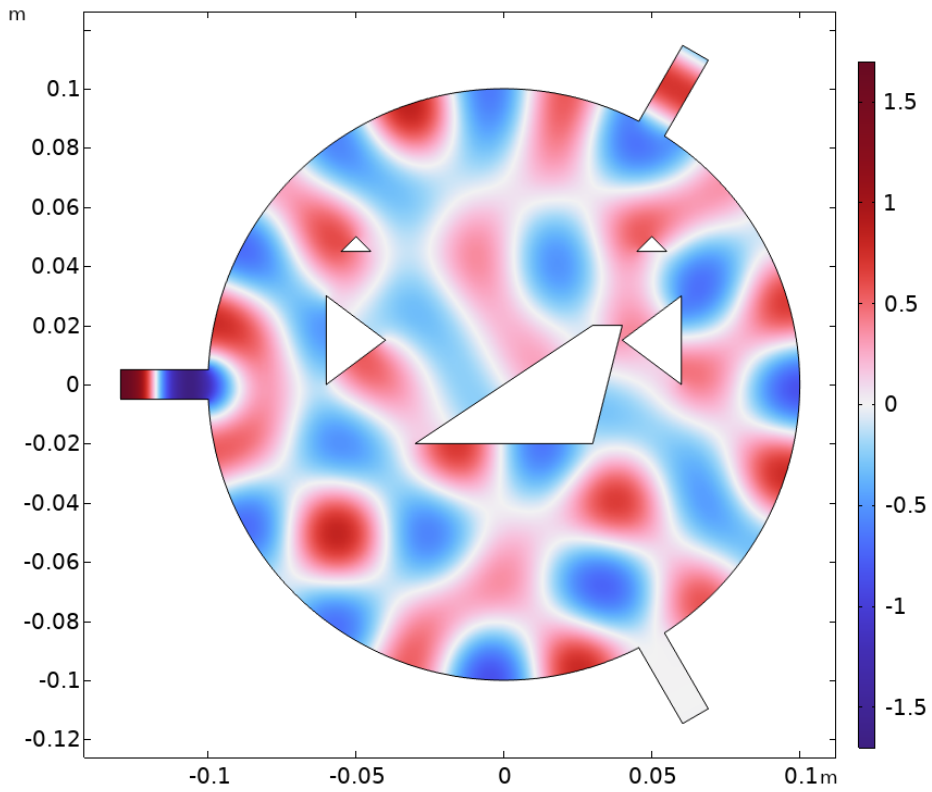


Figure 14: Acoustic pressure distribution in the optimal acoustic demultiplexer design obtained by our framework, with reward 14.260.

Packing in disk. Figure 20 demonstrates the packing of 26 circles inside a disk, reaching a total radius sum of 4.664465.

D.3 Machine learning tasks

We further demonstrate how our framework can be applied to classical machine learning problems, using both classification and regression benchmarks. The first example focuses on the Adult dataset, where we design a rich set of engineered features that combine polynomial transformations, ratios, interaction terms, and domain-specific indicators. This structured feature space, coupled with a LightGBM classifier and hyperparameter tuning, enables our model to achieve a strong performance of 82.07 in macro F1 score (Figure 21).

For regression, we turn to the Boston Housing dataset. Here, we integrate robust preprocessing with advanced feature transformations. Missing values in numeric features are imputed with KNN and scaled robustly, while categorical variables undergo smoothed target encoding. Additional interaction and polynomial features are then injected through a transformer pipeline. With these enhancements, our model coupled with an XGBoost regressor attains a reward of 1.742, corresponding to an RMSE of 1.813 (Figure 22).

E LLM usage statement

Large Language Models (LLMs) were used solely to aid writing and polishing the manuscript. All research ideas, experiments, and analyses were conceived and conducted by the authors, who take full responsibility for the content.

