ASI-Evolve: Towards AI-Driven Discovery Across the AI Research Stack

Weixian Xu[‡,1,3] **Tiantian Mi**[*,1,2,3] **Yixiu Liu**[*,1,2,3]

Yang Nan[*,3] **Zhimeng Zhou**[*,2] **Lyumanshan Ye**[1,2,3]

Lin Zhang[1,2,3] **Yu Qiao**[2] **Pengfei Liu**[†,1,2,3]

[1]Shanghai Jiao Tong University [2]Shanghai Innovation Institute [3]GAIR

Abstract

Can AI accelerate the development of AI itself? While recent agentic systems have shown strong performance on well-scoped tasks, it remains unclear whether they can tackle long-horizon research loops that drive real AI progress. In this paper, we present ASI-EVOLVE, an agentic framework for AI-for-AI research based on a learn–design–experiment–analyze cycle. ASI-EVOLVE delivers strong results across core areas of AI development: **architectures, data, and algorithms**. In neural architecture design, it discovers **105 SOTA linear attention architectures**, with the best model surpassing DeltaNet by nearly 3× the gain of recent human-designed improvements. In pretraining data curation, its evolved data preparation pipeline yields training data that improves downstream model performance by +3.96 points on average, with gains **exceeding 18 points on MMLU**. In reinforcement learning algorithm design, it discovers algorithms that outperform GRPO by up to **+12.5 points on AMC32, +11.67 points on AIME24, and +5.04 points on OlympiadBench**. Further experiments in **mathematics** and **biomedicine** demonstrate the real-world effectiveness of this paradigm, showing that AI can iteratively improve both itself and its broader applications. ASI-EVOLVE is fully open-sourced at https://github.com/GAIR-NLP/ASI-Evolve.

1 Introduction

Artificial intelligence (AI) advances through **data, architectures and algorithms**. Progress in each depends on cycles of hypothesizing, implementation, experimentation, and analysis (Ghareeb et al., 2025), yet these cycles remain constrained by human bottlenecks (Zhang et al., 2025): limited hypotheses (Liu et al., 2025a), manual experiments (Feng et al., 2025), and insights that are hard to preserve or transfer (Kosmyna et al., 2025). These bottlenecks limit the pace and scope of AI research itself. Meanwhile, AI capabilities are advancing rapidly, making it natural to ask whether AI can help accelerate its own scientific development (Didolkar et al., 2024; Wang et al., 2023).

AI systems for scientific discovery have progressed from specialized tools such as AlphaFold (Jumper et al., 2021), GraphCast (Lam et al., 2023), and GNoME (Merchant et al., 2023) to LLM-based and agentic workflows (Wei et al., 2025). Still, current systems remain narrow: SciMaster (Chai et al., 2025) answers known questions; ML-Master (Liu et al., 2025b) and MLEvolve (Du et al., 2025) solve bounded optimization; AI Scientist (Lu et al., 2024) automates paper production; and AlphaEvolve (Novikov et al., 2025) improves candidates through coding agents. Real AI research is harder: architectures, data pipelines, and training

‡Leading author. ∗ Core contributors. † Corresponding author.

algorithms require prior knowledge, large-codebase changes, costly experiments, multidimensional interpretation, and long-horizon coordination. No prior framework tackles these challenges together.

To address this gap, we present ASI-EVOLVE, an agentic framework for **AI-for-AI** research. ASI-EVOLVE follows the scientific cycle of grounding in literature, formulating hypotheses, running experiments, and distilling insights (Ghareeb et al., 2025) through a *learn–design–experiment–analyze* loop. A structured **cognition base** grounds each round in human research, while a dedicated **analyzer** converts complex outcomes into reusable insights for long-horizon research with expensive, indirect, noisy feedback.

Using ASI-EVOLVE, we show that AI can accelerate multiple layers of its own development stack. **To our knowledge, this is the first unified demonstration of AI-driven discovery across data, architectures, and algorithms.** In **model architecture** design, ASI-EVOLVE generated 1,350 candidates over 1,773 rounds and found 105 architectures surpassing DeltaNet (Yang et al., 2024b); the best improved by +0.97 points, nearly triple recent manual SOTA gains (Dao & Gu, 2024). In **data curation**, evolved strategies improved average scores by 3.96 points and MMLU by over 18 points (Hendrycks et al., 2021). In **training algorithm** design, discovered algorithms outperform GRPO (Shao et al., 2024) by up to +12.5 on AMC32, +11.67 on AIME24, and +5.04 on OlympiadBench. Together, these results demonstrate AI self-improvement across model design, data preparation, and learning algorithms.

We further analyze ASI-EVOLVE and test the applicability of its discovered designs. On circle packing, it reaches SOTA-level results in 17 rounds, faster than OpenEvolve and GEPA. An architecture created by ASI-EVOLVE also improves cold-start AUROC in drug-target interaction prediction by 6.94 points, suggesting that AI-evolved designs can be useful beyond core AI tasks.

2 Background and Related Works

ASI-EVOLVE targets long-horizon scientific tasks not jointly addressed by existing systems. We introduce **Scientific Task Length (L_{task})** along three dimensions: Execution Cost (C_{exec}), the computational and engineering burden per trial; Search Space Complexity (S_{space}), the openness and scale of the solution space; and Feedback Complexity (D_{feedback}), the difficulty of extracting actionable insights from outcomes.

Low-to-moderate complexity work. Most existing systems operate where at least one dimension of L_{task} remains low. Scientific question answering (Rein et al., 2024; Phan et al., 2025; Chai et al., 2025) and idea generation (Wang et al., 2024) incur little C_{exec} and require no iterative experimental loop. Structured execution systems such as SWE-bench (Jimenez et al., 2024), AIDE (Jiang et al., 2025), and AI Scientist (Lu et al., 2024) run experiments but target predefined tasks with bounded search and simple feedback. Lightweight evolutionary frameworks, including AlphaEvolve (Novikov et al., 2025), FunSearch (Romera-Paredes et al., 2023), and follow-up work (Sharma, 2025; Agrawal et al., 2025; Liu et al., 2026), raise S_{space} and C_{exec} through open-ended search, yet keep D_{feedback} low because each trial modifies short code and returns direct feedback.

Large-scale scientific exploration. Neural architecture design, pretraining data curation, and training algorithm design occupy the upper end of L_{task}, pushing all three dimensions beyond existing systems. Validating one candidate can require tens to hundreds of GPU hours and deep codebase changes ($C_{\text{exec}} \uparrow$); the design space is open-ended ($S_{\text{space}} \uparrow$); and feedback spans benchmarks, loss dynamics, and efficiency metrics ($D_{\text{feedback}} \uparrow$). In ASI-EVOLVE (Section 3), a cognition base steers exploration with domain knowledge while a dedicated analyzer distills experimental signals into actionable insights, so the system evolves both candidate solutions and its capacity to decide where to search next.

