BIODYAD: SYNCHRONIZE BIOMEDICAL DISCOVERY AND MACHINE LEARNING ENGINEERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Agentic biomedical machine learning (ML) draws on complementary advances in biomedical evidence acquisition and executable program search. Existing systems connect aspects of these capabilities, but coordinating them throughout program search remains challenging. New evidence must guide candidate construction, execution outcomes must inform subsequent discovery and reuse, and validation demands must fit the search budget. We introduce *BioDyad*, which couples biomedical discovery and ML engineering through two hierarchies within Monte Carlo graph search. Its scientific hierarchy combines prior biomedical guidance with iterative discovery, then links biomedical plans to execution outcomes in memory for reuse across candidates. Its engineering hierarchy moves candidate programs from smoke execution, through train/validation evaluation, to full-data retraining. We evaluate BioDyad on the 76-task BioXArena benchmark under a two-hour per-task budget with three matched LLM backends. It achieves the highest penalized all-task score and task success rate among four agent methods and a one-shot baseline under each backend. These results support coordinating biomedical discovery and ML engineering to integrate external knowledge into executable programs across heterogeneous biomedical tasks. Code Available: `https://anonymous.4open.science/r/BioDyad-7755/`

1 INTRODUCTION

Machine learning (ML) agents are increasingly capable of constructing and refining complete learning programs rather than selecting models in isolation (Chan et al., 2025; Jiang et al., 2025; Du et al., 2026). This capability is especially consequential in biomedicine, where a useful program must translate task-specific biomedical evidence into executable modeling decisions across heterogeneous data modalities and task families (Miller et al., 2025; Li et al., 2026). Knowing biomedical methods does not itself provide access to new measurements or private institutional resources. Using such resources requires identifying relevant evidence, integrating it into candidate programs, and checking its empirical value. As search proceeds, newly acquired evidence and execution outcomes can change which representations, models, and validation strategies are appropriate (Huang et al., 2025; Jin et al., 2025). Autonomous biomedical ML therefore requires biomedical discovery and machine learning engineering (MLE) to advance together.

Existing work broadly follows two complementary research streams with different emphases. General-purpose MLE agents develop executable programs through iterative search, execution feedback, and experience reuse (Jiang et al., 2025; Du et al., 2026; Zhu et al., 2026). Biomedical agents acquire and interpret domain evidence through specialized resources, tools, and reasoning workflows (Huang et al., 2025; Jin et al., 2025; Roohani et al., 2025). These streams already intersect in systems such as VCHarness, which combines biomedical profiling and biological models with program search and feedback memory (Cheng et al., 2026). Its node lifecycle centers on full fine-tuning and does not explicitly adapt exploration to the remaining budget.

Table 1 distinguishes the capabilities involved in coordinating these two streams. Biomedical discovery acquires task-specific evidence, and biomedical memory retains discovery plans and execution outcomes across candidates. Program search compares executable alternatives, while staged evaluation and budget-adaptive search govern how candidates are tested and exploration effort is allocated. However, sustaining this coordination throughout program search remains challenging:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: Capabilities of representative agentic systems for biomedical ML program discovery.

Type	Method	Scientific hierarchy		Search and engineering		
		Bio. discovery	Bio. memory	Program search	Staged eval.	Budget adapt.
Harness	BioXArena harness (Li et al., 2026)	✗	✗	✗	✗	✗
Bio. agents	Biomni (Huang et al., 2025)	✓	✗	✗	✗	✗
	STELLA (Jin et al., 2025)	✓	✗	✗	✗	✗
	VCHarness* (Cheng et al., 2026)	△	△	✓	✗	✗
MLE agents	MLEvolve (Du et al., 2026)	✗	✗	✓	✗	✓
	ML-Master 2.0 (Zhu et al., 2026)	✗	✗	✓	✗	✓
Coupled	**BioDyad (ours)**	✓	✓	✓	✓	✓

* VCHarness profiles biomedical data and curated biological models; its memory primarily stores program feedback.

new biomedical evidence must guide candidate revisions, while execution feedback informs subsequent discovery under a fixed budget. Integrating such evidence can introduce errors and additional validation demands, making execution cost part of the coordination problem.

To address this gap, we introduce *BioDyad*, a hierarchical agentic search framework that coordinates biomedical discovery and MLE throughout program search. Within a shared Monte Carlo graph search (MCGS) backbone (Du et al., 2026), its *scientific hierarchy* combines task-level guidance with discovery tailored to the candidate being expanded. Biomedical memory links these plans to execution outcomes, allowing later candidates to reuse productive integrations and revisit failed attempts. Its *engineering hierarchy* evaluates programs through progressively more demanding execution stages, providing feasibility and predictive feedback on biomedical proposals during search.

We evaluate BioDyad on the heterogeneous BioXArena benchmark (Li et al., 2026) under a two-hour per-task budget. Under each of three matched LLM backends, BioDyad attains the highest penalized all-task score and success rate among four baselines. Component ablations on two domains support the combined use of biomedical discovery, memory, and staged execution.

This paper makes four main contributions.

1) We introduce **BioDyad**, a hierarchical agentic search framework that coordinates **biomedical discovery and ML engineering** throughout program search under a fixed budget. Biomedical evidence and execution outcomes jointly guide subsequent candidate revisions.

2) We develop a **scientific hierarchy** that combines prior guidance, candidate-level discovery, and plan-outcome memory to support iterative biomedical resource integration. This enables later candidates to reuse and revise prior integration decisions.

3) We develop an **engineering hierarchy** that advances knowledge-augmented programs from smoke execution through train/validation evaluation to full-data retraining and submission. These stages separate feasibility screening, predictive assessment, and final model fitting.

4) We establish BioDyad's aggregate advantage on the 76-task BioXArena benchmark through a comparison under three matched LLM backends. Component ablations examine the contribution of the coupled hierarchies, and extended-search trajectories characterize progress within BioDyad as more time becomes available.

2 Related Work

Agents for Biomedical Discovery. Biomedical agents combine LLM reasoning with domain tools, data, and experimental context. Biomni unifies biomedical resources through a code interface (Huang et al., 2025), while STELLA adds evolving reasoning templates and autonomous tool creation (Jin et al., 2025). Targeted systems close narrower loops: BioDiscoveryAgent uses knowledge and prior outcomes to propose genetic perturbations (Roohani et al., 2025), while the Virtual Lab coordinates specialized agents with human feedback to design and validate nanobodies (Swanson et al., 2025). VCHarness directly targets biomedical ML program development by coupling a coding agent with biological foundation models for perturbation-response search (Cheng et al., 2026). Together, these systems establish knowledge use, tool orchestration, experimental design, and domain-specific program search. BioDyad centers their coupling by revising candidate-level findings and retaining them with execution outcomes to guide later search.

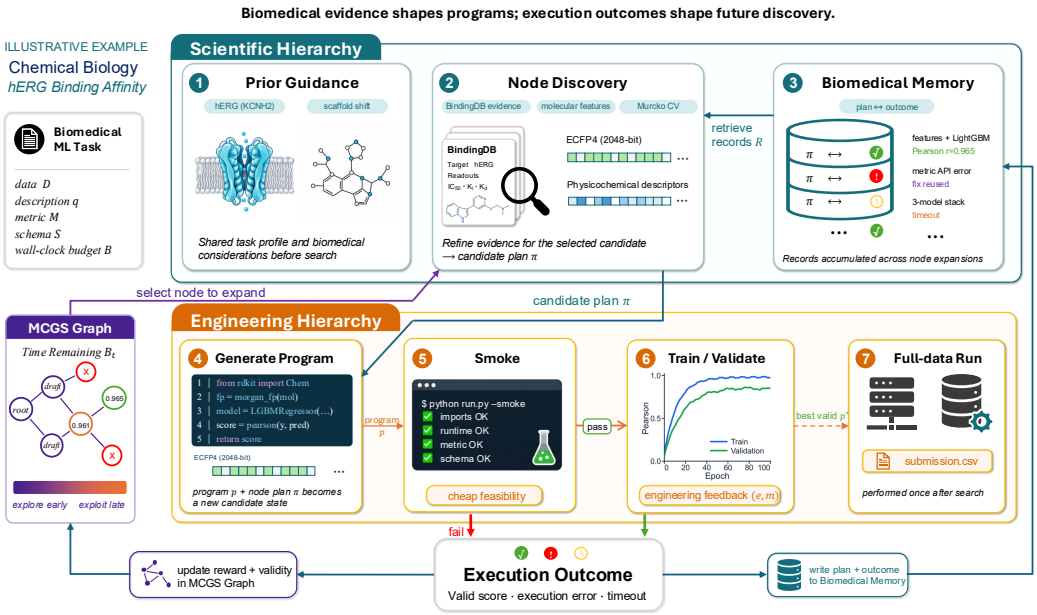


Figure 1: BioDyad couples biomedical discovery and ML engineering within MCGS. The scientific hierarchy combines prior guidance, node-level discovery, and biomedical memory; the engineering hierarchy advances programs through smoke execution, train/validation evaluation, and full-data runs. Biomedical evidence guides candidate expansion, while execution feedback updates the search graph and memory within the wall-clock budget. The *hERG Binding Affinity* task from BioXArena (Li et al., 2026) provides an illustrative example.

