# PoggioAI/`MSc`: ML Theory Research with Humans *on* the Loop

A Technical Report

Mahmoud Abdelmoneum*,[1,2], Pierfrancesco Beneventano*,[1], Tomaso Poggio[1]

[1] *Massachusetts Institute of Technology*
[2] *Perseus Labs*

March 22, 2026

**Abstract**

We present **PoggioAI/`MSc`**[3], an open-source, customizable, modular multi-agent system for academic research workflows. Our goal is not autonomous scientific ideation, nor fully automated research. It is narrower and more practical: to reduce by orders of magnitude the human steering required to turn a specified hypothesis into a literature-grounded, mathematically established, experimentally supported, submission-oriented manuscript draft. **PoggioAI/`MSc`** is built with a current emphasis on machine learning theory and adjacent quantitative fields.

## 1 Introduction

LLMs first, agentic systems now, are changing how research is conducted, including in academia. Reports of AI systems making progress on difficult mathematical problems [1–26] naturally force us—*mathematicians* and *computer scientists*—to ask unpleasant and controversial questions:

*How much of academic research can, in fact, be automated?*

*To what extent, and how soon, might some of our skills be displaced by such systems? Which ones?*

The *scientific* goal of this effort is to understand to what extent agentic systems can automate ***high-quality*** research. The long-term *engineering* ambition of this line of work is to identify, approach, and if possible attain this upper bound of automation compatible with high-quality research.

As of February 2026, we believe current agentic systems can be impressive, but the quality of the ideas and artifacts they produce still falls short of serious academic standards and for high-impact research. Our focus, thus, is developing the best possible system for what we see as the **key necessary step toward that goal.**

*Equal contribution. Correspondence: `pierb@mit.edu` or `Discord`. Please check our website for updates. Code available at `github.com/PoggioAI/PoggioAI_MSc`.

[3] Which stands for **P**roof-**O**riented **G**enerative **G**eneral **I**ntelligence **O**rchestration: **A**gentic **I**nvestigators / **M**ulti-agent **S**cientists **C**ollaboration.

## Practical Objective

A natural dream for many researchers is to have a junior collaborator who, given the core idea and limited guidance, can turn that idea into something as close as possible to a submission-ready article. In our experience, moving from a strong human-developed idea to a solid machine-learning-theory manuscript still often takes on the order of $10^2$ to $10^3$ prompts—if possible at all—with frontier reasoning models and agentic systems as Claude Code or GPT Pro. Perhaps some of this is a prompting problem on our side. But to the best of our knowledge, current AI systems are not yet enough to satisfy this naive dream. The goal here is:

*Can we design an AI system that, in at most 10 human steers,*

*pushes a strong **Hypothesis** to a **Written Article** of serious academic quality?*

More precisely, can we build an agentic system that requires no more than 10 explicit interventions from the human researcher to develop an idea rigorously, establish it at the highest academic standards, and produce a strong manuscript draft? Equivalently, one can view the problem as one of compression: can we reduce the human control budget from roughly $\sim 10^5$ tokens of interactive instruction to something closer to $\sim 10^3$, without losing rigor and quality?

Importantly, this project is *not*[1] about fully automating the path from idea to article, *nor* is it about having AI systems originate the research idea itself[2]:

*The key constraint is **high quality**, the objective function*

*to minimize is the **human steering burden** (or oversight) required to reach that.*

## Researching Ethically: AI Systems Leverage

A growing literature argues that machine learning scholarship often states central claims more strongly than they are mathematically, causally, or empirically established [27–31]. Direct critiques point to explanation–speculation slippage, "mathiness," and unclear attribution of empirical gains; reproducibility reforms explicitly identify over-claiming and hypothesis–claim mismatch as part of the field's standards problem; and domain-specific audits have found systematic overoptimism driven by weak baselines, reporting bias, shortcut learning, and fragile benchmark design [27, 32–36]. In causal settings, additional work warns that predictive performance is frequently insufficient to justify intervention or fairness claims [37–40].

Producing only rigorously established work is not a methodological preference—it is an *ethics of research.* If researchers will increasingly rely on AI systems, then the highest-leverage place to enforce that ethic is *inside those systems*: bias the tools toward establishment, and establishment scales with adoption.

**PoggioAI**/`MSc` is our attempt to do exactly this. System prompts and control flows are designed to make novelty claims explicit, stress-test them against competing explanations, align mathematics with experiments, and demand intervention experiments when causal language is involved. A field with more papers is not bad per se—provided those papers push the knowledge boundary. We believe optimizing for quality the AI tools that researchers use is the intervention most likely to move the needle.

## Contributions

- We present **PoggioAI**/`MSc`, an open-source, customizable, modular multi-agent system for hypothesis-to-paper workflows with explicit human oversight.

[1] Yet.
[2] Yet.

- We introduce an artifact contract in which progress is represented by named intermediate outputs and stage-level validation gates rather than by free-form dialogue alone.
- We implement a fixed workflow that decomposes work into 23 specialized agents spanning literature review, theory, experimentation, synthesis, and editorial stages, with repair and follow-up routes when artifacts are incomplete, inconsistent, not novel, or not adequately established.
- We integrate optional rigor-enhancing components—including theorem-oriented reasoning, multi-model debate counsel, and tree-search-based exploration—within the same execution framework.
- We provide a technical report on the scope, architecture, and operational limits of the current release, with an explicit separation between structural success and scientific validity.