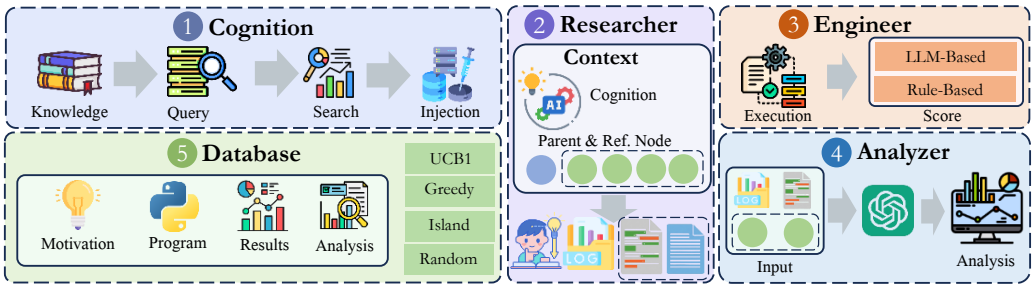


Figure 1: ASI-EVOLVE pipeline: sample context nodes from database, retrieve cognition items via embedding search, generate a candidate program, run evaluation under timeouts, and store the analysis report back into the database.

3 ASI-Evolve

ASI-EVOLVE is an end-to-end experimental evolution pipeline. Each iteration has four stages (Figure 1): (i) **learn** knowledge and historical experience from cognition storage and database, (ii) **design** the next candidate program, (iii) run an **experiment** to obtain evaluation signals, and (iv) **analyze** outcomes into reusable lessons. The decomposition reflects a basic property of research: useful hypotheses follow from connecting prior knowledge, failed trials, and measured outcomes, not from the latest score alone.

We view each evolution round t as searching a program space $\mathcal{P}$ of code artifacts. The system maintains: (1) a database $\mathcal{D}$ of past nodes (motivation, code, results, analysis, score, and metadata), and (2) a cognition store C of task-relevant textual items indexed by embeddings. Given sampled context nodes $S_t \sim \text{Sample}(\mathcal{D})$ and retrieved cognition items $R_t = \text{Retrieve}(C; S_t)$, the system generates

$$p_t \sim P(p \mid S_t, R_t),$$

evaluates it with an experiment-specific procedure that produces structured metrics and a primary score f_t, and appends $z_t = (m_t, p_t, e_t, a_t, f_t)$ to $\mathcal{D}$, where m_t is the motivation, e_t the experimental record, and a_t the analyzer report, so each run becomes evidence for future proposals rather than only a selection score.

3.1 Learn

For long-horizon research tasks, starting from scratch leaves a broad but costly hypothesis space, so Learn combines a **Cognition Base** of domain priors—heuristics, pitfalls, and design principles from literature and prior runs—with a **Database** of evolution nodes. Each node stores motivation, program, results, analysis, score, and metadata. In each round, the system samples n history nodes, retrieves related cognition from their analyses or motivations, and injects them as priors. Cognition supplies stable constraints before enough experiments exist, while the database supplies task-specific evidence.

3.2 Design

The **Researcher** generates the next candidate from the task description, sampled history, and retrieved cognition. Conditioned on this context, an LLM outputs a complete program and concise motivation, both stored as a new node. The motivation exposes the intended mechanism behind a change, helping the Analyzer distinguish unlucky evaluations from flawed assumptions. For large codebases, ASI-EVOLVE uses incremental editing: the model proposes local changes to a parent program, preserving working scaffolds while reducing regeneration.

3.3 Experiment

The **Engineer** runs the candidate program and returns the quantitative signal for selection. A user-specified evaluator executes end-to-end and reports structured metrics, including primary fitness, grounding evolution in executable evidence. To control cost, wall-clock limits and quick tests reject flawed candidates before expensive runs. When rule-based metrics miss quality, ASI-EVOLVE adds an LLM judge (Zheng et al., 2023) and combines its score with the primary metric.

3.4 Analyze

Selection uses a scalar task score, but each run also yields auxiliary signals—metrics, feature importances, logs, and traces—that are useful for diagnosis yet too verbose for later rounds. The **Analyzer** receives the current program and full experimental output, including raw logs and metrics, and distills them into a compact, decision-oriented report. The report is stored for retrieval in later rounds, keeping context manageable without losing analytical depth.

4 Main Tasks

We apply ASI-EVOLVE to three central components of AI development—model architecture design, training data preparation, and training algorithm design. Each has limited priors, long iteration cycles, implementation complexity, and indirect, multi-dimensional feedback, forming a demanding testbed for autonomous AI research at scale.

4.1 Scenario 1: Model Architecture Design

Task Formulation We study architecture design for efficient sequence models, focusing on linear attention. Starting from DeltaNet (Yang et al., 2024b), the goal is to discover sub-quadratic attention layers that preserve capacity, support chunk-wise parallel training, and integrate into an existing codebase. Routing, memory updates, and path allocation must be searched jointly under strict efficiency and causal constraints, so many plausible ideas fail after implementation and training.

Methodology The cognition store starts with about 150 entries from 100 papers on linear attention, state space models, and efficient transformers. Because evaluations take hours and invalid designs are common, we add a *static check agent* for complexity, chunking, and causal validity, a *debug agent* for runtime failures, and a *novelty check* for near duplicates. Evaluation has three stages: *exploration* on $\sim$20M models trained for 2000 steps on 1B tokens and 10 benchmarks, *verification* on $\sim$340M models trained on 1B tokens, and *large-scale validation* on $\sim$1.3B models trained on 100B tokens and tested on 16 benchmarks, including 6 held-out OOD sets. Fitness combines normalized benchmark/loss signals with LLM-as-a-Judge scores of novelty, efficiency, and code complexity.

Results Across 1,773 exploration rounds, **105** architectures beat DeltaNet in verification, and we select 5 for large-scale validation. Table 1 compares them with DeltaNet, Gated DeltaNet (Yang et al., 2024a), and Mamba2 (Dao & Gu, 2024). On development benchmarks, the best evolved models reach 57.28% average accuracy versus DeltaNet's 55.76%; on generalization benchmarks, they reach 45.40% versus 44.74%. Our best model delivers nearly **3**$\times$ the gain over DeltaNet achieved by the strongest human-designed comparator (Mamba2's +0.34 points), putting evolved designs above the strongest human baseline in this saturated regime.