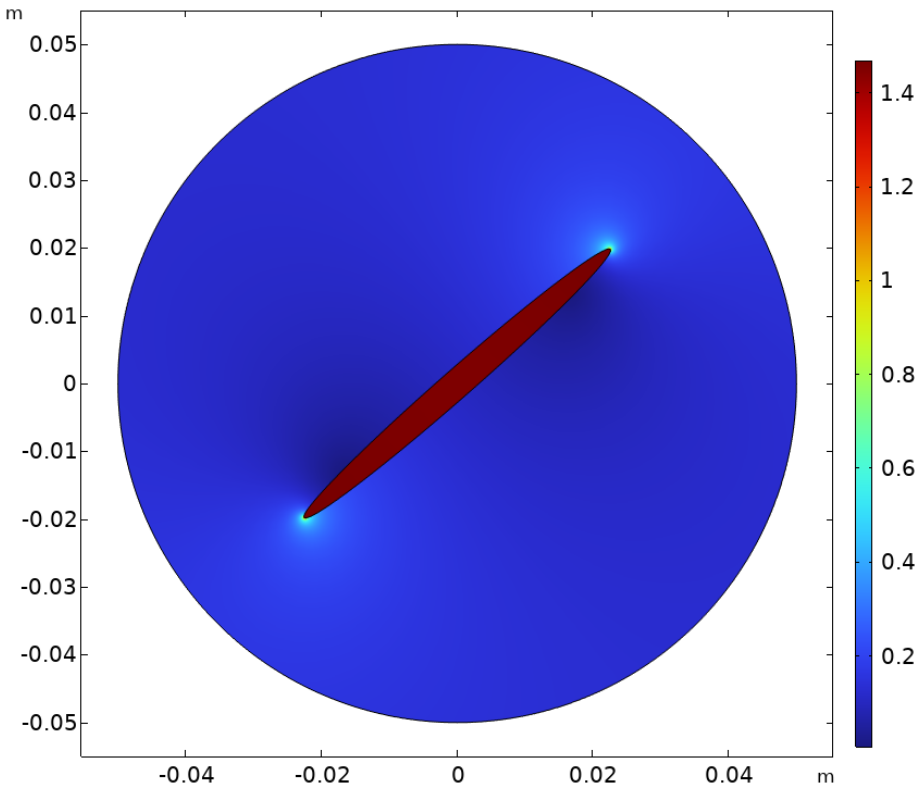


Figure 15: Magnetic flux density norm field for the optimized iron core configuration identified by our framework, achieving reward 11.045.

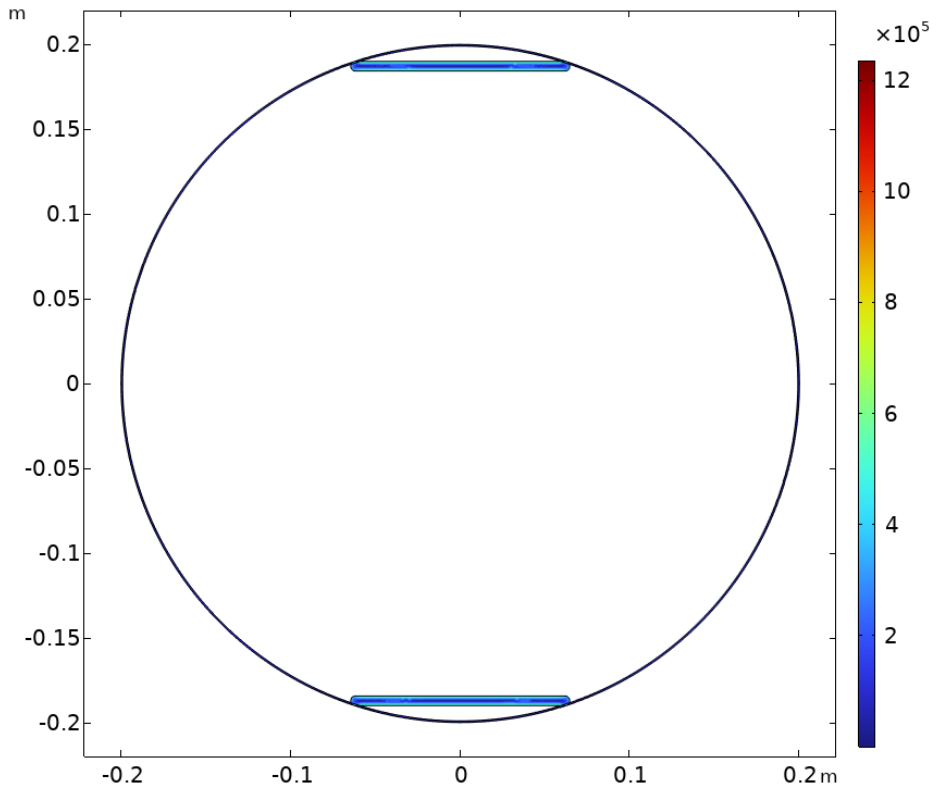
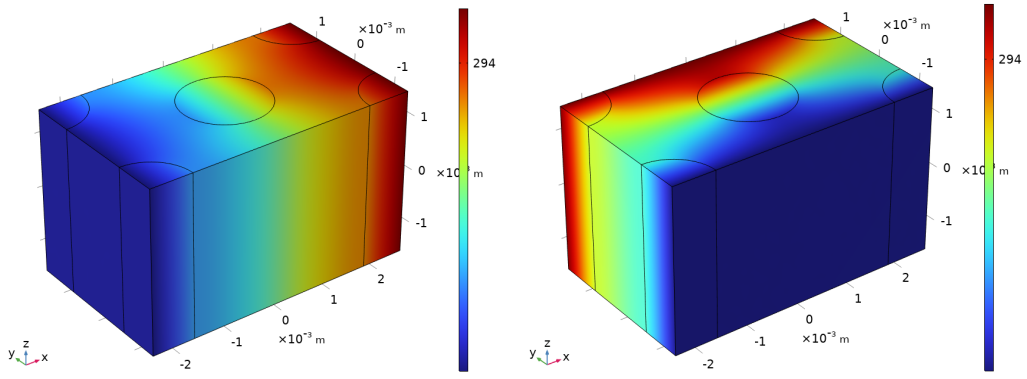