## Acknowledgements

We are extremely thankful to Perseus Labs who meaningfully helped both economically and in terms of software development, a particular thank you to Hilal Hussain. We also want to thank deeply Riccardo Neumarker, Emanuele Rimoldi, Yulu Gan, Qianli Liao, Marc Bacvanski, Liu Ziyin, and Theodoros Evgeniou for many fundamental discussions. *We welcome contributions and are seeking support for the API costs of further development.*

## Contents

## 2 Engineering Lessons and Design Requirements

This section has two goals. First, it makes explicit the main engineering difficulties we encountered while building **PoggioAI**/`MSc`, so that future work can attack them directly rather than rediscover them piecemeal. Second, it serves as guidance for users who may customize or extend the system: it documents which design choices were responses to concrete workflow failures, which problems we believe are only partially solved, and which interface contracts should be preserved if one wants the pipeline to remain reliable. We present these lessons in the same order in which they typically appear during a research-to-paper run.

### 2.1 Difficulties Encountered

**1. Ideation quality improved when planning became debate rather than monologue.** One of our earliest observations was that single-agent ideation tended to converge too quickly to generic or aesthetically plausible project plans. Quality improved substantially when the early planning phase was reformulated as a structured debate among multiple agents rather than a single agent trying to ideate, justify, and plan everything at once. This was also oserved, but implemented differently by Du et al. [41].

**2. Give agents explicit, competing objectives.** A second related observation was that debate alone was not enough: each agent needed an explicit goal. In practice, the quality of the resulting research plans increased when we assigned different roles and incentives to different ideation agents. In the current design, these roles include one agent pushing for timely and practitioner-relevant questions, one pushing for rigorous and potentially novel mathematical theory, and one pushing for the strongest narrative arc together with the right alignment or disalignment with existing folklore. This does not solve ideation in general, but it consistently produced better decompositions and more informative disagreements than a single planner.

**3. Theory and experiments should be coordinated, but not fully entangled.** A natural design instinct is to maximize context sharing between the mathematical and experimental tracks. In our experience, that improved local alignment but weakened both tracks. When theory and experiments shared too much context too early, they tended to collapse toward the same framing, inherit each other's blind spots, and produce intermediate outputs that were better matched but less strong. Better final papers came from a weaker form of coordination: both tracks share an explicit decomposition and common contribution claims, but do not continuously share unrestricted working context. Instead, they exchange information at controlled synchronization points, especially before synthesis and at the beginning of a new cycle. This design tries to preserve independence of reasoning while still forcing the two tracks to remain on a single paper trajectory.

**4. Long-horizon runs need explicit stopping criteria.** Earlier versions of the system often kept iterating long after the core artifacts were already sufficiently established. This led to wasted computation, wasted budget, and occasionally worse outputs due to late over-editing or scope drift. The underlying difficulty is that the relevant notion of diminishing returns is conceptual rather than merely token-based: a run can continue to generate text while adding little scientific value. Our current response is to expose explicit stage gates, bounded iteration, and steering checkpoints. When the system judges that additional loops are likely to bring only marginal conceptual gain, it is encouraged to stop, surface the current state, and request a human steer. We do not regard this problem as fully solved, but treating stopping as a first-class systems problem already improves both cost and reliability.

**5. Parallel debugging remains expensive and only partially solved.** Debugging a long, partially parallel research workflow is much slower than debugging a single-turn assistant. Failures may arise from malformed artifacts, routing errors, stale intermediate state, synchronization bugs between tracks, or interactions between multiple agents that only become visible late in the run. When these failures occur after several hours of execution, the cost of diagnosis can be substantial. We do not currently know how to make this problem easy. What we do instead is make failure state external and inspectable: the system persists checkpoints, intermediate messages, budget traces, and mode-specific workspaces so that a run can be resumed, audited, or post-mortemed rather than restarted from scratch. This does not remove the debugging burden, but it makes it more manageable and far less opaque.

## 2.2 Implications on Workflow-Level Design Requirements

Taken together, the observations above led us to treat workflow design itself as a first-class systems problem. In our experience, long research runs become unreliable when intermediate state is implicit, when theory and experiments evolve without a shared contract, or when apparently polished drafts are produced without the artifacts needed to inspect how those drafts were assembled. The current release therefore follows seven design requirements.

- **Reduce steering burden without hiding human responsibility.** The aim is to compress the amount of human orchestration needed to move from an initial research objective to a paper draft. At the same time, novelty, correctness, citation faithfulness, and publication decisions remain human responsibilities.
- **Keep theory and experiments on a single paper trajectory.** When both tracks are active, they must remain tied to the same contribution claims and follow-up logic, even when they are developed semi-independently.
- **Externalize intermediate state as named artifacts.** Progress should live in files and structured outputs rather than only in conversational context, so that runs are inspectable, resumable, and auditable.
- **Support bounded iteration rather than a single forward pass.** Real research workflows need loopbacks: feasibility checks may fail, experiments may invalidate theory, theory may require new experiments, and writing may reveal unresolved inconsistencies.
- **Separate structural validation from scientific truth.** The system can validate artifact existence, parseability, selected internal consistency properties, and review-gate thresholds. It cannot certify scientific correctness, novelty, or acceptance probability.
- **Make long runs resumable, steerable, and budget-aware.** Checkpointing, stage-based resume, live steering hooks, and budget traces are operational requirements rather than convenience features.
- **Expose multiple enforcement levels.** The same architecture should support lighter exploratory runs and stricter paper-oriented runs, with different validation thresholds and artifact expectations.