Analysis The top architectures share one principle: adaptive multi-scale routing beats fixed allocation. Representative designs differ—hierarchical routing in **PathGateFusionNet**, temperature-controlled content routing in **ContentSharpRouter**, and token-level balanced sparse gating in **AdaMultiPathGateNet**—but all allocate compute dynamically across local,

Table 1: Development and generalization benchmarks for 1.3B validation. **Bold** marks the best result and underline the second best. Abbreviations: PG = PathGateFusionNet, C = ContentSharpRouter, FG = FusionGatedFIRNet, H = HierGateNet, AM = AdaMultiPathGateNet.

	Human Discovered			AI Discovered				
Benchmarks	**DeltaNet**	**Gated-DeltaNet**	**Mamba2**	**PG**	**C**	**FG**	**H**	**AM**
Development								
Wiki ppl↓	17.00	16.84	16.66	16.18	16.05	**15.77**	16.65	16.26
LMB ppl↓	13.63	13.31	13.33	12.62	13.45	**12.34**	13.06	13.75
LMB	45.47	46.26	46.24	**47.60**	46.13	47.53	46.56	45.04
PIQA	73.12	74.10	73.78	72.91	74.37	72.91	**74.37**	74.10
Hella	56.29	57.55	**58.58**	56.99	57.00	58.47	56.85	57.17
Wino	55.88	58.01	58.48	57.22	57.85	**60.14**	57.38	57.62
ARC-e	73.40	72.14	72.98	73.06	72.05	74.28	73.11	**74.28**
ARC-c	**40.61**	36.95	39.33	40.36	39.76	40.02	39.33	39.33
SIQA	40.74	41.71	41.81	42.37	41.81	**42.78**	42.07	42.07
BoolQ	60.58	53.98	60.52	62.45	62.51	62.11	**63.03**	56.27
Generalization								
RACE	34.45	33.78	32.15	35.22	34.55	**35.60**	35.22	35.02
BBQ	29.53	29.75	29.43	29.95	30.55	**31.46**	30.88	30.27
MetaBench	26.97	28.67	27.70	25.64	**29.55**	26.79	29.38	28.98
QA4MRE	**40.00**	35.00	39.17	39.17	39.17	38.33	38.33	38.33
SCIQ	89.80	90.30	90.30	89.60	89.50	89.20	**90.40**	89.20
SWAG	47.69	48.17	**48.88**	48.22	47.80	48.57	48.18	47.80
Averages								
Dev. Avg	55.76	55.09	56.47	56.62	56.44	**57.28**	56.59	55.74
Gen. Avg	44.74	44.28	44.61	44.63	45.19	44.99	**45.40**	44.93
Overall Avg	51.04	50.46	51.38	51.48	51.61	**52.01**	51.79	51.11

global, and delta-style memory paths. Search moves from exploration to refinement: 78% of high-performing architectures found in the second half build on pre-round-900 designs. Provenance also shifts: across all proposals, 51.7% derive from cognition, 38.2% from accumulated experience, and 10.1% are novel; among SOTA architectures, experience rises to 44.8% and novelty falls to 6.6%, suggesting cognition accelerates early search while accumulated experience drives later gains.

4.2 Scenario 2: Pretraining Data Curation

Task Formulation This scenario studies category-specific pretraining data curation. For each corpus category, the system inspects samples, identifies quality issues, and writes executable cleaning rules that improve quality while preserving useful content. The challenge is heterogeneity: deletion, normalization, and preservation must vary by domain and quality level, so rules that help one category can harm another.

Methodology The cognition store is built from sampled documents in each category, recording HTML artifacts, incomplete fragments, formatting inconsistencies, and domain-specific noise. In each round, the Researcher proposes a category-specific strategy, the Engineer applies it to 500 sampled documents, and the Analyzer scores 50 random (original, cleaned) pairs on a 1–10 scale with coverage and executability feedback. The database stores strategies and analyses, and newly identified issues return to cognition.

Table 2: Benchmark comparison for 3B models trained on 500B tokens with different pretraining corpora and curation strategies. Bold and underlined entries denote the best and second-best results.

	Human Discovered				AI Discovered	
Benchmark	**Fineweb-Edu**	**Ultra-Fineweb**	**DCLM**	**Nemotron-CC**	**Nemotron-CC$_{ASI}$**	**Nemotron-CC$_{ASI+}$**
BBH	3.01	7.42	24.16	**26.82**	26.69	26.16
ARC-E	73.39	73.96	75.13	74.94	77.55	**78.59**
ARC-C	43.45	43.77	45.02	43.52	48.41	**49.32**
MMLU	28.38	25.53	28.54	27.49	32.55	**46.13**
AGIEval	16.96	17.72	17.90	18.15	**18.30**	18.21
HellaSwag	64.33	65.32	**70.39**	65.32	64.36	62.21
TriviaQA	0.67	0.42	**42.85**	25.33	26.96	26.65
RACE	35.43	34.28	**36.08**	35.04	35.63	34.28
DROP	6.78	7.78	**24.31**	19.57	19.48	18.49
WinoGrande	61.02	60.93	**64.99**	57.96	59.89	58.09
PIQA	75.80	75.63	**77.93**	76.79	76.80	76.15
CSQA	19.54	19.90	20.16	20.31	20.61	**39.12**
SIQA	45.02	43.43	**47.51**	44.36	43.58	43.57
OpenBookQA	39.92	39.84	**43.36**	39.80	42.20	41.44
GPQA	24.51	23.04	25.67	24.37	23.93	**27.10**
MedQA	26.36	24.84	24.88	26.77	26.11	**40.25**
MedMCQA	25.80	24.92	28.15	28.86	30.28	**40.97**
PubMedQA	67.04	64.44	66.56	67.68	**68.32**	67.68
Avg	36.52	36.29	42.42	40.17	41.20	**44.13**

Results The system finds effective strategies for all selected Nemotron-CC (Su et al., 2024) categories, covering 672B academic tokens across mathematics, computer science, medicine, and other STEM fields at two quality levels. Table 2 compares models trained on different corpora, where Nemotron-CC$_{ASI}$ and Nemotron-CC$_{ASI+}$ apply the suboptimal and the optimized evolved strategies respectively. Applying the optimized strategies yields Nemotron-CC$_{ASI+}$ (504B tokens), which reaches 44.13 average score when a 3B model is trained from scratch on 500B tokens. This exceeds raw Nemotron-CC by 3.96 points and outperforms DCLM (Li et al., 2024), FineWeb-Edu (Lozhkov et al., 2024), and Ultra-FineWeb (Wang et al., 2025) under the same budget. Gains are largest on knowledge-intensive tasks: **MMLU +18.64**, **CSQA +18.81**, and **MedQA +13.48**.