Agentic Search for Executable ML Programs. Classical AutoML optimizes model components and hyperparameters within predefined search spaces, as exemplified by auto-sklearn (Feurer et al., 2015). LLM agents instead search executable code: MLAgentBench and MLGym support iterative experimentation (Huang et al., 2024; Nathani et al., 2025), MLE-Bench evaluates competition-scale performance (Chan et al., 2025), and AIDE performs tree search over code solutions (Jiang et al., 2025). For longer horizons, MLEvolve combines progressive MCGS with retrospective cross-branch memory (Du et al., 2026), while ML-Master 2.0 uses hierarchical cognitive caching (Zhu et al., 2026). BioML-bench and BioXArena extend end-to-end evaluation to heterogeneous biomedical tasks and modalities (Miller et al., 2025; Li et al., 2026). These systems establish program search, memory, and empirical feedback. BioDyad builds on MLEvolve's search to integrate external biomedical resources through candidate-level discovery and staged, budget-aware validation.

3 Problem Formulation

We consider a biomedical ML task $\tau = (\mathcal{D}, q, M, \mathcal{S}, B)$, where $\mathcal{D}$ denotes the public task data, q the natural-language task description, M the evaluation metric, $\mathcal{S}$ the required submission schema, and B the total wall-clock budget. The search space contains complete programs that load data, train and validate a predictive model, and produce a submission conforming to $\mathcal{S}$. Let $m_\tau(p)$ be the validation score of a valid program and let $V_\tau(p) \in \{0, 1\}$ indicate whether p executes and satisfies the task contract. If $d_\tau \in \{-1, +1\}$ encodes the optimization direction of M, the objective is

$$p^\star = \arg\max_{p \in \mathcal{P}_B} \; d_\tau m_\tau(p) \quad \text{subject to} \quad V_\tau(p) = 1, \tag{1}$$

where $\mathcal{P}_B$ is the set of candidates explored within budget B. The program and its supporting biomedical evidence are coupled: execution outcomes inform subsequent resource use, while incorporating new evidence can increase validation cost and fragility. BioDyad addresses it by evolving biomedical evidence across search and advancing the resulting programs through staged execution.

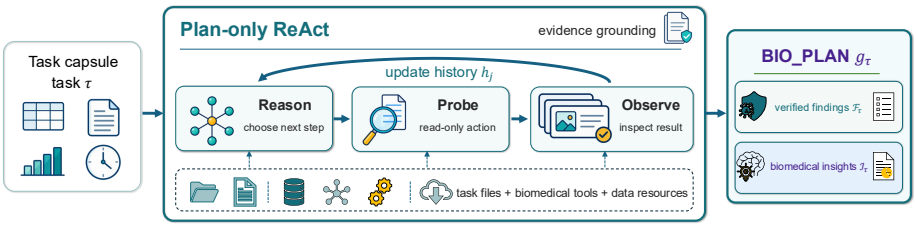


Figure 2: ReAct-based prior biomedical guidance. A plan-only agent alternates reasoning with read-only probes to compile verified data findings and biomedical insights.

4 BioDyad

Overview. Fig. 1 summarizes the coupled search loop. At iteration t, BioDyad maintains an MCGS program graph G_t and a biomedical memory $\mathcal{K}_t$. Each graph node v indexes a candidate state

$$x_v = (p_v, \pi_v, e_v, m_v), \tag{2}$$

where p_v is its program, π_v its biomedical node plan, e_v its execution outcome, and m_v its validation score; e_v and m_v remain pending until execution. Given a parent node v_t, the scientific hierarchy retrieves relevant records and produces a plan for a child node v', after which the engineering hierarchy instantiates and evaluates its program. Here, g_τ denotes the task-level prior guidance:

$$\begin{aligned} R_t &= \mathrm{Retrieve}(\mathcal{K}_t, x_{v_t}), & \pi_{v'} &= \mathrm{Discover}(\tau, g_\tau, x_{v_t}, R_t), \\ p_{v'} &= \mathrm{Generate}(p_{v_t}; g_\tau, \pi_{v'}, R_t), & (e_{v'}, m_{v'}) &= \mathrm{Engineer}(p_{v'}). \end{aligned} \tag{3}$$

The child produces G_{t+1}, while its plan and observed outcome produce $\mathcal{K}_{t+1}$. Biomedical evidence therefore changes the programs, and execution changes the evidence available to later expansions.

4.1 Scientific Hierarchy

The scientific hierarchy controls how biomedical evidence is initialized, revised, and retained. It combines shared prior guidance, node-level discovery, and a memory that links candidate plans to their computational consequences.

4.1.1 Biomedical Discovery

Prior guidance. Before search, BioDyad runs a plan-only biomedical agent to establish shared prior guidance g_τ (Fig. 2). Its ReAct-style loop (Yao et al., 2023) alternates reasoning with lightweight, read-only probes of task files and, when relevant, retrieved biomedical tools and data resources.

Let h_{j-1} denote the reasoning-observation history before probe step j. The biomedical planner $\mathcal{A}_{\mathrm{bio}}$ selects action a_j, executes it through the available tool interface $\mathcal{T}$, and appends the observation o_j:

$$a_j \sim \mathcal{A}_{\mathrm{bio}}(\cdot \mid \tau, h_{j-1}), \qquad o_j = \mathcal{T}(a_j), \qquad h_j = h_{j-1} \oplus (a_j, o_j). \tag{4}$$

When the agent terminates the loop, it compiles the accumulated evidence into

$$g_\tau = \mathrm{Compile}(h_J) = \big(\mathcal{F}_\tau, \mathcal{I}_\tau\big), \tag{5}$$

where $\mathcal{F}_\tau$ contains verified data findings and $\mathcal{I}_\tau$ contains biomedical implications supported by those findings. The resulting `BIO_PLAN` conditions subsequent node discovery without prescribing a model family or acting as an MCGS operator.

Discovery during node expansion. Before generating each draft, improvement, or debugging node, BioDyad invokes a biomedical discovery agent with the task, prior guidance, selected parent, and retrieved memory. The agent may inspect relevant data and resources, then returns a candidate-specific `NODE_PLAN` $\pi_{v'}$ that binds the biomedical rationale to an implementable modeling and validation strategy. Since discovery occurs after earlier candidates have been evaluated, it can refine supported directions, abandon contradicted ones, or branch to a distinct hypothesis. Biomedical knowledge is therefore revised throughout program evolution rather than injected only at initialization.

4.1.2 BIOMEDICAL MEMORY

Biomedical memory extends MLEvolve's experience reuse (Du et al., 2026) to plans informed by biomedical discovery and their execution outcomes. For each MCGS node, biomedical memory stores a compact record

$$k_v = (\pi_v, s_v, e_v, m_v, \Delta m_v, \ell_v), \qquad \mathcal{K}_{t+1} = \mathrm{Upsert}(\mathcal{K}_t, k_{v'}). \tag{6}$$

Here, s_v is the search stage, Δm_v is the score change relative to the parent, and ℓ_v marks the record as successful or failed. The plan is inserted before code generation and the same keyed record is updated after execution. Both successful and failed attempts remain available for retrieval, alongside a contrasting viable direction when useful. Subsequent discovery can reuse or revise earlier attempts to integrate external biomedical knowledge using their recorded execution outcomes. The record connects what a candidate attempted with what happened during execution and how its score changed. This preserves the distinction between an integration error and an executable change with little predictive benefit, giving subsequent discovery a basis for different revisions.

4.2 ENGINEERING HIERARCHY

The engineering hierarchy turns knowledge-augmented programs into comparable feedback without committing the full search budget to every candidate. It separates feasibility, comparative evaluation, and final retraining: *1) Smoke Execution.* Each generated program first undergoes static checks and smoke execution. A reduced-cost smoke run retains the candidate's feature representation, model, and output format to test whether the proposed integration is executable. The candidate advances only if it completes without an exception, reports the required validation score, and produces a submission artifact. Otherwise, its failure is recorded and control returns to MCGS rather than entering a nested repair loop. *2) Train/Validation Evaluation.* Smoke-passing candidates run their intended train/validation program. The evaluator extracts the task metric, verifies the required artifact and schema, and returns validity and a score m_v. These outcomes update both MCGS and the plan-outcome record in biomedical memory. This stage assesses predictive quality under the candidate's intended training and validation settings, supplying the scores used to compare executable alternatives. *3) Full-Data Retraining and Submission.* After search, BioDyad selects the best valid program using its pre-retraining validation score. A final variant retrains the same approach on all labeled public data and writes the submission.