Figure 16: Von Mises stress distribution of the optimized beam design obtained by our framework, with reward 17.298.



(a) Load 1 (b) Load 2

Figure 17: Optimized temperature fields of the meta-material designed by our framework under two different load conditions, achieving reward 1.278.

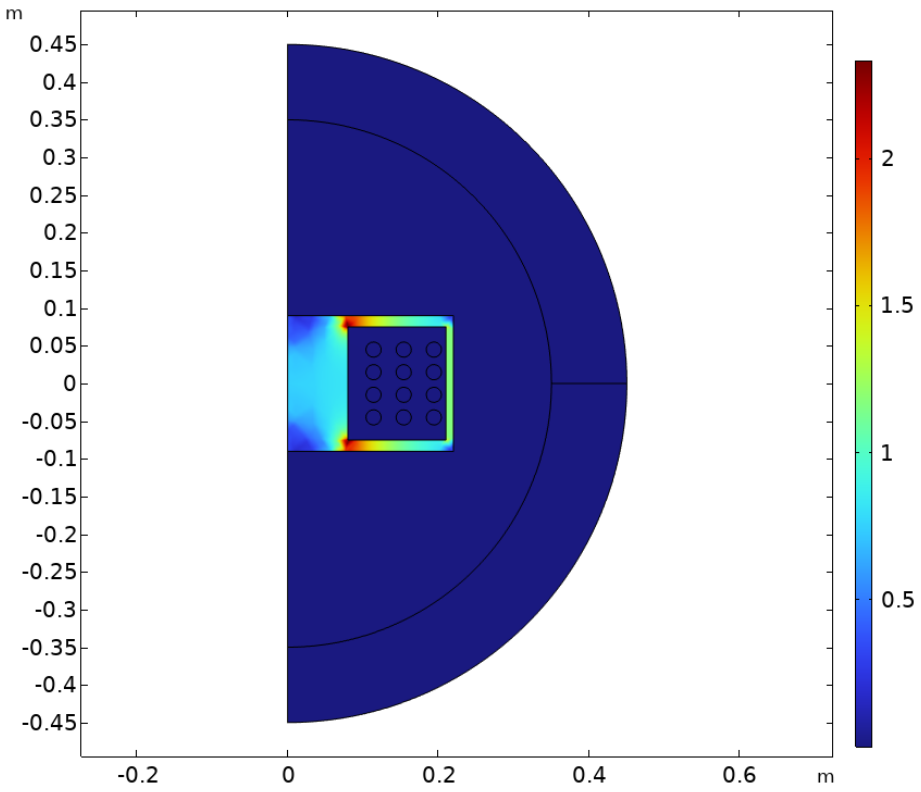


Figure 18: Magnetic flux density norm field of the best inductor configuration identified by our framework, achieving reward 9.609.