Analysis Across categories, strategies converge on targeted noise removal, format normalization, and domain-aware preservation without a fixed rule set. Strong strategies use concrete thresholds, category-specific deletion rules, and preservation constraints to avoid over-cleaning. The 2.93-point gap between optimized and suboptimal strategies shows the value of iterative refinement: gains come from accumulated diagnostics, not one-shot heuristics.

4.3 Scenario 3: Reinforcement Learning Algorithm Design

Task Formulation We study RL algorithm design for LLM training with Group Relative Policy Optimization (GRPO) (Shao et al., 2024) as the baseline. The search target is the advantage assignment rule across sequence groups and the induced gradient update. Small mathematical changes can look promising in short noisy runs yet fail at scale, so the system must find stable modifications rather than lucky ones.

Methodology We seed cognition with 10 post-GRPO papers on variance reduction and KL-penalty design. Evaluation has two stages: exploration on a 4B model trained for 150 steps on 6 math benchmarks, then verification on a 14B model trained for 300 steps across abstract reasoning, STEM, finance, and coding. Fitness combines benchmark accuracy with LLM-as-a-Judge assessments of performance and algorithmic coherence.

Results Across 300 rounds, the system evaluates diverse policy-gradient variants; 10 outperform GRPO in exploration and 3 retain clear gains at 14B verification. Table 3 reports representative results at 250 RL steps. On math benchmarks, the best variant improves over GRPO by +12.5 on AMC32 (67.5 $\rightarrow$ 80.0), +11.67 on AIME24 (20.00 $\rightarrow$ 31.67), and +5.04

Table 3: Mathematical reasoning results for GRPO and five evolved variants on Qwen-3-14B-base, evaluated at 250 RL steps. **Bold** marks the best result.

Method	Math500	AMC32	AIME25	AIME24	OlympiadBench
$\text{GRPO}_{\text{Human}}$	82.0	67.5	20.00	20.00	45.92
$\text{Algorithm1}_{\text{ASI-Evolve}}$	**85.6**	**80.0**	29.58	**31.67**	**50.96**
$\text{Algorithm2}_{\text{ASI-Evolve}}$	84.6	77.5	23.75	23.33	48.74
$\text{Algorithm3}_{\text{ASI-Evolve}}$	84.8	77.5	**30.00**	30.00	49.18
$\text{Algorithm4}_{\text{ASI-Evolve}}$	82.4	75.0	20.00	20.00	46.81
$\text{Algorithm5}_{\text{ASI-Evolve}}$	82.0	72.5	23.33	20.00	45.62

on OlympiadBench (45.92 $\rightarrow$ 50.96). Gains concentrate on hard reasoning, indicating that search improves the update rule itself rather than sampling variance.

Analysis The strongest algorithms make principled changes to advantage estimation and update control. One family uses pairwise comparative advantages, asymmetric clipping, and high-impact gradient dropout to avoid overfitting high-reward tokens. Another uses percentile normalization plus a global update budget that shrinks trusted update radii for unstable tokens. Both remain interpretable GRPO extensions rather than opaque benchmark artifacts.

5 Empirical Analysis

Having shown ASI-EVOLVE's effectiveness across three real-world domains in Section 4, we analyze the framework more systematically. We use circle packing for controlled comparisons, base-model and sampling studies, and component ablations, then test whether evolved solutions transfer beyond AI/ML through a biomedical task.

5.1 Benchmarking ASI-Evolve on Circle Packing

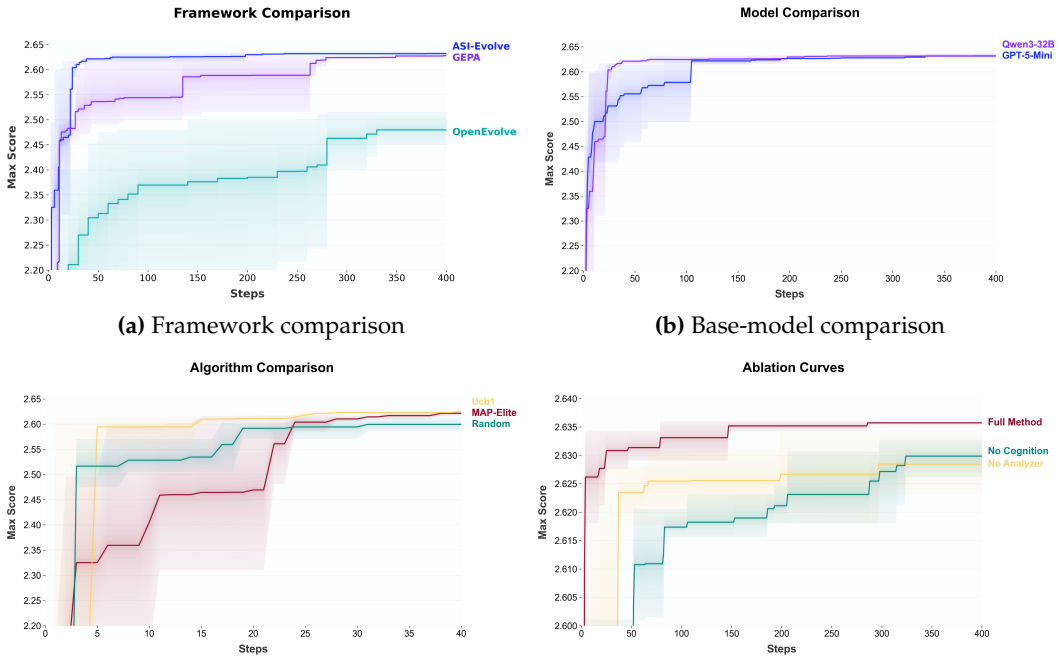
Task and results. We use the AlphaEvolve circle-packing benchmark: place 26 circles in a 1×1 square to maximize the sum of radii. As Table 4 shows, ASI-EVOLVE reaches 2.63597 in 17 steps and a best score of 2.635983, matching top reported results with the fastest reported convergence.

Table 4: Circle packing comparison; higher is better, "—" is unreported, and **bold** marks the fewest rounds to the best reported score.