4.3 COUPLING THE HIERARCHIES

At each search iteration, MCGS (Du et al., 2026) selects a node to expand according to its execution history and the remaining wall-clock budget. The selected parent and retrieved plan-outcome records condition the next biomedical plan, which guides construction of the child program in Eq. 3. Staged execution then updates two states with different roles. Validity and validation performance inform subsequent node selection in the search graph, while biomedical memory retains the rationale associated with each outcome. Later discovery can therefore revise resource integration using feedback from earlier candidates.

Implementation settings are detailed in Appdx. B.2, and the MCGS backbone and search configuration in Appdx. B.3.

5 EXPERIMENTS

5.1 BENCHMARK AND EVALUATION PROTOCOL

Benchmark. BioXArena (Li et al., 2026) contains 76 held-out prediction tasks across nine biomedical domains: sequence, single-cell, structure, network biology, chemical biology, perturbation dynamics, phenotype-disease, imaging, and text-integrated biology. The benchmark is deliberately heterogeneous, with 46 of the 76 tasks combining multiple modality families. Each public task capsule contains a task description, public train/test inputs, native-format assets when required, a sample submission, and a task-specific metric; test labels and evaluators remain hidden.

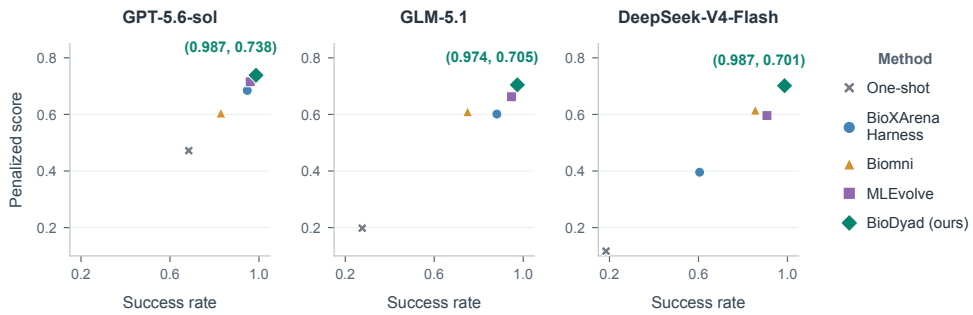


Figure 3: Success rate (x-axis) and penalized all-task score (y-axis) under three LLM backends. Each panel compares four agent methods with a two-hour per-task budget and a one-shot baseline retaining only the first attempt, evaluated on the same 76 tasks.

Execution protocol. We follow the official BioXArena interface. An agent receives the public task capsule in a no-internet sandbox, trains predictive models under a two-hour wall-clock limit and the prescribed CPU/GPU allocation, and writes `submission.csv`. BioXArena permits additional biomedical knowledge through a locally mounted data cache (Li et al., 2026); BioDyad uses local Biomni data and tools (Huang et al., 2025) during biomedical discovery (Appdx. B.1). All phases, including prior planning, candidate discovery and evaluation, and final retraining and submission, are included in the two-hour limit. The primary comparison fixes the task interface, hardware allocation, and time limit across five methods under each LLM backend. Extended-budget and repeated-run protocols are specified separately below.

Methods and backends. The primary comparison evaluates BioDyad, MLEvolve (Du et al., 2026), Biomni (Huang et al., 2025), and BioXArena Harness (Li et al., 2026) with each of GPT-5.6-sol (OpenAI, 2026b), GLM-5.1 (Z.AI, 2026), and DeepSeek-V4-Flash Preview (Xu et al., 2026). In this comparison, Biomni accesses its own data lake and tools, whereas MLEvolve and BioXArena Harness do not. The resource-matched control, MLEvolve w/ discovery, is reported in the component ablations (Table 4). Fig. 3 and Table 2 additionally include a one-shot baseline for each backend, retaining only the first attempt per task from the corresponding BioXArena Harness run and scoring failures as zero. For broader context, Appdx. A compares BioDyad with the results presented on BioXArena's leaderboard, retaining each baseline's reported backend configuration.

Scoring. BioXArena maps each task metric to $[0, 1]$, with larger values indicating better performance. Let $s_{a,i}$ be the score of agent a on task i, with $s_{a,i} = 0$ for a failed or missing submission. We report the penalized mean score $\bar{s}_a$ over all 76 tasks and the success rate R_a, defined as the fraction of tasks with successful execution and valid submissions.

5.2 Main Results

BioDyad achieves the highest penalized all-task score and execution success rate under each of the three LLM backends (Fig. 3). This aggregate advantage extends across most domains: BioDyad attains the highest or second-highest distinct reported score in eight, eight, and nine of nine domains with GPT-5.6-sol, GLM-5.1, and DeepSeek-V4, respectively (Table 2). Perturbation dynamics is a consistent strength: BioDyad leads this domain under every backend. The leading methods in other domains vary with the backend, showing that the aggregate result combines a consistent domain advantage with broad competitiveness across task families. BioDyad also successfully completes nearly all tasks under each backend. Additionally, all four agent methods exceed the corresponding one-shot average under every backend. The one-shot success rates of 0.684, 0.276, and 0.184, respectively, highlight execution reliability as a limitation alongside predictive quality. Biomni, despite using the same data lake, stays within 0.605-0.615.

More specifically, BioDyad improves the penalized score over the strongest matched alternative by 0.023, 0.042, and 0.086 with GPT-5.6-sol, GLM-5.1, and DeepSeek-V4, respectively (computed before rounding). It also narrows the penalized-score gap between GPT-5.6-sol and DeepSeek-V4 from 0.119 with MLEvolve to 0.037. With DeepSeek-V4, BioDyad reaches 0.701, approaching MLEvolve with GPT-5.6-sol (0.715). These gains support coupling biomedical discovery with ex-

Table 2: Domain breakdowns of four agent methods and a one-shot baseline under three matched LLM backends on the 76 BioXArena tasks. Scores are penalized average scores; higher is better. Avg. is weighted by task count across all 76 tasks. **Bold** and underlining indicate the highest and second-highest distinct displayed scores, respectively, within each backend and column.

LLM backend	Method	Seq	SC	Str	Net	Chem	Pert	Phen	Img	Text	Avg.
GPT-5.6-sol	One-shot	0.416	0.288	0.727	0.458	0.576	0.291	0.508	0.500	0.546	0.472
	BioXArena Harness	0.663	0.786	**0.834**	0.458	**0.928**	0.517	**0.606**	**0.714**	0.637	0.685
	Biomni	0.773	0.652	0.591	0.490	0.690	0.404	0.431	0.702	0.656	0.605
	MLEvolve	**0.790**	0.798	0.794	**0.527**	0.922	0.597	0.575	0.673	**0.720**	0.715
	BioDyad (ours)	0.776	**0.861**	0.815	0.521	0.926	**0.776**	0.594	0.666	0.671	**0.738**
GLM-5.1	One-shot	0.187	0.102	0.236	0.062	0.556	0.121	0.374	0.139	0.030	0.198
	BioXArena Harness	0.649	0.753	**0.824**	**0.511**	0.894	0.336	0.487	0.513	0.393	0.601
	Biomni	0.580	0.669	0.796	0.487	0.912	0.618	0.310	0.639	0.467	0.609
	MLEvolve	0.713	**0.791**	0.729	**0.511**	0.919	0.621	0.584	0.439	**0.610**	0.662
	BioDyad (ours)	**0.776**	0.771	0.729	0.509	**0.932**	**0.737**	**0.600**	**0.700**	0.552	**0.705**
DeepSeek-V4	One-shot	0.085	0.095	0.095	0.137	0.229	0.111	0.072	0.000	0.244	0.117
	BioXArena Harness	0.623	0.423	0.095	0.508	0.506	0.183	0.398	0.353	0.408	0.396
	Biomni	0.673	0.618	0.720	0.431	0.870	0.571	0.510	0.621	0.504	0.615
	MLEvolve	**0.739**	0.761	0.733	0.442	0.902	0.578	0.440	0.386	0.306	0.596
	BioDyad (ours)	0.717	**0.791**	**0.789**	**0.510**	**0.909**	**0.717**	**0.600**	**0.646**	**0.608**	**0.701**

Domains: Seq, sequence; SC, single-cell; Str, structure; Net, network biology; Chem, chemical biology; Pert, perturbation dynamics; Phen, phenotype-disease; Img, imaging; Text, text-integrated.

ecution throughout search across the tested backends. The component ablations below examine the contributions of the two hierarchies.

Repeated-Run Robustness. To assess run-to-run variability across biomedical domains, we fix an 18-task subset comprising the first two tasks in each domain in BioXArena's Table A4 (Li et al., 2026). We use GPT-5.6-sol for BioXArena Harness, Biomni, MLEvolve, and BioDyad with three independent runs per method-task pair under the two-hour budget. Each run starts from a fresh workspace. The task list and per-run results are in Appdx. E.1. Across three runs with GPT-5.6-sol, BioDyad combines the highest mean penalized score with a small sample SD across runs (Table 3).

Table 3: Repeated-run robustness on the fixed 18-task subset with GPT-5.6-sol. Values are mean ± sample standard deviation (SD) across three independent runs.