## 2.3 Artifact Contracts

A run is not treated as successful merely because it produces fluent prose. It is expected to produce a paper-centered workspace with explicit deliverables for discovery, planning, execution, synthesis, and editorial review. This artifact contract is the main interface between the workflow and the researcher. It serves both goals of the present section: it tells future researchers which interface stabilized the pipeline, and it warns customizers which artifacts are dangerous to remove if they want the system to remain inspectable, resumable, and debuggable.

| **Bundle** | **Concrete artifacts** | **Role** | **Gate / mode** |
|---|---|---|---|
| Paper workspace core | `final_paper.tex`, `paper_workspace/literature_review.pdf`, `research_plan.pdf`, `results_assessment.pdf`, `followup_decision.json`, `track_decomposition.json` | Minimal paper-facing workspace linking discovery, planning, synthesis, and manuscript generation | `-enforce-paper-artifacts` |
| Optional outputs | `final_paper.pdf`, `experiments_to_run_later.md` | Tighten success definition for PDF or deferred experiment planning | `-require-pdf`; `-require-experiment-plan` |
| Editorial strictness | `author_style_guide.md`, `intro_skeleton.tex`, `style_macros.tex`, `reader_contract.json`, `editorial_contract.md`, `theorem_map.json`, `revision_log.md`, `copyedit_report.tex`, `review_report.tex`, `review_verdict.json`, opt. `claim_traceability.json` | Manuscript governance and revision history for stricter runs | `-enforce-editorial-artifacts` |
| Operational bookkeeping | `checkpoints.db`, `run_token_usage.json`, `budget_state.json`, `budget_ledger.jsonl`, `inter_agent_messages/` | Resume, budget tracking, post-hoc inspection | Operational trace |
| Mode-specific artifacts | `experiment_workspace/*`, `experiment_runs/*`, `math_workspace/*`, `counsel_sandboxes/*`, `tree_search_state.json`, `tree_branches/*` | Evidence from empirical, mathematical, counsel, and tree-search modes | Checked by downstream stages |

Table 1: Artifact bundles in the current **PoggioAI**/`MSc` release. The first two rows define the minimal paper-facing contract; remaining rows extend it by run mode.

In stricter paper-oriented modes, validation does more than check file existence. The final gate can also enforce internal review thresholds, paper-quality checks, and, when theorem-oriented or editorial modes are active, additional consistency requirements on theorem dependencies, acceptance status, and claim traceability. These checks are useful because they prevent the system from calling an obviously incomplete run a success. They should not, however, be confused with guarantees of scientific correctness.

A further design choice is that proofreading and reviewer stages emit editable source artifacts such as `copyedit_report.tex` and `review_report.tex`. Validation is keyed to those source artifacts rather than only to rendered PDFs. This keeps the workflow aligned with a manuscript-development process rather than a screenshot-like end product.

In summary, the artifact contract serves three purposes. It makes the pipeline legible to a human researcher, it provides a stable external interface for resume and post-hoc debugging, and it lets this report state a narrow but defensible notion of success: **PoggioAI**/`MSc` can reliably produce and validate a structured research workspace, even though the scientific content of that workspace still requires human scrutiny.

These observations shaped every architectural decision described in the following sections.

## 3 Mechanisms for Quality and Rigor

Every multi-agent system has stages. What distinguishes **PoggioAI**/`MSc` is the *care* baked into pushing quality at each stage. This section describes six concrete mechanisms—each a direct response to the engineering lessons in Section 2—that together bias the system toward rigorous establishment rather than fluent generation.

## 3.1 Persona council: debate with purpose

Single-agent ideation tends to converge too quickly to generic or aesthetically plausible plans (Lesson 1 in Section 2). Our response is the `persona_council`: a structured debate among three agents with distinct, competing goals.

- **Practical Compass.** Pushes for timely, practitioner-relevant research questions. Its job is to keep the project grounded in problems that real researchers care about.
- **Rigor & Novelty.** Pushes for rigorous, potentially novel mathematical theory. Its job is to ensure that the formal content is strong enough to survive serious scrutiny.
- **Narrative Architect.** Pushes for the strongest narrative arc and the right alignment—or deliberate disalignment—with existing folklore. Its job is to make the paper tell a story that is both accurate and compelling.

The three personas do not simply vote. They debate through structured rounds, and synthesis rules force explicit conflict resolution: disagreements must be surfaced and adjudicated, not papered over by averaging. The result is not consensus for its own sake, but a research plan whose tradeoffs have been examined from multiple angles.

This mechanism directly addresses Lesson 2 from Section 2: giving each agent an explicit, competing objective consistently produced better decompositions and more informative disagreements than a single planner trying to ideate, justify, and plan everything at once.

## 3.2 Adversarial novelty falsification

The literature review agent's default stance is *skepticism*—it assumes the central claim is already known and tries to find the evidence. This is not "find related work." It is "try to kill the novelty claim."

The agent assigns each claim one of four statuses:

```
OPEN  |  PARTIAL  |  KNOWN  |  EQUIVALENT_KNOWN
```

Multi-source search—including deep-research APIs, semantic scholar queries, and citation traversal—is employed not to decorate the introduction with related work, but to build a genuine case for or against novelty. A claim rated `KNOWN` or `EQUIVALENT_KNOWN` triggers a rethink: the system must either reformulate the contribution or provide explicit evidence that the existing result does not subsume the new one.

This mechanism matters because one of the most common failure modes of automated research is confident production of work that is, in fact, already established. By making skepticism the default, the system forces early confrontation with the prior literature rather than late-stage embarrassment.

## 3.3 Theory–experiment independence

A natural design instinct is to maximize context sharing between the mathematical and experimental tracks. In our experience, this improves local alignment but weakens both tracks (Lesson 3 in Section 2).

The non-obvious design: share *less* context, get *better* papers. The theory and experiment tracks share an explicit decomposition (`track_decomposition.json`) and common contribution claims, but they do *not* continuously share unrestricted working context. Instead, they exchange information at controlled synchronization points—especially before synthesis and at the beginning of a new cycle.

Why does this work? When theory and experiments share too much context too early, they tend to collapse toward the same framing, inherit each other's blind spots, and produce intermediate outputs that are better matched but individually less strong. Independence preserves the possibility that the experiment track discovers something the theory track missed, or vice versa. The merge and duality stages then force reconciliation, but only after each track has had the chance to develop its own strongest argument.