Framework	Model	Rounds	Best Score
AlphaEvolve (Novikov et al., 2025)	Gemini 2.0 Flash + Claude 3.7	—	2.6359
OpenEvolve (Sharma, 2025)	Gemini 2.0 Flash + Claude 3.7	460	2.6343
LoongFlow (Baidu Baige Team, 2025)	DeepSeek-R1-250528	—	2.6360
SkyDiscover (Liu et al., 2026)	GPT-5	89	2.6360
ASI-Evolve **(Ours)**	GPT-5-mini	**17**	2.6360

5.2 Comparison Experiments

Framework Comparison. Using Qwen3-32B (Qwen Team, 2025) as the base model, we compare ASI-EVOLVE with OpenEvolve (Sharma, 2025) and GEPA (Agrawal et al., 2025) under aligned prompts; full configurations are in Appendix B. Figure 2(a) shows that OpenEvolve improves with high variance and a low plateau, while GEPA is stable but converges around 2.630. ASI-EVOLVE starts stronger, improves steadily, and reliably reaches SOTA-level performance.



(a) Framework comparison

(b) Base-model comparison

(c) Sampling-algorithm comparison

(d) Component ablation

Figure 2: **Comparison and ablation experiments** on circle packing. All panels plot best score against evolution step; shading shows repeated-run variability.

Model Comparison. Running ASI-EVOLVE with GPT-5-mini and Qwen3-32B, Figure 2(b) shows that both converge to a similar range and sustain mid-to-late-stage improvement: early cadence differs, but final levels are comparable, so evolution is not tied to one model family.

Algorithm Comparison. We compare three parent-sampling rules: **MAP-Elites** (Mouret & Clune, 2015) preserves niches, **UCB1** (Auer et al., 2002) adds value-guided exploration, and **Random** samples uniformly. Figure 2(c) shows that Random starts strong but plateaus, while UCB1 reaches high-score regions faster and with lower variance than MAP-Elites. The same ordering holds on GPT-5-mini, where UCB1 reaches 2.63597 in 17 steps versus 79 for MAP-Elites, so once Cognition and Analyzer supply direction, value-guided parent selection beats extra niche diversity.

5.3 Ablation Study: Validating Component Effectiveness

Setting. We compare the **Full Method** against **No Analyzer**, which stores raw scores and logs directly in the Database, and **No Cognition**, which relies on self-driven trial-and-error. Each configuration is run three times in Figure 2(d).

Ablation results. No Analyzer starts high due to Cognition's cold-start prior, then plateaus with less reproducible gains, showing that Analyzer supports reliable refinement. No Cognition starts slower and spends longer at low scores before self-guided improvement, indicating that Cognition supplies priors that reduce unproductive early search without being required for the loop to run.

5.4 Scientific Applicability: Drug–Target Interaction Discovery

To test transfer beyond AI-for-AI, we evolve DrugBAN (Bai et al., 2022) for drug–target interaction (DTI) prediction over 100+ candidates in 100+ rounds, seeded with cognition from roughly 80 papers. Appendix A gives the full protocol and architecture analysis.

Table 5: DTI prediction performance. We report AUROC (%) and F1 (%); bold indicates best performance.

Discovered by	Model	BindingDB-Dev		BindingDB-Random		Human		BioSNAP	
		AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1
Human	TransformerCPI	-	-	93.96	87.02	96.03	91.35	86.35	78.53
	PSICHIC	-	-	91.67	83.30	98.55	94.85	91.64	84.87
	ConPlex	-	-	93.59	83.39	97.04	90.38	88.40	72.86
	ColdStartCPI	-	-	89.27	85.27	98.29	93.57	**93.39**	**86.84**
	DrugBAN	94.15	86.89	94.89	87.96	98.61	**95.40**	89.43	82.69
AI	**ASI-Evolve**	**96.06**	**89.84**	**95.94**	**89.35**	**98.89**	95.32	89.68	82.92

Table 6: Cold-start DTI performance on unseen drugs, proteins, or both.

Discovered by	Model	Unseen Drug		Unseen Protein		Unseen Drug and Protein	
		AUROC	F1	AUROC	F1	AUROC	F1
Human	DrugBAN	79.15	77.39	82.26	75.76	76.47	71.53
AI	**ASI-Evolve**	**86.09**	**82.35**	**85.82**	**78.44**	**80.83**	**74.51**

In-distribution results. Table 5 compares the evolved architecture with its DrugBAN seed and four expert-designed DTI models. It improves over DrugBAN on BindingDB-Dev (+1.91 AUROC, +2.95 F1), BindingDB-Random (+1.05 AUROC, +1.39 F1), and Human (98.89 AUROC), and is competitive on BioSNAP, where ColdStartCPI remains the strongest baseline.

Cold-start results. The larger gains appear out of distribution (Table 6): **+6.94** AUROC on unseen drugs, **+3.56** on unseen proteins, and **+4.36** when both are unseen. The evolved design combines Sinkhorn attention (Tay et al., 2020) for balanced drug–protein matching, domain-specific marginalization, and top-k sparse gating. Every cold-start gain exceeds both in-distribution AUROC gains, suggesting the design captures more transferable molecular interaction patterns rather than dataset-specific shortcuts.

6 Conclusion

We presented ASI-EVOLVE, an agentic evolution framework for AI-for-AI scientific research, and showed that its cognition base and analyzer enable fast cold starts, steady improvement, and SOTA-level results. Across architecture, data, and training-algorithm design, the closed *learn–design–experiment–analyze* loop advances core AI research despite implementation complexity, iteration cost, and indirect feedback, and our circle-packing and DTI studies indicate that it carries over to mathematical and biomedical tasks. As AI handles more of the discovery loop, researchers can concentrate on problem design, objectives, evaluations, and research directions. See Appendix C for limitations and broader impact.

References

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. Gepa: Reflective prompt evolution can outperform reinforcement learning, 2025. URL https://arxiv.org/abs/2507.19457.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2–3):235–256, 2002. doi: 10.1023/A:1013689704352. URL https://doi.org/10.1023/A:1013689704352.

Peizhen Bai, Filip Miljkovic, Bino John, and Haiping Lu. Interpretable bilinear attention network with domain adaptation improves drug-target prediction. *Nature Machine Intelligence*, 5:126–136, 2022. doi: 10.1038/s42256-022-00605-1. URL https://doi.org/10.1038/s42256-022-00605-1.

Baidu Baige Team. Loongflow: Directed evolutionary search via a cognitive plan-execute-summarize paradigm, 2025. URL https://arxiv.org/abs/2512.24077.