Method	Penalized score	Success rate
BioXArena Harn.	0.699 ±0.005	**0.944** ±0.000
Biomni	0.647 ±0.012	0.889 ±0.000
MLEvolve	0.688 ±0.019	0.907 ±0.032
BioDyad	**0.711** ±0.002	**0.944** ±0.000

5.3 Component Ablations

Table 4: Component ablations with the GPT-5.4 backend. Scores are penalized task averages. Avg. is the task-weighted mean across all 16 tasks in the two domains; higher is better.

Hierarchy	Mechanism	MLEvolve	MLEvolve w/ discovery	BioDyad w/o memory	BioDyad w/o Sci.	BioDyad w/o Eng.	BioDyad (full)
Sci. hierarchy	Bio. discovery	✗	✓	✓	✗	✓	✓
	Bio. memory	✗	✗	✗	✗	✓	✓
Eng. hierarchy	Staged evaluation	✗	✗	✓	✓	✗	✓
Performance ↑	Chemical biology	0.898	0.795	0.913	0.861	0.896	**0.919**
	Phenotype-disease	0.538	0.599	0.560	0.468	0.570	**0.606**
	Average	0.718	0.697	0.736	0.665	0.733	**0.763**

Following BioXArena's scaling analysis (Li et al., 2026), we ablate the scientific and engineering hierarchies on chemical-biology and phenotype-disease tasks (Table 4). The ablations remove either the scientific hierarchy, comprising biomedical discovery (Sec. 4.1.1) and memory (Sec. 4.1.2), or staged evaluation (Sec. 4.2). We also remove biomedical memory alone, retaining discovery and staged evaluation. MLEvolve w/ discovery adds discovery without biomedical memory or staged evaluation, matching the full framework's biomedical resources and discovery capabilities. To decouple component analysis from the three main evaluation backends, all ablation variants use GPT-5.4 (OpenAI, 2026a) under the same two-hour task budget.

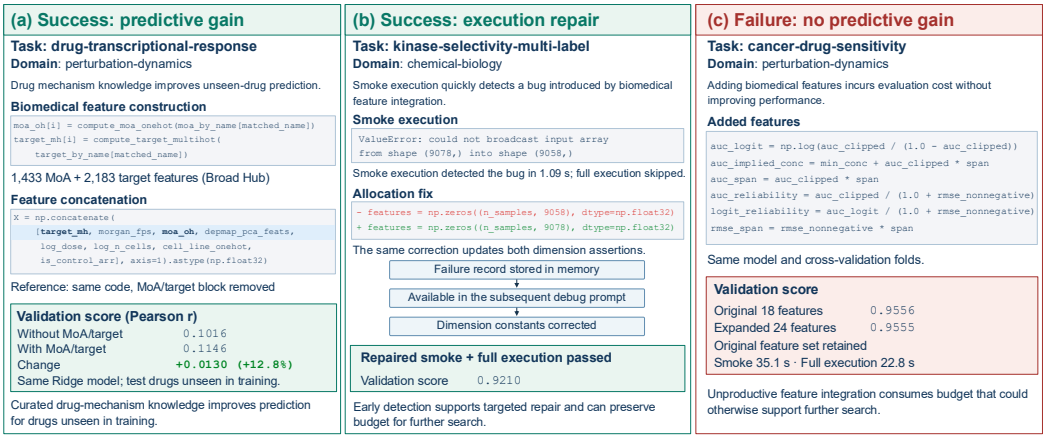


Figure 4: Biomedical feature integration in BioDyad: (a) predictive gain, (b) smoke-guided execution repair, and (c) no predictive gain despite successful execution. Scores are task-specific validation metrics; full execution denotes candidate train/validation evaluation.

The full framework achieves the highest average score (0.763), compared with 0.718 for MLEvolve and 0.697 for MLEvolve w/ discovery (Table 4). Biomedical resources thus pay off only when coupled with memory or staged evaluation. Removing biomedical memory or staged evaluation reduces the average to 0.736 and 0.733, respectively. These comparisons support contributions from both memory and staged evaluation when biomedical discovery is retained.

Discovery alone improves phenotype-disease prediction but reduces chemical-biology performance relative to MLEvolve. Staged evaluation without the scientific hierarchy yields 0.665, below MLEvolve. Adding discovery alone therefore does not reproduce the benefit of the complete framework. The two domains favor different partial configurations, yet both reach their highest score when discovery, memory, and staged evaluation are used together. Candidate-level diagnostics in Appdx. C.2 and C.3 characterize the corresponding screening and execution outcomes.

5.4 Case Studies

Fig. 4 illustrates why biomedical discovery and staged execution are complementary. Biomedical evidence can improve predictive representations, but its integration can also introduce execution errors. Smoke execution provides early feedback for repair, while train/validation evaluation tests whether executable changes improve prediction. Appdx. E.3 details the tasks and candidate changes.

Successful case: Predictive gain. On the *drug-transcriptional-response* task (Srivatsan et al., 2020), test drugs are unseen during training and identified only by name. Mechanism-of-action (MoA) and target annotations from the Drug Repurposing Hub (Corsello et al., 2017) raised validation Pearson r from 0.1016 to 0.1146 with the same model.

Successful case: Smoke-guided repair. On the *kinase-selectivity-multi-label* task (Mendez et al., 2019), additional molecular features introduced a dimension mismatch. Smoke execution detected the error in 1.09 s and skipped full execution. The recorded feedback guided a subsequent repair that retained the added features and restored successful execution.

Failed case: No predictive gain. On the *cancer-drug-sensitivity* task (Iorio et al., 2016), added biomedical features executed successfully but did not improve prediction, so the original representation was retained. The unproductive extension consumed search budget.

5.5 Budget Extension within BioDyad

We examine whether extending search helps BioDyad discover better candidate programs with GPT-5.6-sol. We use the tasks as in the component ablations. At each two-hour checkpoint of a 12-hour run, we retrospectively record the highest normalized hidden-test score among candidates completed by that time. Reading all checkpoints from one trajectory per task tracks how the candidate pool improves as search continues.

Table 5: Retrospective best-so-far normalized test scores during 12-hour BioDyad runs with GPT-5.6-sol. Each checkpoint averages task-level maxima over all eight tasks in a domain, using the same runs as Appdx. D. Gain columns report absolute differences computed before rounding.

Domain	Best-so-far test score						Score gain		
	2 h	4 h	6 h	8 h	10 h	12 h	2-4 h	4-12 h	2-12 h
Chemical biology	0.927	0.932	0.936	0.936	0.936	0.937	+0.006	+0.004	+0.010
Phenotype-disease	0.640	0.663	0.688	0.690	0.690	0.690	+0.022	+0.027	+0.050

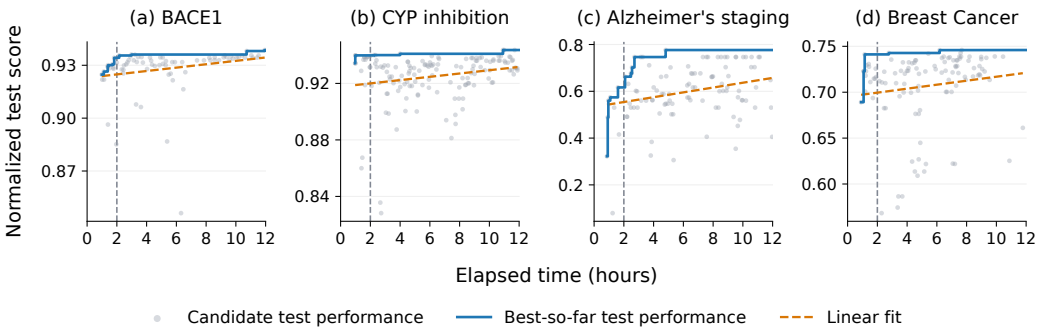


Figure 5: Candidate test scores during 12-hour searches: (a) BACE1 binding affinity, (b) CYP inhibition, (c) Alzheimer's disease staging, (d) breast-cancer subtype. Gray points show candidate scores; blue lines show retrospective maxima; orange dashed lines show linear fits against time.

Mean best observed normalized test scores rise by 0.010 in chemical biology and 0.050 in phenotype-disease from 2 to 12 hours (Table 5). Both domains gain during 2-4 h and 4-12 h, with slightly larger later gains in phenotype-disease. Extended search thus uncovers better candidates after the standard budget.

Fig. 5 shows how these gains emerge from candidate exploration. Alzheimer's disease staging improves after the two-hour checkpoint and plateaus at about five hours, showing that better candidates can emerge beyond the initial search period. BACE1 and CYP inhibition reach high scores early and improve more modestly thereafter. Breast-cancer subtype shows a similar early plateau despite substantial variation among later candidates. The positive slopes of ordinary least-squares linear fits in all four examples indicate an upward trend in average candidate test performance over time, despite variation among individual candidates.

Returns to additional search are thus task-dependent. The domain means aggregate gains accumulated at different stages of these trajectories. Appdx. D provides the complete task-level test-score trajectories and checkpoints.