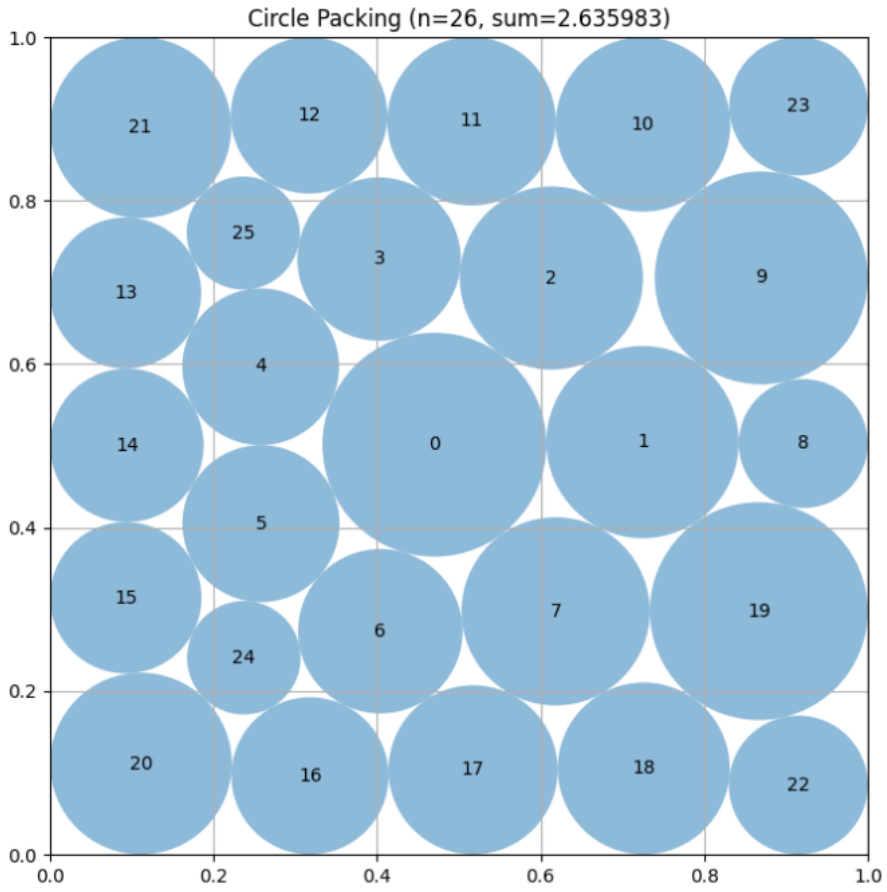


Figure 19: Arrangement of 26 circles within a square obtained by our framework, achieving a record-breaking sum of radii of 2.635983.