Jingyi Chai, Shuo Tang, Rui Ye, Yuwen Du, Xinyu Zhu, Mengcheng Zhou, Yanfeng Wang, Weinan E, Yuzhi Zhang, Linfeng Zhang, and Siheng Chen. Scimaster: Towards general-purpose scientific ai agents, part i. x-master as foundation: Can we lead on humanity's last exam?, 2025. URL https://arxiv.org/abs/2507.05241.

Lifan Chen, Xiaoqin Tan, Dingyan Wang, Feisheng Zhong, Xiaohong Liu, Tianbiao Yang, Xiaomin Luo, Kaixian Chen, Hualiang Jiang, and Mingyue Zheng. Transformercpi: improving compound-protein interaction prediction by sequence-based deep learning with self-attention mechanism and label reversal experiments. *Bioinformatics*, 36(16): 4406–4414, 2020. doi: 10.1093/bioinformatics/btaa524. URL https://doi.org/10.1093/bioinformatics/btaa524.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality, 2024. URL https://arxiv.org/abs/2405.21060.

Aniket Didolkar, Anirudh Goyal, Nan Rosemary Ke, Siyuan Guo, Michal Valko, Timothy Lillicrap, Danilo Rezende, Yoshua Bengio, Michael Mozer, and Sanjeev Arora. Metacognitive capabilities of llms: An exploration in mathematical problem solving, 2024. URL https://arxiv.org/abs/2405.12205.

Shangheng Du, Xiangchao Yan, Dengyang Jiang, Jiakang Yuan, Yusong Hu, Xin Li, Liang He, Bo Zhang, and Lei Bai. Automlgen: Navigating fine-grained optimization for coding agents. *arXiv preprint arXiv:2510.08511*, 2025.

Haodong Feng, Lugang Ye, and Dixia Fan. Towards an ai fluid scientist: Llm-powered scientific discovery in experimental fluid mechanics, 2025. URL https://arxiv.org/abs/2512.04716.

Ali Essam Ghareeb, Benjamin Chang, Ludovico Mitchener, Angela Yiu, Caralyn J. Szostkiewicz, Jon M. Laurent, Muhammed T. Razzak, Andrew D. White, Michaela M. Hinks, and Samuel G. Rodriques. Robin: A multi-agent system for automating scientific discovery, 2025. URL https://arxiv.org/abs/2505.13400.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. Aide: Ai-driven exploration in the space of code, 2025. URL https://arxiv.org/abs/2502.13138.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues?, 2024. URL https://arxiv.org/abs/2310.06770.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. doi: 10.1038/s41586-021-03819-2.

Huan Yee Koh, Anh T. N. Nguyen, Shirui Pan, Lauren T. May, and Geoffrey I. Webb. Physicochemical graph neural network for learning protein-ligand interaction fingerprints from sequence data. *Nature Machine Intelligence*, 2024. doi: 10.1038/s42256-024-00847-1. URL https://doi.org/10.1038/s42256-024-00847-1.

Nataliya Kosmyna, Eugene Hauptmann, Ye Tong Yuan, Jessica Situ, Xian-Hao Liao, Ashly Vivian Beresnitzky, Iris Braunstein, and Pattie Maes. Your brain on chatgpt: Accumulation of cognitive debt when using an ai assistant for essay writing task, 2025. URL https://arxiv.org/abs/2506.08872.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, Alexander Merose, Stephan Hoyer, George Holland, Oriol Vinyals, Jacklynn Stott, Alexander Pritzel, Shakir Mohamed, and Peter Battaglia. Graphcast: Learning skillful medium-range global weather forecasting, 2023. URL https://arxiv.org/abs/2212.12794.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alexandros G. Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. Datacomp-lm: In search of the next generation of training sets for language models. *arXiv preprint arXiv:2406.11794*, 2024.

Shu Liu, Mert Cemri, Shubham Agarwal, Alexander Krentsel, Ashwin Naren, Qiuyang Mang, Zhifei Li, Akshat Gupta, Monishwaran Maheswaran, Audrey Cheng, Melissa Pan, Ethan Boneh, Kannan Ramchandran, Koushik Sen, Alexandros G. Dimakis, Matei Zaharia, and Ion Stoica. Skydiscover: A flexible framework for ai-driven scientific and algorithmic discovery, 2026. URL https://skydiscover-ai.github.io/blog.html.

Yixiu Liu, Yang Nan, Weixian Xu, Xiangkun Hu, Lyumanshan Ye, Zhen Qin, and Pengfei Liu. Alphago moment for model architecture discovery, 2025a. URL https://arxiv.org/abs/2507.18074.

Zexi Liu, Yuzhu Cai, Xinyu Zhu, Yujie Zheng, Runkun Chen, Ying Wen, Yanfeng Wang, Weinan E, and Siheng Chen. Ml-master: Towards ai-for-ai via integration of exploration and reasoning, 2025b. URL https://arxiv.org/abs/2506.16499.

Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. Fineweb-edu: the finest collection of educational content, 2024. URL https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024.

Amil Merchant, Simon Batzner, Samuel S. Schoenholz, Muratahan Aykol, Anubhav Jain, and Ekin Dogus Cubuk. Scaling deep learning for materials discovery. *Nature*, 624 (7990):80–85, 2023. doi: 10.1038/s41586-023-06735-9. URL https://doi.org/10.1038/s41586-023-06735-9.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites, 2015. URL https://arxiv.org/abs/1504.04909.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. Alphaevolve: A coding agent for scientific and algorithmic discovery, 2025. URL https://arxiv.org/abs/2506.13131.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 2023. doi: 10.1038/s41586-023-06924-6.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

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

Rohit Singh, Sam Sledzieski, Lenore Cowen, and Bonnie Berger. Contrastive learning in protein language space predicts interactions between drugs and protein targets. *Proceedings of the National Academy of Sciences*, 120(24):e2220778120, 2023. doi: 10.1073/pnas.2220778120. URL https://doi.org/10.1073/pnas.2220778120.

Dan Su, Kezhi Kong, Ying Lin, Joseph Jennings, Brandon Norick, Markus Kliegl, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-cc: Transforming common crawl into a refined long-horizon pretraining dataset. *arXiv preprint arXiv:2412.02595*, 2024.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention, 2020. URL https://arxiv.org/abs/2002.11296.

Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Chao Liu, Peter Van Katwyk, Andreea Deac, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 2023. doi: 10.1038/s41586-023-06221-2. URL https://doi.org/10.1038/s41586-023-06221-2.

Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. Scimon: Scientific inspiration machines optimized for novelty. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 279–299. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.18. URL http://dx.doi.org/10.18653/v1/2024.acl-long.18.