6 Conclusion

We introduced BioDyad, a hierarchical agentic search framework that coordinates biomedical discovery and ML engineering under a fixed budget. Its scientific hierarchy combines prior guidance, candidate-level discovery, and biomedical memory, while its engineering hierarchy supplies feasibility and predictive feedback through staged evaluation. On 76 BioXArena tasks, BioDyad achieves the highest penalized all-task score and success rate under each of three matched LLM backends. Ablations in chemical biology and phenotype-disease support the joint use of discovery, memory, and staged evaluation, beyond adding discovery alone. Together, these results support using execution outcomes to guide how biomedical evidence is integrated and reused throughout program search. The current study has three limitations: (i) biomedical discovery depends on the quality, coverage, and accessibility of external resources; (ii) prior guidance and candidate-level discovery require additional LLM calls within the available budget; and (iii) evaluation covers supervised BioXArena tasks, with component and extended-budget analyses restricted to two domains. Extension to wet-lab or other long-horizon scientific workflows remains to be evaluated.

AI USE STATEMENT

We used generative AI tools to polish the language and improve the readability of the manuscript, and to generate initial drafts of figures. The authors take full responsibility for the accuracy and integrity of the final manuscript, including all AI-assisted text and figures.

REFERENCES

Alibaba Cloud. Alibaba Unveils Qwen3.6-Plus to Accelerate Agentic AI Deployment for Enterprises and Alibaba's AI Applications, April 2 2026. URL https://www.alibabacloud.com/press-room/alibaba-unveils-qwen3-6-plus-to-accelerate-agentic.

Anthropic. Introducing Claude 4, May 22 2025. URL https://www.anthropic.com/news/claude-4.

Anthropic. Introducing Claude Opus 4.6, February 5 2026a. URL https://www.anthropic.com/news/claude-opus-4-6.

Anthropic. Introducing Claude Sonnet 4.6, February 17 2026b. URL https://www.anthropic.com/news/claude-sonnet-4-6.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, et al. Mle-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations*, volume 2025, pp. 50466–50494, 2025.

Xingyi Cheng, Pan Li, Han Guo, Youwei Liang, Jing Gong, William de Vazelhes, Changjiang Gou, Pengtao Xie, Le Song, and Eric Xing. Harnessing ai to build virtual cells. *bioRxiv*, pp. 2026–04, 2026.

Steven M Corsello, Joshua A Bittker, Zihan Liu, Joshua Gould, Patrick McCarren, Jodi E Hirschman, Stephen E Johnston, Anita Vrcic, Bang Wong, Mariya Khan, et al. The drug repurposing hub: a next-generation drug library and information resource. *Nature medicine*, 23(4): 405–408, 2017.

Shangheng Du, Xiangchao Yan, Jinxin Shi, Zongsheng Cao, Shiyang Feng, Zichen Liang, Boyuan Sun, Tianshuo Peng, Yifan Zhou, Xin Li, et al. Mlevolve: A self-evolving framework for automated machine learning algorithm discovery. *arXiv preprint arXiv:2606.06473*, 2026.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. *Advances in neural information processing systems*, 28, 2015.

Kexin Huang, Serena Zhang, Hanchen Wang, Yuanhao Qu, Yingzhou Lu, Yusuf Roohani, Ryan Li, Lin Qiu, Gavin Li, Junze Zhang, et al. Biomni: A general-purpose biomedical ai agent. *biorxiv*, 2025.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentbench: Evaluating language agents on machine learning experimentation. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=1Fs1LvjYQW.

Francesco Iorio, Theo A Knijnenburg, Daniel J Vis, Graham R Bignell, Michael P Menden, Michael Schubert, Nanne Aben, Emanuel Gonçalves, Syd Barthorpe, Howard Lightfoot, et al. A landscape of pharmacogenomic interactions in cancer. *Cell*, 166(3):740–754, 2016.

Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. Aide: Ai-driven exploration in the space of code. *arXiv preprint arXiv:2502.13138*, 2025.

Ruofan Jin, Zaixi Zhang, Mengdi Wang, and Le Cong. Stella: Self-evolving llm agent for biomedical research. *arXiv preprint arXiv:2507.02004*, 2025.

Loka Li, Duzhen Zhang, Xingbo Du, Leonard Song, Zixiao Wang, Assanali Aukenov, Noel Thomas, Shakhnazar Sailaukan, Yonghan Yang, Feilong Chen, et al. Bioxarena: Benchmarking llm agents on multi-modal biomedical machine learning tasks. *arXiv preprint arXiv:2605.15766*, 2026.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

David Mendez, Anna Gaulton, A Patrícia Bento, Jon Chambers, Marleen De Veij, Eloy Félix, María Paula Magariños, Juan F Mosquera, Prudence Mutowo, Michał Nowotka, et al. Chembl: towards direct deposition of bioassay data. *Nucleic acids research*, 47(D1):D930–D940, 2019.

Henry E Miller, Matthew Greenig, Benjamin Tenmann, and Bo Wang. Bioml-bench: Evaluation of ai agents for end-to-end biomedical ml. *bioRxiv*, pp. 2025–09, 2025.

Deepak Nathani, Lovish Madaan, Nicholas Roberts, Nikolay Bashlykov, Ajay Menon, Vincent Moens, Mikhail Plekhanov, Amar Budhiraja, Despoina Magka, Vladislav Vorotilov, Gaurav Chaurasia, Dieuwke Hupkes, Ricardo Silveira Cabral, Tatiana Shavrina, Jakob Nicolaus Foerster, Yoram Bachrach, William Yang Wang, and Roberta Raileanu. MLGym: A new framework and benchmark for advancing AI research agents. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=ryTr83DxRq.

OpenAI. Introducing GPT-5.4, March 5 2026a. URL https://openai.com/index/introducing-gpt-5-4/.

OpenAI. Previewing GPT-5.6 Sol: a next-generation model, June 26 2026b. URL https://openai.com/index/previewing-gpt-5-6-sol/.

Yusuf Roohani, Andrew Lee, Qian Huang, Jian Vora, Zachary Steinhart, Kexin Huang, Alexander Marson, Percy Liang, and Jure Leskovec. Biodiscoveryagent: An ai agent for designing genetic perturbation experiments. In *International Conference on Learning Representations*, volume 2025, pp. 26417–26466, 2025.

Sanjay R Srivatsan, José L McFaline-Figueroa, Vijay Ramani, Lauren Saunders, Junyue Cao, Jonathan Packer, Hannah A Pliner, Dana L Jackson, Riza M Daza, Lena Christiansen, et al. Massively multiplex chemical transcriptomics at single-cell resolution. *Science*, 367(6473):45–51, 2020.

Kyle Swanson, Wesley Wu, Nash L Bulaong, John E Pak, and James Zou. The virtual lab of ai agents designs new sars-cov-2 nanobodies. *Nature*, 646(8085):716–723, 2025.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

The Gemini Team. Gemini 3.1 Pro: A Smarter Model for Your Most Complex Tasks, February 19 2026. URL https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-pro/.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Z.AI. GLM-5.1, April 7 2026. URL https://docs.z.ai/guides/llm/glm-5.1.

Xinyu Zhu, Yuzhu Cai, Zexi Liu, Bingyang Zheng, Cheng Wang, Rui Ye, Yuzhi Zhang, Linfeng Zhang, Siheng Chen, Yanfeng Wang, et al. Toward ultra-long-horizon agentic science: Cognitive accumulation for machine learning engineering. *arXiv preprint arXiv:2601.10402*, 2026.

Table 6: Comparison with the penalized score presented on BioXArena's leaderboard across all 76 tasks. **Avg.** is task-weighted across all tasks. Baseline scores are reported by BioXArena (Li et al., 2026); BioDyad scores use the same evaluation protocol. Bold marks the highest score in each column. Backends differ across systems (Appdx. E.2).