## 3.4 Reviewer hard blockers

The internal reviewer agent does not produce a single smooth score. It evaluates five binary "hard blockers" (B1–B5):

- **B1** No explicit research questions stated in the introduction.
- **B2** No evidence pointers for key takeaways (claims without traceable support).
- **B3** Placeholders, stubs, or unfinished sections remain in the manuscript.
- **B4** Critical experimental results are missing or unreported.
- **B5** Formal claims lack proof sketches or verification traces.

If *any* hard blocker is true, the reviewer score is capped at 4 regardless of prose quality. A score of 8 or above is assigned only when the manuscript is genuinely approaching publication readiness. This shows that the system does not rubber-stamp its own output: a fluent but incomplete paper is flagged as incomplete, and the workflow routes back to revision.

Hard blockers are the mechanism through which the design requirement of separating structural validation from scientific truth (Section 2) becomes operationally concrete in the editorial phase.

## 3.5 Multi-model counsel as structured disagreement

Counsel is not redundancy for robustness. It is structured disagreement for quality.

When counsel is enabled, a specialist stage proceeds as follows. Three frontier models—in the default configuration, Claude Opus 4.6, GPT-5.4, and Gemini 3 Pro Preview—each work in an isolated sandbox copy of the workspace. They produce independent candidate outputs without seeing each other's work. The candidates are then exposed to multiple debate rounds in which each model critiques the others. Finally, a synthesis model (by default Claude Sonnet 4.6) integrates the strongest elements into a single authoritative output that is promoted back to the main workspace.

The goal is to surface alternative reasoning, not to vote. A majority-rules scheme would converge to the blandest common denominator. Instead, the synthesis model must explicitly address disagreements, choosing between alternatives with stated rationale. Engineering discipline is maintained through circuit breakers and quorum logic: if models fail, time out, or produce degenerate outputs, the system falls back gracefully rather than blocking the entire pipeline.

This design targets stages where disagreement is informative—literature review, experiment design, and manuscript writing—and it changes the failure mode from "one model silently committed to one line of reasoning" to "multiple models exposed and critiqued alternative drafts before promotion."

## 3.6 Tree search over proof strategies

Proof construction has genuine combinatorial structure, and a single linear pass through a proof attempt is often insufficient. When tree search is enabled, the theory track's prover stage is replaced by a DAG-layered best-first search controller.

The controller selects frontier claims from the claim graph, generates alternative proof strategies for each, and scores the resulting branches using a composite function:

| | |
|---|---|
| 40% | LLM-estimated promise of the strategy |
| 25% | Impact on the claim graph (how many downstream claims are unblocked) |
| 15% | Cost efficiency (expected tokens per unit of progress) |
| 10% | Depth penalty (prefer shallower proofs) |
| 10% | Sibling diversity (penalize strategies too similar to existing branches) |

The top-scoring candidates are executed in forked workspaces. Failed branches can spawn debugging children up to a configurable depth, while low-scoring branches are pruned. The corresponding artifacts—`tree_search_state.json` and `tree_branches/`—make the entire search process inspectable rather than hidden inside one monolithic prover call.

The scoring formula itself reflects the care behind the system: it is not a generic "try again" loop, but a deliberate balance between exploration breadth, graph-level impact, and computational discipline.

## 4 The System

This section gives a single, self-contained walkthrough of **PoggioAI**/`MSc`. The workflow contains 23 specialist agents and 30 total graph nodes; the additional nodes are routing gates and control points shown in Figure 1. We describe the pipeline phases, reference the artifact contract from Section 2, summarize the optional modules, and close with the operational model. Readers who want full implementation details should consult Appendix A.

### 4.1 Pipeline phases

At the core of the release is a fixed LangGraph workflow. "Fixed" refers to the execution topology: the graph always traverses the same high-level stages and routing logic, even though the generated content remains stochastic. The system is not deterministic in output, but it is explicit and repeatable in control flow. The graph is organized into six phases.

**Phase 1: Discovery and feasibility screening.** The run begins with a `persona_council` stage (Section 3.1), followed by `literature_review_agent`, a feasibility gate, `brainstorm_agent`, and `formalize_goals_agent`. This phase grounds the task in prior work, checks whether the direction appears feasible, and converts a high-level objective into a concrete technical plan.

**Phase 2: Track planning and routing.** Planning writes a decomposition into theory, experiment, or mixed tracks, recorded in `paper_workspace/track_decomposition.json`. The track router uses this decomposition to decide which execution branches to launch.

**Phase 3: Parallel technical execution.** When theory work is selected and theorem-oriented modules are enabled, the graph runs the theory path: mathematical literature review, proposal, proving, rigorous verification, empirical verification, and proof transcription. When empirical work is selected, it runs experiment literature review, design, execution, verification, and transcription. The two tracks can run independently or in parallel.

**Phase 4: Completion verification and results formalization.** Track outputs are merged and passed to `verify_completion`, which routes the run in three directions: proceed, iterate because the work is incomplete, or rethink the direction more fundamentally. On success, `formalize_results_agent` produces a structured assessment of what has been established.

**Phase 5: Internal consistency and follow-up.** A duality check can be applied after results formalization. Failure triggers follow-up literature work and renewed planning; success moves the run to paper production.

**Phase 6: Paper production and editorial quality assurance.** The pipeline prepares resources, writes the paper, proofreads it, runs an internal reviewer, and applies a final validation gate. Validation failure routes back to writeup for revision before a success state is reported.

## 4.2 Artifact contract

All six phases communicate through the artifact contract described in Section 2 and summarized in Table 1. A run is not treated as successful merely because it produces fluent prose; it must materialize the required named artifacts for discovery, planning, execution, synthesis, and editorial review. In stricter modes, the final gate additionally enforces internal review thresholds, theorem-dependency consistency, and claim traceability.