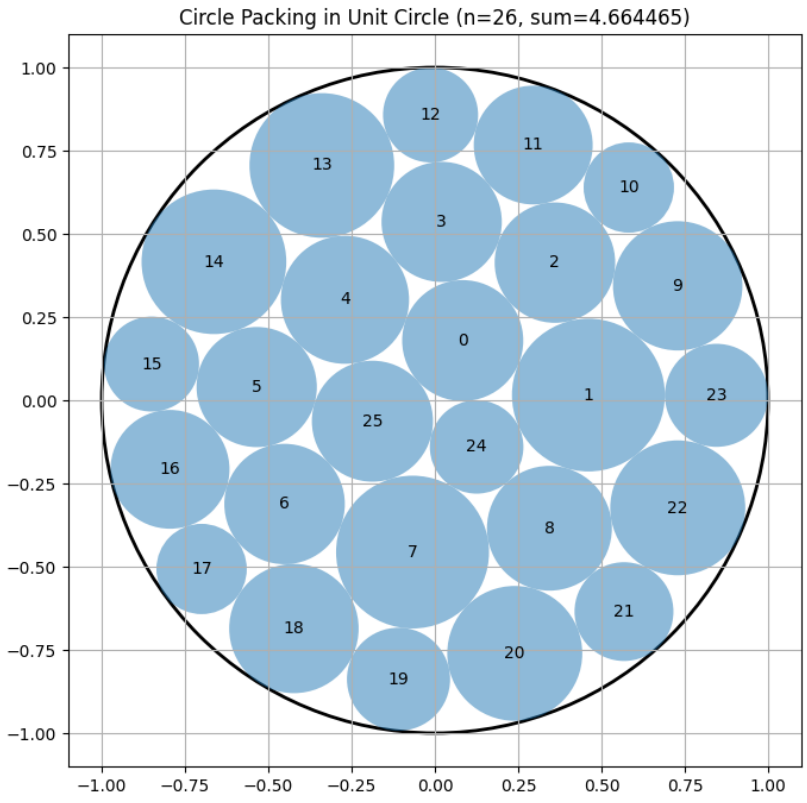


Figure 20: Optimized circle packing of 26 disks within a unit disk by our model, yielding a sum of radii of 4.664465.

```python
def engineer_features(X):
    features = X.copy()
    num_cols = [col for col in X.columns if X[col].dtype != 'object' and col not in ['fnlwgt', 'education-num']]

    # Core interaction features
    features['age_hpw_product'] = features['age'] * features['hours-per-week']
    features['capital_total'] = features['capital-gain'] + features['capital-loss']
    features['log_fnlwgt'] = np.log(features['fnlwgt'] + 1)
    # Enhanced polynomial features
    for col in ['age', 'hours-per-week', 'capital-gain', 'capital-loss']:
        features[f'{col}_sq'] = features[col] ** 2
        features[f'{col}_cb'] = features[col] ** 3
    # Age-based features with log transformation
    features['log_age'] = np.log(features['age'] + 1)
    # Capital features with log transformations
    features['capital_gain'] = np.log(features['capital-gain'] + 1)
    features['capital_loss'] = np.log(features['capital-loss'] + 1)
    # Binned features for age and hours per week
    features['age_group'] = pd.cut(features['age'], bins=5, labels=False)
    features['hour_group'] = pd.cut(features['hours-per-week'], bins=5, labels=False)
    # Economic status features combining multiple variables
    features['economic_status'] = (features['age'] / features['hours-per-week']) * (features['capital_total'])
    # Additional indicators for capital gains and losses
    features['has_capgain'] = (features['capital-gain'] > 0).astype(int)
    features['has_caploss'] = (features['capital-loss'] > 0).astype(int)
    # Professional education indicator
    features['isProfessional'] = ((features['education'] == 'Prof-specialty') | (features['education'] == 'Exec-managerial') | (features['education'] == 'Assoc-acdm')).astype(int)
    # Managerial education indicator
    features['isManagerial'] = ((features['education'] == 'Exec-managerial') | (features['education'] == 'Assoc-voc')).astype(int)
    # Interaction between numerical features
    interaction_cols = ['age', 'hours-per-week', 'capital_gain', 'capital_loss']
    for i in range(len(interaction_cols)):
        for j in range(i+1, len(interaction_cols)):
            col1 = interaction_cols[i]
            col2 = interaction_cols[j]
            features[f'{col1}_x_{col2}'] = features[col1] * features[col2]
    # Ratio and difference features
    features['capital_gain_ratio'] = features['capital_gain'] / features['capital_loss'].replace(0, 1)
    features['capital_diff'] = features['capital_gain'] - features['capital_loss']

    return features
```

Figure 21: Python code of feature engineering for solving classification task on Adult dataset. Together with a LGBMClassifier and parameter search, our model achieved 82.07 marco f1 score.

```python
# Engineer more comprehensive interaction and polynomial features
def create_engineered_features(df):
    # Interaction features
    df['RM_LSTAT'] = df['RM'] * df['LSTAT']
    df['NOX_DIS'] = df['NOX'] * df['DIS']
    df['CRIM_DIS'] = df['CRIM'] * df['DIS']
    df['INDUS_NOX'] = df['INDUS'] * df['NOX']
    df['CHAS_RM'] = df['CHAS'] * df['RM']
    df['AGE_DIS'] = df['AGE'] * df['DIS']
    df['RAD_NOX'] = df['RAD'] * df['NOX']
    df['PTRATIO_RM'] = df['PTRATIO'] * df['RM']
    df['INDUS_CHAS'] = df['INDUS'] * df['CHAS']
    df['RAD_DIS'] = df['RAD'] * df['DIS']
    df['RAD_CHAS'] = df['RAD'] * df['CHAS']  # New interaction
    df['CRIM_CHAS'] = df['CRIM'] * df['CHAS']  # Enhanced interaction
    # Polynomial features
    df['NOX_SQ'] = df['NOX'] ** 2
    df['RM_SQ'] = df['RM'] ** 2
    df['LSTAT_SQ'] = df['LSTAT'] ** 2
    df['NOX_CUBED'] = df['NOX'] ** 3
    df['RM_CUBED'] = df['RM'] ** 3
    df['LSTAT_CUBED'] = df['LSTAT'] ** 3
    df['NOX_FOUR'] = df['NOX'] ** 4  # Higher degree polynomial
    return df

engineered_features_transformer = Pipeline([
    ('engineer', FunctionTransformer(create_engineered_features))
])

# Preprocess numeric features
numeric_transformer = Pipeline([
    ('imputer', KNNImputer(n_neighbors=3, weights='uniform')),
    ('scaler', RobustScaler())
])

# Preprocess categorical features
categorical_transformer = Pipeline([
    ('imputer', SimpleImputer(strategy='mode')),
    ('target_encode', FunctionTransformer(lambda df: df.astype(object).
    where(df.notna(), df.mode().iloc[0]))
    )
])

# Combine transformations
preprocessor = ColumnTransformer(
    transformers=[
        ('eng', engineered_features_transformer, numeric_features),
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)
    ],
    remainder='drop'
)
```

Figure 22: Key pre-processing steps the model implemented on Boston Housing dataset. Together with a XGBRegressor, the model achieved reward of 1.758, which means 1.747 in RMSE.