Agent	Seq	SC	Str	Net	Chem	Pert	Phen	Img	Text	Avg.
BioDyad (GPT-5.6-sol)	**0.776**	**0.861**	0.815	**0.521**	0.926	**0.776**	0.594	0.666	**0.671**	**0.738**
BioDyad (GLM-5.1)	**0.776**	0.771	0.729	0.509	**0.932**	0.737	**0.600**	**0.700**	0.552	0.705
BioDyad (DeepSeek-V4)	0.717	0.791	0.789	0.510	0.909	0.717	**0.600**	0.646	0.608	0.701
MLEvolve (Du et al., 2026)	0.765	0.798	0.750	0.510	0.869	0.601	0.502	0.606	0.534	0.666
STELLA (Jin et al., 2025)	0.677	0.559	0.481	0.497	0.878	0.555	0.574	0.651	0.641	0.613
Biomni (Huang et al., 2025)	0.526	0.646	0.569	0.497	0.723	0.628	0.547	0.659	0.415	0.579
ML-Master 2.0 (Zhu et al., 2026)	0.582	0.773	0.337	0.509	0.904	0.401	0.411	0.601	0.342	0.547
GPT-5.4 (OpenAI, 2026a)	0.748	0.673	**0.836**	0.507	0.866	0.391	0.501	0.601	0.565	0.636
GLM-5.1 (Z.AI, 2026)	0.649	0.753	0.824	0.511	0.894	0.336	0.487	0.513	0.393	0.601
Claude Opus 4.6 (Anthropic, 2026a)	0.665	0.496	0.726	0.509	0.605	0.416	0.507	0.656	0.565	0.572
Gemini 3.1 Pro (The Gemini Team, 2026)	0.596	0.681	0.738	0.510	0.827	0.400	0.490	0.635	0.607	0.611
Gemma 4 31B (Team et al., 2026)	0.568	0.683	0.730	0.430	0.746	0.214	0.411	0.490	0.293	0.513
Qwen3.6-Plus (Alibaba Cloud, 2026)	0.606	0.764	0.666	0.445	0.844	0.363	0.528	0.622	0.436	0.591
DeepSeek-V3.2 (Liu et al., 2025)	0.256	0.521	0.399	0.434	0.365	0.144	0.287	0.507	0.262	0.355

A Comparison with the BioXArena Leaderboard

Table 6 places BioDyad alongside the agent and BioXArena harness results presented on BioXArena's leaderboard (Li et al., 2026), retaining each baseline's reported backend configuration. All three BioDyad configurations exceed the highest reported overall score of 0.666 among the listed baselines. At least one BioDyad configuration achieves the highest score in eight of the nine domains. Because backend configurations differ across these entries, the matched-backend comparison in Section 5.2 isolates the framework's contribution.

B Implementation Details

B.1 Biomedical Resources from Biomni

Biomni (Huang et al., 2025) provides a biomedical environment comprising domain-specific tools, curated data resources, and scientific software.[1] Its tools support operations such as sequence analysis, molecular analysis, and cell-type annotation; its data resources include expression profiles, functional annotations, and disease associations; and its software environment provides executable analysis routines. These resources span heterogeneous biomedical workflows, so their relevance depends on the task and the candidate being developed.

In BioDyad, Biomni resources support prior guidance and candidate-level discovery. The discovery agent uses resource descriptions and data schemas to identify relevant evidence, inspect compatibility with task inputs, and propose concrete representation or analysis changes. For example, drug mechanism-of-action annotations can supply transferable information after drug names are matched to resource entries. The configured workspace exposes data-lake schemas and tool/software catalogs for this purpose. Resource use is selective: when no suitable resource can be matched to the task, the candidate can retain its existing representation. Proposed changes subsequently pass through staged execution and validation, with their outcomes recorded for later search.

Staged validation then determines which integrations improve prediction, as illustrated by the contrasting cases in Appendix E.3.

B.2 Discovery, Memory, and Execution Settings

The settings below complement the scientific and engineering hierarchies in the main text. Resource catalogs and schemas are described in Appendix B.1.

[1] https://biomni.stanford.edu/

Component	Implementation setting
Prior guidance	The Biomni A1-based planner runs once before search, with at most six read-only probe blocks and no candidate training.
Memory retrieval	Keyword and embedding rankings are combined through reciprocal-rank fusion. Each retrieval supplies up to two related successful or pending plans, two related failures, and one dissimilar successful plan; duplicates are removed.
Smoke execution	Static checks precede a reduced-cost smoke run that preserves the full program's feature representation, model, and output format. Passing requires successful execution, a validation score, and a submission artifact.
Finalization	The selected program retains its pre-retraining score for model selection. Its full-data variant is smoke-gated and executed once; if this fails, the original valid program and submission are retained.

B.3 MCGS Backbone and Search Configuration

Search representation and loop. Monte Carlo graph search (MCGS) organizes candidate programs as nodes and iteratively selects, expands, and evaluates them. Following MLEvolve (Du et al., 2026), primary edges record parent-child generation and carry reward updates back through the selected node's ancestors. Cross-branch references provide additional context while preserving this primary ancestry. Selecting the root produces a new draft; selecting an invalid candidate triggers debugging; selecting a valid candidate triggers improvement. In BioDyad, the scientific hierarchy supplies the biomedical plan and retrieved memory for each expansion, and the engineering hierarchy evaluates the resulting program. Validity and validation improvement determine its reward, which updates the search graph; its plan and outcome also update biomedical memory.

Balancing exploration and refinement. The inherited selection strategy combines upper confidence bounds applied to trees (UCT) with elite-guided sampling. UCT balances observed reward against exploration of less-visited nodes; elite-guided sampling prioritizes high-scoring valid candidates. For a visited node v, UCT uses

$$u_t(v) = \frac{Q_v}{N_v} + C_t \sqrt{\frac{\log(N_{\mathrm{pa}(v)} + 1)}{N_v}}, \tag{7}$$

where Q_v is cumulative reward, N_v is visit count, $\mathrm{pa}(v)$ is the primary parent, and C_t controls exploration. Unvisited nodes receive priority. Elite-guided selection samples the top-K valid candidates with probability proportional to inverse score rank.

Budget-dependent configuration. Selection increasingly favors elite candidates. Let ρ_t denote the elapsed wall-clock budget fraction. UCT is chosen with probability

$$w_{\exp}(\rho_t) = \begin{cases} 1, & \rho_t < \rho_s, \\ 1 - (1 - w_{\min}) \dfrac{\rho_t - \rho_s}{\rho_e - \rho_s}, & \rho_s \le \rho_t < \rho_e, \\ w_{\min}, & \rho_t \ge \rho_e, \end{cases} \tag{8}$$

with $\rho_s = 0.5$, $\rho_e = 0.7$, and $w_{\min} = 0.2$; otherwise, selection uses the elite strategy. The elite pool contains up to five candidates before ρ_e and three afterward, with at most three and two candidates from each branch, respectively. These search rules follow the MLEvolve backbone; BioDyad connects them to biomedical discovery, memory, and staged execution.

C Ablation Studies and Search Diagnostics

Table 4 reports aggregate predictive performance. This appendix examines the same component ablations at the candidate level, asking how each component changes the reliability of the programs that search actually executes. The runs use GPT-5.4, the 16 chemical-biology and phenotype-disease tasks, and a two-hour task budget. Counts are pooled across tasks within each configuration.

C.1 Diagnostic Setup and Counting Rules

Full-execution runs are candidates that start train/validation evaluation. Without staged evaluation, every generated candidate enters this stage directly; with staged evaluation, only candidates that pass static checks and smoke execution do. Each full-execution run receives one of two terminal labels. *Valid* runs complete with a finite validation score, no buggy flag, and the required submission artifact. *Invalid* runs end with an execution defect, an unparseable score or output record, or no terminal record, typically due to timeout or interrupted bookkeeping. Percentages use each configuration's number of full-execution runs as the denominator. These candidate-level outcomes differ from the task-level success rate in the main results.

C.2 Candidate Execution Outcomes

Table 7 reports execution outcomes for all six configurations. Because the configurations differ by one component at a time, the table contains matched pairs for each mechanism: two pairs that add biomedical discovery, two that add biomedical memory, and three that add staged evaluation.

Table 7: Candidate outcomes in the component ablations. Each entry gives a count and its percentage of that configuration's full-execution runs.

Hierarchy	Mechanism	MLEvolve	MLEvolve w/ discovery	BioDyad w/o memory	BioDyad w/o Sci.	BioDyad w/o Eng.	BioDyad (full)
Sci. hierarchy	Bio. discovery	✗	✓	✓	✗	✓	✓
	Bio. memory	✗	✗	✗	✗	✓	✓
Eng. hierarchy	Staged evaluation	✗	✗	✓	✓	✗	✓
Outcomes	Full runs	552	672	639	446	668	566
	Valid	304 (55.1%)	314 (46.7%)	425 (66.5%)	319 (71.5%)	411 (61.5%)	458 (80.9%)
	Invalid	248 (44.9%)	358 (53.3%)	214 (33.5%)	127 (28.5%)	257 (38.5%)	108 (19.1%)

The complete framework has both the highest valid share (80.9%) and the largest number of valid runs (458) among the six configurations, together with the fewest invalid runs (108).

Staged evaluation. Adding staged evaluation increases the valid share in all three matched pairs: from 55.1% to 71.5% without the scientific hierarchy (MLEvolve → BioDyad w/o Sci.), from 46.7% to 66.5% with discovery alone (MLEvolve w/ discovery → BioDyad w/o memory), and from 61.5% to 80.9% with discovery and memory (BioDyad w/o Eng. → BioDyad full). Absolute counts show where this gain comes from. In each pair, staged evaluation reduces the number of full executions (552 → 446, 672 → 639, and 668 → 566) and reduces invalid runs by 40–58% (248 → 127, 358 → 214, and 257 → 108), while the number of valid runs increases (304 → 319, 314 → 425, and 411 → 458). The engineering hierarchy therefore shifts full-execution budget away from infeasible candidates without reducing the number of valid programs available for comparison.