## 4.3 Optional modules

**Counsel.** When `-enable-counsel` is active, selected specialist stages become a multi-model protocol with sandboxing, debate rounds, synthesis, and artifact promotion. Three frontier models work in isolated sandboxes, critique each other's outputs, and a synthesis model promotes a single authoritative result. Details are in Section 3.5.

**Tree search.** When `-enable-tree-search` is active, the linear proof stage in the theory track is replaced by a DAG-layered best-first controller that explores multiple proof strategies in parallel. See Section 3.6.

**Mathematical reasoning pipeline.** When `-enable-math-agents` is active, the theory track gains a full claim graph, proof workspace, lemma library, and staged verification. The math workspace stores claims, proofs, and verification traces as structured artifacts rather than free-form text.

**Campaign orchestration.** The optional campaign layer uses heartbeat scripts, cron, or OpenClaw to repeatedly launch or resume runs, validate stage artifacts, distill stage memory, track budget, attempt autonomous repair, and notify the user. This extends the artifact-centric philosophy beyond a single run.

## 4.4 Operational model

**PoggioAI**/`MSc` is designed as a human-on-the-loop system. A *steer* is a deliberate human intervention—distinct from the graph's own automatic loopbacks—that changes the direction, constraints, or scope of a run. The current release exposes steering through: the initial task specification; a TCP control socket and HTTP API during execution; resume from a named stage; injection of context files through an `inputs/` directory; and campaign-level control for multi-stage pipelines.

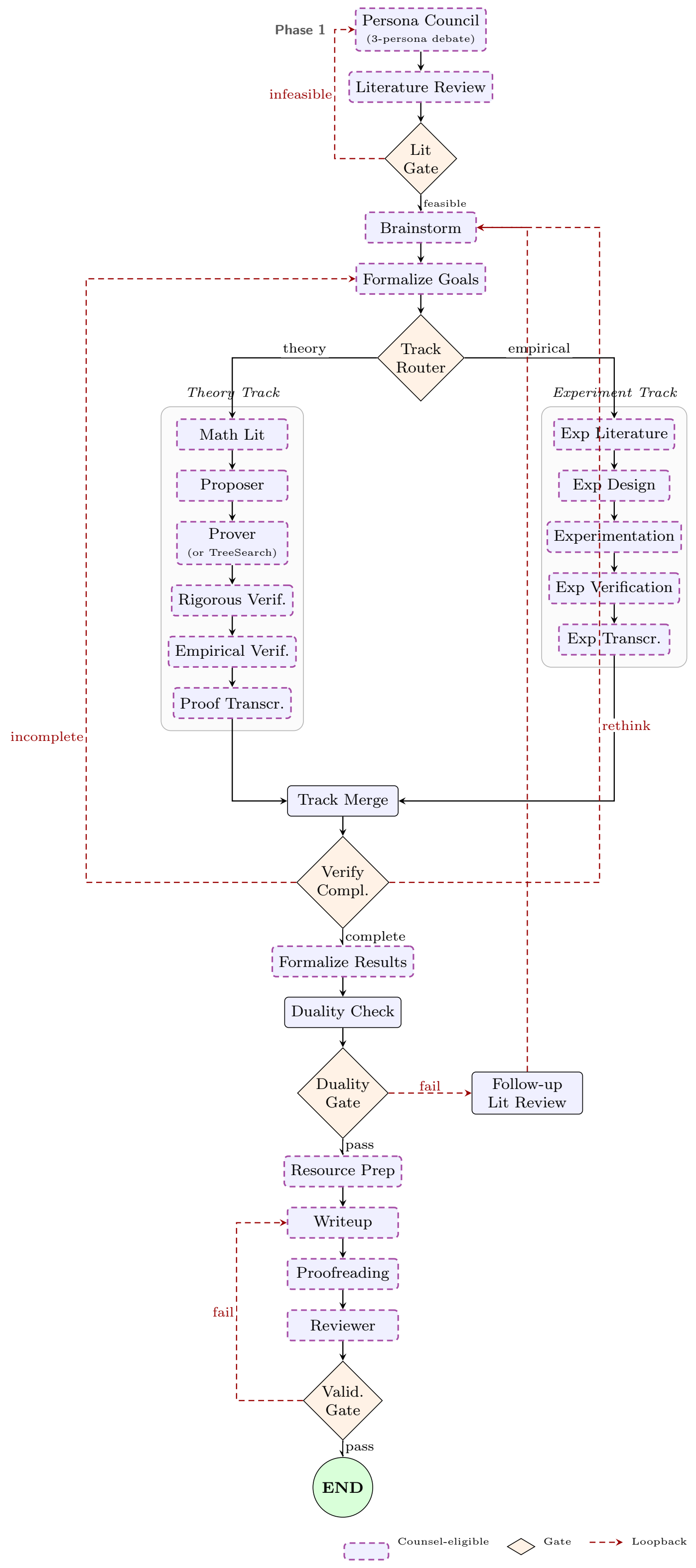

Figure 1: The **PoggioAI/**`MSc` execution graph. Dashed violet borders mark counsel-eligible agents. Red dashed arrows are loopbacks triggered by gate failures. Theory and experiment tracks run in parallel when both are selected.

| **Configuration** | **Typical cost** | **Typical runtime** | **Notes** |
|---|---|---|---|
| Quickstart (markdown, no counsel) | \$2–10 | 15–40 min | Single-model run, light paper draft |
| Base pipeline + LaTeX/PDF | \$10–40 | 30–90 min | Requires LaTeX toolchain |
| Base pipeline + math agents | \$20–60 | 60–150 min | Adds theorem-oriented stages |
| Counsel mode | \$50–200 | 2–5 hrs | Multi-model debate and synthesis |
| Tree search (no counsel) | \$60–180 | 2–4 hrs | Parallel proof exploration |
| Tree search + counsel | \$200–600 | 4–10 hrs | Highest-cost single-run mode |
| Full paper campaign | \$100–400 | 6–12 hrs | Multi-stage orchestration over several runs |

Table 2: Runtime and spend envelopes for representative **PoggioAI**/`MSc` configurations. Actual values vary with task scope, model choice, revision loops, and experiment intensity.