Yudong Wang, Zixuan Fu, Jie Cai, Peijun Tang, Hongya Lyu, Yewei Fang, Zhi Zheng, Jie Zhou, Guoyang Zeng, Chaojun Xiao, Xu Han, and Zhiyuan Liu. Ultra-FineWeb: Efficient data filtering and verification for high-quality llm training data, 2025.

Jiaqi Wei, Yuejin Yang, Xiang Zhang, Yuhan Chen, Xiang Zhuang, Zhangyang Gao, Dongzhan Zhou, Guangshuai Wang, Zhiqiang Gao, Juntai Cao, Zijie Qiu, Ming Hu, Chenglong Ma, Shixiang Tang, Junjun He, Chunfeng Song, Xuming He, Qiang Zhang, Chenyu You, Shuangjia Zheng, Ning Ding, Wanli Ouyang, Nanqing Dong, Yu Cheng, Siqi Sun, Lei Bai, and Bowen Zhou. From ai for science to agentic science: A survey on autonomous scientific discovery, 2025. URL https://arxiv.org/abs/2508.14111.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule, 2024a. URL https://arxiv.org/abs/2412.06464.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length, 2024b. URL https://arxiv.org/abs/2406.06484.

Sha Zhang, Suorong Yang, Tong Xie, Xiangyuan Xue, Zixuan Hu, Rui Li, Wenxi Qu, Zhenfei Yin, Tianfan Fu, Di Hu, Andres M Bran, Nian Ran, Bram Hoex, Wangmeng Zuo, Philippe Schwaller, Wanli Ouyang, Lei Bai, Yanyong Zhang, Lingyu Duan, Shixiang Tang, and Dongzhan Zhou. Position: Intelligent science laboratory requires the integration of cognitive and embodied ai, 2025. URL https://arxiv.org/abs/2506.19613.

Qichang Zhao, Heng Zhao, Liang Guo, Ke Zheng, Yu Li, Qiang Ling, Jun Tang, Yang Li, and Jun Wang. Coldstartcpi: Induced-fit theory-guided dti predictive model with improved generalization performance. *Nature Communications*, 16:6436, 2025. doi: 10.1038/s41467-025-61745-7. URL https://doi.org/10.1038/s41467-025-61745-7.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL https://arxiv.org/abs/2306.05685.

A Drug–Target Interaction Discovery Details

This appendix gives the full protocol and architecture analysis for the drug–target interaction (DTI) study reported in Section 5.4, which tests whether AI-discovered designs have practical value beyond AI benchmarks.

Task Formulation Drug–target interaction prediction asks whether a drug molecule and target protein bind strongly enough to interact. Accurate prediction prioritizes compounds before wet-lab screening. The model must encode drug and protein features and reason over cross-modal interactions. Because the design space over feature extraction, fusion, and interaction modules is large and discrete, we use DrugBAN (Bai et al., 2022) as the seed architecture.

Methodology We initialize cognition from roughly 80 papers on graph models, attention, and DTI. Each round, the Researcher proposes edits, the Engineer trains the modified DrugBAN on BindingDB-Dev, and the Analyzer reviews AUROC, AUPRC, F1, and MCC while flagging failures such as protein-length sensitivity or scaffold bias. We evaluate on BindingDB, Human, BioSNAP, and C.elegans under random, unseen-drug, unseen-protein, and double-unseen settings. Fitness uses AUROC/AUPRC as primary signals and F1/MCC as secondary criteria. Overall, we test 100+ candidates across 100+ rounds.

Analysis. The best architecture (ban_sinkhorn_ds_marginal_topk_v6) differs from DrugBAN in three ways: **Sinkhorn Attention** for balanced drug–protein matching, **Domain-Specific Marginalization** for compositional aggregation, and **Top-k Sparse Gating** to suppress noisy interactions. During evolution, cognition-guided proposals combine optimal transport and bipartite matching ideas, while the Analyzer surfaces collapse, overfitting, and attention saturation, later mitigated with entropy regularization and orthogonality constraints. Compared with expert baselines (Chen et al., 2020; Koh et al., 2024; Singh et al., 2023; Zhao et al., 2025), the evolved model is competitive or better, suggesting AI-driven evolution can help biomedical tasks.

B Analysis Configuration Details

This appendix summarizes the concrete experiment settings used in the circle-packing analysis section.

Framework and model comparison (Section 5.2).

- **Compared settings:** aligned comparison with OpenEvolve-style prompting, and backbone comparison between GPT-5-mini and Qwen3-32B under the same system design.
- **Alignment protocol:** prompt style, full-file evolution mode, MAP-Elites/island database, and population-related settings were aligned to the OpenEvolve circle-packing setup.
- **GPT-5-mini setting:** temperature 0.7, top-$p = 0.95$, max tokens 32768.
- **Qwen3-32B setting:** temperature 0.6, top-$p = 0.95$, max tokens 65536, seed $= 42$, thinking enabled, top-$k = 20$, and min-$p = 0$.
- **Shared system setting:** max size $= 70$, `sample_n=3`, engineer timeout $=$ 300s, 4 parallel workers, cognition retrieval top-$k = 5$, Judge disabled, and 3 runs per model.

Sampling algorithm comparison (Section 5.2).

- **Backbone model:** Qwen3-32B with the same decoding setting as above: temperature 0.6, top-$p = 0.95$, max tokens 65536, seed $= 42$, thinking enabled, top-$k = 20$, and min-$p = 0$.
- **Controlled factors:** prompt template, cognition setting, full-file evolution mode, engineer timeout, parallelism, and database size were held fixed.
- **Only changed factor:** database sampling algorithm. MAP-Elites uses the island configuration above, while UCB1 uses `algorithm=ucb1` with exploration constant $c = 1.414$.
- **Repeats:** 3 independent runs per sampling strategy.

Ablation study (Section 5.3).

- **Base model:** GPT-5-mini with temperature 0.7, top-$p = 0.95$, and max tokens 32768.
- **System loop:** Researcher–Engineer–Analyzer enabled by default; the "No Analyzer" setting disables Analyzer only, while "No Cognition" removes cognition usage but keeps the rest of the loop unchanged.
- **Researcher setting:** full-file generation, max code length 100000, and up to 3 retries.
- **Engineer setting:** timeout $=$ 300s, up to 2 retries, and 4 parallel workers.
- **Database setting:** max size $= 70$, sampling algorithm = island-based MAP-Elites, 5 islands, migration interval $= 10$, migration rate $= 0.1$, exploration ratio $= 0.2$, exploitation ratio $= 0.6$, feature dimensions = {`complexity`, `diversity`}, and 10 bins per feature.
- **Cognition setting:** retrieval top-$k = 5$, score threshold $= 0.4$, sentence-transformer embedding dimension $= 384$, and web search disabled.
- **Other shared settings:** `sample_n=3`, Judge disabled, and 3 independent runs per condition. All curves report the mean trend with variability across runs.