C.3 Scientific Hierarchy and Execution Reliability

Biomedical discovery. Adding discovery without memory lowers the valid share in both matched pairs: from 55.1% to 46.7% without staged evaluation (MLEvolve → MLEvolve w/ discovery), and from 71.5% to 66.5% with it (BioDyad w/o Sci. → BioDyad w/o memory). Without staged evaluation, invalid runs increase from 248 to 358. This is consistent with the coordination problem stated in the introduction: integrating biomedical evidence into programs introduces execution errors and additional validation demands, and discovery alone does not resolve them.

Biomedical memory. Adding memory to discovery raises the valid share by a similar margin in both matched pairs: from 46.7% to 61.5% without staged evaluation (MLEvolve w/ discovery → BioDyad w/o Eng.), and from 66.5% to 80.9% with it (BioDyad w/o memory → BioDyad full). In both pairs, memory brings the valid share above that of the corresponding configuration without the scientific hierarchy (55.1% and 71.5%, respectively). Memory links each biomedical plan to its execution outcome, so later discovery can revise integrations that failed rather than propose them again. Memory thus turns discovery from a net cost in execution reliability into a net gain.

Relation to predictive performance. Execution reliability alone does not determine task-level performance. BioDyad w/o Sci. has the second-highest valid share (71.5%) but the lowest average score in Table 4 (0.665), whereas BioDyad w/o memory and w/o Eng. reach higher scores (0.736 and 0.733) with lower valid shares. Staged evaluation makes candidate programs executable, but predictive gains additionally require the biomedical content supplied by the scientific hierarchy. Only the complete framework is best on both measures, combining the highest valid share with the highest average score (0.763). These diagnostics complement the task-level ablations: biomedical discovery supplies predictive content, memory and staged evaluation absorb the execution cost of integrating it, and the benefit appears when discovery and ML engineering advance together.

D Budget-Dependent Score Analysis

Section 5.5 summarizes budget effects by domain and illustrates four tasks in Fig. 5. Here we provide normalized test-score trajectories and checkpoints for all 16 chemical-biology and phenotype-disease tasks using GPT-5.6-sol, with one 12-hour run per task. All analyses use the same fixed set of runs, documented in the accompanying source-data manifest.

D.1 Trajectory Reconstruction

For each task, we retain candidates with a finite normalized test score and a recorded completion time within the 12-hour budget. Elapsed time is measured from the run-start timestamp to the logged candidate completion. The cumulative best at a checkpoint is the highest normalized test score among candidates completed by that time. A score is carried forward until another completed candidate improves it; the last observed best is retained through 12 hours. Pearson correlations are mapped from $[-1, 1]$ to $[0, 1]$ using $(r + 1)/2$; metrics already in $[0, 1]$ retain their recorded values.

Figure 6 shows all 16 trajectories. These retrospective test-score maxima characterize the evaluated candidate pool, and all checkpoints for a task belong to the same run.

D.2 Task-Level Gains and Their Timing

Table 8 separates gains between 2 and 4 hours from gains over the remaining eight hours. All checkpoint values and gains are computed from the same candidate records as the curves, before rounding. This task-level view exposes timing differences behind the domain summaries in Table 5.

Table 8: Normalized test-score checkpoints and gains in the 12-hour traces. CB denotes chemical biology and PD denotes phenotype-disease. Each checkpoint is a retrospective maximum over completed candidates. Gains are absolute differences computed before rounding.

Domain	Task	Test 2 h	Test 4 h	Test 12 h	Δ 2-4 h	Δ 4-12 h
CB	BACE1 Binding Affinity	0.934	0.936	0.939	+0.002	+0.003
CB	Cell Painting Perturbation	0.982	1.000	1.000	+0.018	+0.000
CB	CYP Inhibition (Multi-label)	0.940	0.941	0.944	+0.001	+0.003
CB	EGFR Binding Affinity	0.925	0.927	0.927	+0.003	+0.000
CB	GPCR Binding (Multi-class)	0.948	0.953	0.970	+0.005	+0.017
CB	hERG Binding Affinity	0.898	0.908	0.909	+0.010	+0.001
CB	Kinase Selectivity (Multi-label)	0.921	0.930	0.930	+0.008	+0.001
CB	Tox21 SR-ARE	0.865	0.865	0.875	+0.000	+0.010
PD	Alzheimer's Disease Staging	0.617	0.746	0.776	+0.130	+0.030
PD	Autism Diagnosis	0.660	0.660	0.660	+0.000	+0.000
PD	Breast Cancer Subtype	0.741	0.743	0.746	+0.002	+0.003
PD	COVID-19 Severity Classification	0.407	0.433	0.603	+0.026	+0.170
PD	Diabetes Readmission	0.469	0.471	0.474	+0.002	+0.003
PD	Genotype-to-Phenotype	0.662	0.672	0.673	+0.010	+0.001
PD	Pan-Cancer Survival Prediction	0.789	0.794	0.799	+0.005	+0.005
PD	Spatial Immune Infiltration	0.779	0.783	0.789	+0.004	+0.006

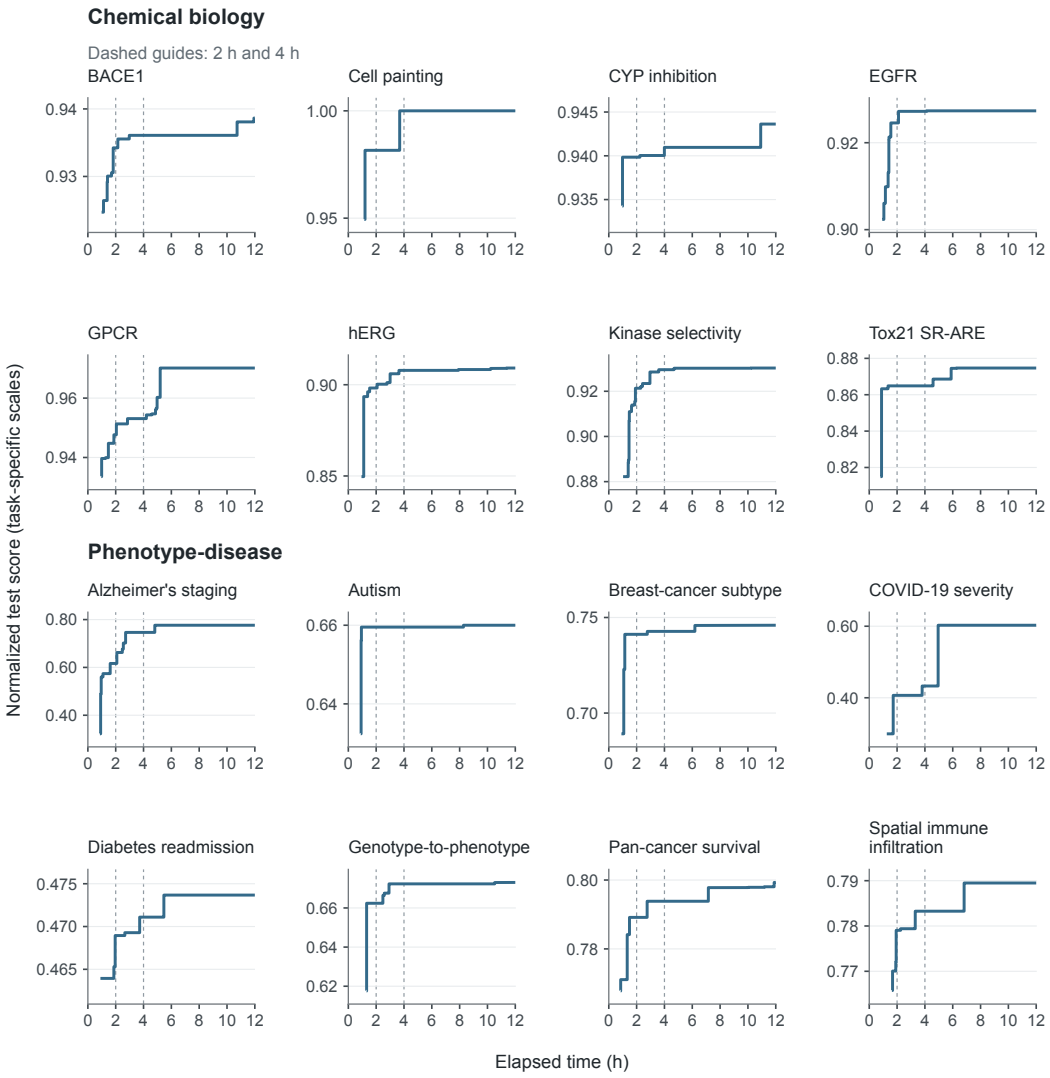


Figure 6: Cumulative best normalized test scores for all 16 tasks, using the same runs as Table 8. The upper two rows show chemical-biology tasks and the lower two show phenotype-disease tasks. Each task has its own vertical scale. Steps occur at logged candidate completion; dashed guides mark 2 and 4 hours. No score is shown before the first completed candidate, and the last observed best is carried forward to 12 hours.