Every completed stage is persisted to SQLite in `checkpoints.db`, making long runs restartable and allowing downstream tools to reason about partially completed workspaces. Budget tracking is implemented through `budget_state.json`, `budget_ledger.jsonl`, and `run_token_usage.json`.

## 5 Related Work

The literature relevant to **PoggioAI**/`MSc` spans four lines. We position the system briefly here; a comprehensive survey is in Appendix B.

**Scholarly writing assistants.** Early LLM-era work focused on citation-grounded text generation rather than full research automation. Systems such as CiteBench [42], ChatCite [43], ScholarCopilot [44], SurveyGen [45], and editor-native tools like OverleafCopilot [46] and PaperDebugger [47] established that retrieval quality and claim–evidence linkage matter more than surface fluency. These are important precursors, but their primary artifact is a section or draft rather than a coordinated research program.

**Manuscript-first research-to-paper agents.** The closest predecessors are end-to-end systems targeting research-to-paper execution. Data-to-paper [48] emphasized provenance and backward traceability. The AI Scientist [49] and its successor [50] pushed toward fully autonomous ML research with tree search and automated review. Agent Laboratory [51] showed that human feedback at stage boundaries materially improves quality. CycleResearcher [52] coupled generation with review loops, AI-Researcher [53] introduced Scientist-Bench for open-ended evaluation, and AgentRxiv [54] enabled cross-run collaboration through a preprint-server abstraction. freephdlabor [55] argued for fully dynamic, user-customizable workflows. Relative to these systems, **PoggioAI**/`MSc` places more weight on a tighter artifact contract, theory–experiment coordination, and bounded workflow control.

**Autonomous discovery and open platforms.** Broader systems target scientific discovery rather than manuscript production. AI co-scientist [56] debates and refines hypotheses in biomedicine; InternAgent [57, 58] pursues long-horizon discovery across domains; AlphaEvolve [59] and Mathematical Exploration at Scale [60] target algorithmic and mathematical discovery; and DeepInnovator [61] trains models for innovative

| **The system can guarantee** | **The system cannot guarantee** |
|---|---|
| Deterministic execution topology: same stages, same routing logic, same loopback structure | Deterministic outputs: two runs with the same task may diverge in content |
| Artifact-complete workspace: all required files produced and structurally valid | Scientifically correct workspace: claims may contain errors, gaps, or undetected prior art |
| Explicit validation gates: hard blockers, review thresholds, editorial checks | Calibrated quality scores: internal reviewer scores are routing heuristics, not acceptance-probability estimates |
| Budget tracking and cost visibility via ledger files | Exact cost prediction: actual provider bills may differ from estimates |
| Resumable, checkpointed execution with stage-level restart | Idempotent restart: resumed runs may produce different outputs from fresh runs |
| Human steering surfaces at multiple points in the pipeline | Reduced need for human scientific judgment: novelty, correctness, and citation faithfulness remain human obligations |

Table 3: Guarantees and non-guarantees of the current **PoggioAI**/`MSc` release. The central distinction is between structural/operational properties (left) and scientific properties (right).

idea generation. On the platform side, ClawdLab and Beach.Science [62] introduce governance, auditability, and decentralized coordination for scientific-agent ecosystems. These systems enlarge the meaning of research automation, but many optimize for open-ended discovery or ecosystem flexibility rather than a governed paper-facing workflow.

**Positioning.** **PoggioAI**/`MSc` is closest in spirit to manuscript-first, end-to-end systems, but differs in emphasis. Relative to writing assistants, it covers more of the research lifecycle. Relative to autonomous discovery systems, it places more weight on explicit intermediate artifacts, theory–experiment coordination, and validation gates tied to manuscript production. Relative to dynamic or decentralized platforms, it favors a tightly governed research-to-paper path designed for human-steerable execution in machine learning theory and adjacent quantitative workflows. It should be read not as a general theory of autonomous science, but as an artifact-driven technical contribution to the research-to-paper segment of that broader landscape.

## 6 Limitations and Honest Assessment

### 6.1 What the system guarantees and what it does not

The most important distinction in this report is between *structural* and *scientific* success. The system's guarantees are structural and operational; its non-guarantees are scientific and epistemic. Table 3 makes this boundary explicit.

### 6.2 What is surprisingly automatable

One of the most interesting lessons is that *research structuring* is more automatable than one might expect. The system can turn a high-level task into a staged workspace with literature synthesis, a research plan, a theory–experiment decomposition, follow-up decisions, and a draft manuscript. Even when the final science requires human scrutiny, this intermediate structuring is already valuable.

A second surprisingly automatable layer is *paper-adjacent discipline*: proofreading, reviewer-style critique, copyediting traces, claim-traceability files, resource inventories, and revision logs. These are tedious but structurally well suited to an artifact-centric pipeline. Some of the "boring graduate student work" is indeed becoming tractable—not because the machine understands the science end to end, but because many parts of scientific production are really state-management and transformation problems.

### 6.3 What remains stubbornly human

The hardest parts of research remain strongly human-centered. Novelty judgments are open-world judgments. Strong baseline selection depends on field knowledge and taste. Mathematical correctness often hinges on subtle proof obligations that resist shallow verification. Experimental interpretation depends on whether the chosen measurements actually address the scientific question. Even manuscript quality has a human-dependent layer: deciding what the paper is *really about*, what should be emphasized, and what evidence is persuasive to a given community.