Cognition repository contents for circle packing. All experiments that use cognition share a common knowledge base initialized before the first evolution step. The 12 cognition items span three categories:

- **Geometric priors (4 items).**
  - *Hexagonal close packing:* theoretical density $\pi/(2\sqrt{3}) \approx 0.9069$ for infinite planes; unit-square boundary effects reduce achievable density; best patterns use hexagonal arrangements with careful corner and boundary handling.
  - *Edge and corner effects:* circles near boundaries are space-constrained; larger circles should be placed at corners and edges (corner radius up to $\frac{\sqrt{2}}{2}$ times the distance from the corner); small epsilon in overlap and bound checks avoids numerical tangency.
  - *Variable radii:* optimal solutions for $n = 26$ use circles of different sizes—larger circles at the center and corners, smaller ones filling gaps—with no uniform-radius assumption.

  - $n = 26$ *target knowledge:* the benchmark target is sum-of-radii ≈ 2.635 (AlphaEvolve); central hexagon plus outer layer arrangements work well; variable radii and corner optimization are the two most critical factors.
- **Optimization methodology (4 items).**
  - *SLSQP constrained optimization:* maximize sum of radii subject to explicit no-overlap ($d_{ij} \geq r_i + r_j + \varepsilon$) and in-bounds constraints via `scipy.optimize.minimize`; warm-start from high-scoring nodes (score ≥ 2.2).
  - *Multi-start strategy:* 3–5 different initial configurations from the best database nodes; optimize each independently and keep the best; tight tolerances with `maxiter` 500–1000.
  - *Differential evolution:* use `scipy.optimize.differential_evolution` for global refinement when local optimization plateaus; optionally refine radii only (lower-dimensional subspace), then polish with SLSQP.
  - *AlphaEvolve reference:* AlphaEvolve achieved 2.635 via constructor-plus-constrained-optimization; key insight is good geometric initialization followed by strict-constraint numerical refinement.
- **Engineering and troubleshooting guidelines (4 items).**
  - *Incremental refinement:* do not rewrite the full program; make targeted changes to optimizer settings, constraint formulation, multi-start count, or add a dedicated refinement stage.
  - *Code structure:* keep construction and optimization as separate stages; use a high-scoring node's code as base and apply diff-style edits.
  - *Numerical stability:* use a small epsilon (e.g. 10^{-8}) in all constraints to avoid numerical tangency; verify that no constraint is violated before reporting the sum of radii.
  - *Plateau-breaking checklist:* if stuck at ≈ 2.3–2.4, try (1) different initial pattern from a high-scoring node, (2) increase `maxiter` to 500–1000, (3) switch to explicit inequality constraints instead of penalty terms, (4) increase multi-start count to 3–5, (5) run differential evolution then SLSQP polishing.

Note on the framework comparison. OpenEvolve is commonly reported with a fast/slow model pair, where a second model can affect proposal quality and selection dynamics. In our aligned comparison, we instead instantiated both ASI-EVOLVE and OpenEvolve-style settings with the same backbone model in each run, so that the comparison focuses on framework design rather than auxiliary-model choice. We acknowledge that this simplification may introduce bias relative to the best possible OpenEvolve configuration, since a different auxiliary model could potentially improve its performance. However, using the same model on both sides avoids an additional confound: results would otherwise depend heavily on how the second model is chosen, and a poorly chosen auxiliary model could distort the comparison even more. For GEPA, we retained all official settings except model-related parameters. We set the score of invalid evaluation outcomes to 0; otherwise the system would record invalid nodes with anomalously high scores and distort evolution. All other GEPA configuration and evaluation logic is unchanged from the official release.

C Limitations, Reproducibility, and Broader Impact

Limitations. ASI-EVOLVE currently focuses on algorithmic discovery rather than low-level infrastructure optimization. For architecture design, the evolved attention-style modules are validated at the model and benchmark level, but we do not develop specialized GPU kernels or training-system optimizations comparable to mature hand-engineered attention implementations. For data curation and RL algorithm discovery, the final validation models are still limited in scale; larger-scale pretraining and RL runs could provide stronger evidence about how the discovered strategies behave in frontier-scale settings. For the RL task, our validation applies RL directly to base models, while many modern pipelines apply RL after mid-training or other intermediate adaptation stages; additional experiments after such mid-training would make the comparison closer to current large-model practice. We also do not run other evolution frameworks, such as OpenEvolve or GEPA, on the three AI-discovery

tasks themselves. These architecture, data-curation, and RL searches are costly, and those frameworks do not include the task-specific efficiency controls, static checks, debugging support, and staged validation used here, so direct large-scale comparisons would require substantially more compute. Nevertheless, we compare with these frameworks on the small-scale circle-packing task, providing a feasible controlled test of relative effectiveness. For DTI, we evaluate by offline prediction benchmarks only and do not conduct wet-lab validation, which would require additional resources.

Reproducibility and compute. The main paper reports the primary setup and results for each study, while the appendix adds detailed configuration for the circle-packing analysis. The framework and our circle-packing experiment setup are released at https://github.com/GAIR-NLP/ASI-Evolve to support controlled verification of effectiveness. The three main AI-discovery studies have broad search and evaluation scopes and include stochastic training/evolution components; therefore, even with matched configurations, reruns may reproduce the overall improvement trends without exactly matching every trajectory or final score. Complete experimental artifacts for the architecture, data-curation, RL, and DTI studies are not released because of their large scale. We used two H200 GPUs for architecture-search validation under the reported training setting. For data-curation training runs, we used 128 H200 GPUs. The RL algorithm validation used 8 H200 GPUs.

Broader impacts and safeguards. The intended positive impact is to let AI systems complete more of the routine search, implementation, evaluation, and analysis in scientific discovery, giving researchers more opportunity to focus on problem design, objectives, evaluations, and research directions. Potential risks include over-trusting automatically generated designs, optimizing to incomplete benchmarks, or applying automated discovery loops to harmful objectives. In this work, experiments are offline and evaluator-bounded; we do not release high-risk pretrained models, scraped datasets, or deployable biomedical products.

LLM usage. LLMs are central to the proposed method: they act as Researcher, Engineer, and Analyzer agents in the automated evolution loop. LLMs were also used for editing assistance, but the scientifically relevant use is their role as agents that propose, implement, evaluate, and analyze candidate designs.