The timing of improvement differs across tasks. Cell painting and Alzheimer's staging gain mainly between 2 and 4 hours, whereas COVID-19 severity and GPCR gain mainly after 4 hours. The genotype-to-phenotype run also gains mainly between 2 and 4 hours, with a smaller subsequent increase. Autism's gain after 2 hours is below 0.001. The full task set therefore exposes timing and magnitude differences that a domain average or a small set of examples can obscure.

Table 9: Fixed repeated-run subset: the first two tasks per domain in BioXArena Table A4, preserving its within-domain order.

Domain	First task	Second task
Sequence	`gene-tissue-expression`	`isoform-expression`
Single-cell	`batch-integration`	`cell-type-from-expression`
Structure	`complex-structure-evaluation`	`enzyme-commission-prediction`
Network biology	`gene-disease-association`	`go-function-multi-label`
Chemical biology	`bace1-binding-affinity`	`cell-painting-perturbation`
Perturbation dynamics	`cancer-drug-sensitivity`	`crispr-perturbation-prediction`
Phenotype-disease	`alzheimers-disease-staging`	`autism-diagnosis`
Imaging	`amos-organ-segmentation`	`drug-moa-prediction`
Text-integrated	`biomedical-figure-vqa`	`dna-enzyme-function`

Table 10: Per-run results on the fixed 18-task subset with GPT-5.6-sol. Score is the penalized mean over all 18 tasks; success rate is the successful task count divided by 18. Summary entries report mean ± sample standard deviation (SD) across three independent runs.

Method	Metric	Run 1	Run 2	Run 3	Mean ± SD
BioXArena Harness	Score	0.695	0.705	0.699	$0.699_{\pm 0.005}$
	Success rate	0.944	0.944	0.944	$0.944_{\pm 0.000}$
Biomni	Score	0.637	0.660	0.644	$0.647_{\pm 0.012}$
	Success rate	0.889	0.889	0.889	$0.889_{\pm 0.000}$
MLEvolve	Score	0.709	0.683	0.672	$0.688_{\pm 0.019}$
	Success rate	0.944	0.889	0.889	$0.907_{\pm 0.032}$
BioDyad	Score	0.712	0.708	0.712	$0.711_{\pm 0.002}$
	Success rate	0.944	0.944	0.944	$0.944_{\pm 0.000}$

E Additional Evaluation Protocols

E.1 Repeated-Run Task Selection and Aggregation

The repeated-run study fixes the first two tasks in each of the nine domains in Table A4 of BioXArena (Li et al., 2026). Table 9 lists the exact 18 task identifiers. This deterministic, domain-balanced rule fixes the subset independently of method performance. All four methods use the same list, GPT-5.6-sol, task inputs, evaluator, hardware allocation, and two-hour limit in each of three independent runs. Mutable search state, generated artifacts, and method-specific memory are reset between task-runs; configured static resources remain fixed.

For method a and repetition $r \in \{1, 2, 3\}$, we aggregate all 18 tasks:

$$\bar{s}_a^{(r)} = \frac{1}{18} \sum_{i=1}^{18} s_{a,i}^{(r)}, \qquad R_a^{(r)} = \frac{1}{18} \sum_{i=1}^{18} c_{a,i}^{(r)}. \tag{9}$$

Here, $c_{a,i}^{(r)} = 1$ if the task program executes successfully and produces a valid submission, and 0 otherwise. All 18 tasks remain in the denominator. Table 3 reports the mean and sample SD of each metric across the three independent runs. These summaries are calculated before rounding the run-level values reported in Table 10.

E.2 Backend Configurations of Reported Systems

Table 11 records the framework-backend assignments for the reported-system reference in Section A. These are the original BioXArena configurations, distinct from the comparison in Sec. 5.2. For direct LLM baselines, the method name identifies the backend.

VCHarness (Cheng et al., 2026) is outside the two-hour quantitative comparison because its fine-tuning-centered node lifecycle targets multi-day execution; compressing that lifecycle would change the method being compared.

Table 11: Original backends of BioXArena's reported agent systems, retained for the overall reference experiment.

Method	LLM backend
Biomni	Claude Sonnet 4 (Anthropic, 2025)
STELLA	Dev/Tool-Creation: Claude Sonnet 4.6 (Anthropic, 2026b); Manager/Critic: Gemini 3.1 Pro (The Gemini Team, 2026)
MLEvolve	Gemini 3.1 Pro (The Gemini Team, 2026)
ML-Master 2.0	DeepSeek-V4-Pro (Xu et al., 2026)

E.3 Case Study Details

The cases in Section 5.4 were reconstructed from candidate programs, execution outputs, biomedical plans, and memory records. Scores are internal validation metrics, distinct from the normalized hidden-test scores in the benchmark comparison. The first two cases illustrate predictive gain and successful execution repair; the third illustrates feature integration that consumes budget without improving prediction.

E.3.1 Case (a): Successful Case-Predictive Gain

Task and rationale. The *drug-transcriptional-response* task (Srivatsan et al., 2020) predicts the expression change of 5,000 genes for each drug, cell line, and dose condition. Each drug is identified only by name, together with its dose, cell line, cell count, and a control indicator. The 104 training drugs and 84 test drugs do not overlap, so drug identity cannot transfer from training to test. With GLM-5.1, the prior guidance of BioDyad noted that generalization to unseen drugs therefore requires drug properties. It found MoA and target annotations in the Drug Repurposing Hub (Corsello et al., 2017) for 70 of the 104 training drugs and 58 of the 84 test drugs, with 40 MoA classes shared across the two sets. The rationale was that drugs sharing a mechanism may induce similar transcriptional programs.

What changed. The program matched drug names to Hub entries and encoded 1,433 MoA one-hot features and 2,183 target multi-hot features. It concatenated these blocks with Morgan fingerprints, cell-line features derived from DepMap, log dose, log cell count, a cell-line one-hot encoding, and the control indicator, and fitted a Ridge regression model. Prior guidance introduced the annotations from the first draft onward. The reference removes only the MoA/target block from the same program, retaining all other features, the Ridge model, and the drug-grouped validation, in which validation drugs are unseen during fitting.

Outcome and interpretation. Mean per-gene Pearson correlation increased from 0.1016 to 0.1146 (+0.0130, or 12.8% relative). The Hub provides pharmacological annotations rather than transcriptional measurements, so the gain reflects transfer of responses from training drugs with shared mechanisms, information that drug names alone do not provide.

E.3.2 Case (b): Successful Case-Smoke-Guided Repair

Task and rationale. The *kinase-selectivity-multi-label* task (Mendez et al., 2019) predicts a compound's activity across multiple kinase targets from its molecular representation. The candidate added Morgan count fingerprints and an Avalon fingerprint to existing molecular features. Count fingerprints encode substructure multiplicity, while Avalon provides a complementary structural encoding. The intended benefit was a richer representation of compounds for the per-kinase classifiers.

What changed and failed. With GLM-5.1, the generated program concatenated these feature blocks but miscalculated their total dimension: it allocated 9,058 columns for a 9,078-dimensional vector. Smoke execution encountered the mismatch during feature fusion and stopped before full execution. The error was recorded in biomedical memory and appeared with its traceback in the subsequent debug prompt. The repaired candidate changed only the allocation and two dimension assertions, retaining the feature computations, ExtraTrees classifiers, and hyperparameters.

Outcome and interpretation. The faulty candidate was rejected after 1.09 s of smoke execution, and full execution was skipped. After repair, the smoke stage took 150.8 s of wall-clock time, followed by 616.4 s of full execution. The candidate produced a submission and achieved macro ROC-AUC of 0.9210 under three-fold stratified validation. Early feedback thus supported a focused repair that preserved the proposed molecular representation.

E.3.3 CASE (C): FAILED CASE-NO PREDICTIVE GAIN

Task and rationale. The *cancer-drug-sensitivity* task (Iorio et al., 2016) predicts log half-maximal inhibitory concentration for drug-cell-line pairs. The reference program already used biomedical measurements summarizing dose-response curves, concentration bounds, and curve-fitting error, together with drug and cancer-related descriptors. BioDyad proposed six additional nonlinear features to make relationships among these measurements explicit, including transformations of curve area and combinations with concentration range and fitting error.

What changed. Using GPT-5.6-sol, the candidate expanded the input from 18 to 24 features. The program evaluated both representations within the same full execution, using the same histogram-based gradient-boosting regressor, hyperparameters, random seed, and three-fold splits grouped by cell line. Selection used Spearman correlation over pooled out-of-fold predictions.

Outcome and interpretation. The expanded representation scored 0.9555, compared with 0.9556 for the original, which was retained. The tree model likely already captured these nonlinear relationships from the original measurements. Excluding discovery and code generation, the trial required 35.1 s in the smoke stage and 22.8 s in full execution. The failed feature extension therefore consumed budget that could otherwise support further search, illustrating that adding biomedical features does not necessarily improve performance.