This is why the right control model for the present release is *human on the loop*, not *human out of the loop*. The system can propose, decompose, execute, and draft; the human remains responsible for owning the claims.

### 6.4 Toward stronger evaluation

The next milestone is not "more features" but a stronger evaluation program. A convincing benchmark would need repeated runs across multiple task families, explicit logging of human steering interventions, external citation-faithfulness audits, reruns of generated experiments, and expert assessments of theorem and manuscript quality. It should also include ablations isolating the value of counsel, tree search, strict editorial enforcement, and campaign-style repair.

The present release is best understood as an infrastructure milestone. It makes those evaluations possible by standardizing workflow state, artifact contracts, and operating modes. The stronger scientific claims should come only after that measurement program is in place.

## 7 Conclusion

**PoggioAI**/`MSc` is best described, in its present form, as a human-on-the-loop, artifact-centric research-to-paper pipeline. Its contribution is not that it solves scientific truth or automates novelty judgment, but that it turns a research task into a traceable workspace with planning artifacts, theory and experiment branches, revision loops, manuscript outputs, and explicit validation gates.

That contribution is already meaningful. It moves the problem from "how do I keep a long sequence of model calls coherent?" to "how do I inspect, guide, and validate a structured research workspace?" The guarantees, limitations, runtime modes, safety notes, and submission checklist all point to the same conclusion: the present system is a substantial step toward reducing human steering burden, but its outputs still require expert verification before any scientific claim is trusted or submitted.

The next phase should therefore focus on benchmarked evaluation. If future work can pair the current architecture with rigorous measurements of citation faithfulness, experiment reproducibility, theorem reliability, and human steering reduction, then **PoggioAI**/`MSc` will provide not only a useful release but also clearer scientific understanding of what parts of research automation are truly becoming tractable.

## References

[1] Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6. URL https://www.nature.com/articles/s41586-023-06924-6.

[2] Google DeepMind. Funsearch. GitHub repository, 2023. URL https://github.com/google-deepmind/funsearch. Repository accompanying the FunSearch Nature paper; accessed 2026-03-21.

[3] 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. *arXiv preprint arXiv:2506.13131*, 2025. doi: 10.48550/arXiv.2506.13131. URL https://arxiv.org/abs/2506.13131.

[4] Bogdan Georgiev, Javier Gómez-Serrano, Terence Tao, and Adam Zsolt Wagner. Mathematical exploration and discovery at scale. *arXiv preprint arXiv:2511.02864*, 2025. doi: 10.48550/arXiv.2511.02864. URL https://arxiv.org/abs/2511.02864.

[5] Google DeepMind. Mathematical problem repository for alphaevolve. GitHub repository, 2025. URL https://github.com/google-deepmind/alphaevolve_repository_of_problems. Repository accompanying the Mathematical exploration and discovery at scale preprint; accessed 2026-03-21.

[6] Ansh Nagda, Prabhakar Raghavan, and Abhradeep Thakurta. Reinforced generation of combinatorial structures: Ramsey numbers. *arXiv preprint arXiv:2603.09172*, 2026. doi: 10.48550/arXiv.2603.09172. URL https://arxiv.org/abs/2603.09172.

[7] Donald E. Knuth. Claude's cycles. Informal note / PDF on Knuth's preprints page, February 2026. URL https://cs.stanford.edu/~knuth/papers/claude-cycles.pdf. Dated 2026-02-28; revised 2026-03-16.

[8] Terence Tao. The story of erdős problem #1026. Blog post on *What's New*, December 2025. URL https://terrytao.wordpress.com/2025/12/08/the-story-of-erdos-problem-126/. Published 2025-12-08.

[9] Mohammed Abouzaid, Andrew J. Blumberg, Martin Hairer, Joe Kileel, Tamara G. Kolda, Paul D. Nelson, Daniel Spielman, Nikhil Srivastava, Rachel Ward, Shmuel Weinberger, and Lauren Williams. First proof. *arXiv preprint arXiv:2602.05192*, 2026. doi: 10.48550/arXiv.2602.05192. URL https://arxiv.org/abs/2602.05192.

[10] First Proof Project. First batch. Project website, February 2026. URL https://1stproof.org/first-batch.html. First-batch page; site lists February 2026 release context; accessed 2026-03-21.

[11] OpenAI. Our first proof submissions. OpenAI research page, February 2026. URL https://openai.com/index/first-proof-submissions/. Published 2026-02-20.

[12] Wenlin Zhang and Haobo Ma. Lean 4 formal verification of 8/10 #1stproof problems: Complete proofs with ai–human pipeline, partial qed for q4 & q6. Zenodo preprint, February 2026. URL https://zenodo.org/records/18635744. Created 2026-02-13. Zenodo also lists a second record with the same title and metadata at DOI 10.5281/zenodo.18635110.

[13] OpenAI. Advancing science and math with gpt-5.2. OpenAI publication, December 2025. URL https://openai.com/index/gpt-5-2-for-science-and-math. Published 2025-12-11.

[14] Mark Sellke and Steven Yin. On learning-curve monotonicity for maximum likelihood estimators. *arXiv preprint arXiv:2512.10220*, 2025. doi: 10.48550/arXiv.2512.10220. URL https://arxiv.org/abs/2512.10220.

[15] Math, Inc. Introducing gauss, an agent for autoformalization. Company blog post, n.d.. URL https://www.math.inc/gauss. Undated page; accessed 2026-03-21.

[16] Math, Inc. Strong pnt. Project page, n.d.. URL https://math-inc.github.io/strongpnt/. Undated page; accessed 2026-03-21.

[17] Math, Inc. strongpnt. GitHub repository, n.d.. URL https://github.com/math-inc/strongpnt. Repository for the Strong PNT formalization; accessed 2026-03-21.

[18] Jared Duker Lichtman. Gauss – an agentic formalization of the prime number theorem. Fields Institute talk page, October 2025. URL https://www.fields.utoronto.ca/talks/Gauss-agentic-formalization-Prime-Number-Theorem. Talk date: 2025-10-28.

[19] Nat Sothanaphan. Resolution of erdős problem #728: a writeup of aristotle's lean proof. *arXiv preprint arXiv:2601.07421*, 2026. doi: 10.48550/arXiv.2601.07421. URL https://arxiv.org/abs/2601.07421.

[20] Harmonic. Today marks a momentous milestone for ai and mathematics. X post, January 2026. URL https://x.com/HarmonicMath/status/2008693723413225814. Posted 2026-01-06; dynamic-source metadata should be rechecked before camera-ready copy if cited in the main text.

[21] Thomas F. Bloom. Erdős problem #728. ErdosProblems.com entry, January 2026. URL https://www.erdosproblems.com/728. Page last edited 2026-01-06; accessed 2026-03-21.

[22] Thomas F. Bloom. Erdős problem #729. ErdosProblems.com entry, January 2026. URL https://www.erdosproblems.com/729. Page last edited 2026-01-11; accessed 2026-03-21.

[23] Thomas F. Bloom. Erdős problem #397. ErdosProblems.com entry, January 2026. URL https://www.erdosproblems.com/397. Page last edited 2026-01-12; accessed 2026-03-21.

[24] teorth. Erdős problem database. GitHub repository, n.d. URL https://github.com/teorth/erdosproblems. Repository README accessed 2026-03-21.

[25] GIGAZINE. An openai researcher posted that "gpt-5 has solved an unsolved mathematical problem," but it turned out that the problem had already been solved, leading to ridicule from rival developers, including google deepmind ceo demis hassabis. News article, October 2025. URL https://gigazine.net/gsc_news/en/20251020-openai-researcher-announced-gpt-5-math-breakthrough/. Published 2025-10-20.

[26] Erwin Vogelaar. Gênant: Openai beweert dat chatgpt wiskundeproblemen oplost, maar dat klopt niet. Bright.nl news article, October 2025. URL https://www.bright.nl/nieuws/1703437/g-nant-openai-beweert-dat-chatgpt-wiskundeproblemen-oplost-maar-dat-klopt-niet.html. Published 2025-10-20; accessed 2026-03-21.

[27] Zachary C. Lipton and Jacob Steinhardt. Troubling trends in machine learning scholarship. *Queue*, 17 (1), 2019. doi: 10.1145/3317287.3328534. URL https://doi.org/10.1145/3317287.3328534. ACM Queue article; multiple secondary indexes report pages 45–77, but page/article-number formatting varies across services, so pages are omitted here deliberately.

[28] Andrew Gelman. "troubling trends in machine learning scholarship". Statistical Modeling, Causal Inference, and Social Science blog, September 2019. URL https://statmodeling.stat.columbia.edu/2019/09/30/troubling-trends-in-machine-learning-scholarship/. Blog commentary pointing to Lipton and Steinhardt and discussing hype, "provably" language, and advertisement-like ML papers. Accessed 2026-03-21.

[29] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Hugo Larochelle. Improving reproducibility in machine learning research (A report from the NeurIPS 2019 reproducibility program). *Journal of Machine Learning Research*, 22(164):1–20, 2021. URL https://www.jmlr.org/papers/v22/20-303.html. Title checked directly against the JMLR PDF first page.

[30] Andrew M. Bean, Ryan Othniel Kearns, Angelika Romanou, Franziska Sofia Hafner, Harry Mayne, Jan Batzner, Negar Foroutan, Chris Schmitz, Karolina Korgul, Hunar Batra, Oishi Deb, Emma Beharry, Cornelius Emde, Thomas Foster, Anna Gausen, María Grandury, Simeng Han, Valentin Hofmann, Lujain Ibrahim, Hazel Kim, Hannah Rose Kirk, Fangru Lin, Gabrielle Kaili-May Liu, Lennart Luettgau, Jabez Magomere, Jonathan Rystrøm, Anna Sotnikova, Yushi Yang, Yilun Zhao, Adel Bibi, Antoine Bosselut, Ronald Clark, Arman Cohan, Jakob Nicolaus Foerster, Yarin Gal, Scott A. Hale, Inioluwa Deborah Raji, Christopher Summerfield, Philip H. S. Torr, Cozmin Ududec, Luc Rocher, and Adam Mahdi. Measuring what matters: Construct validity in large language model benchmarks, 2025. URL https://doi.org/10.48550/arXiv.2511.04703. Accepted to the NeurIPS 2025 Datasets and Benchmarks Track; title, acceptance status, and author list cross-checked against the Oxford RML project page and Oxford ORA record.

[31] Oxford Internet Institute. Study identifies weaknesses in how AI systems are evaluated. Press release, November 2025. URL https://www.oii.ox.ac.uk/news-events/study-identifies-weaknesses-in-how-ai-systems-are-evaluated/. Press release accompanying the benchmark-validity study; includes quoted claims about unclear definitions, weak methods, and misleading benchmark conclusions. Accessed 2026-03-21.

[32] D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison. Hidden technical debt in machine learning systems. In *Advances in Neural Information Processing Systems 28*, pages 2503–2511, 2015. URL https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems.

[33] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pages 3207–3214, 2018. doi: 10.1609/AAAI.V32I1.11694. URL https://doi.org/10.1609/AAAI.V32I1.11694.

[34] Nick McGreivy and Ammar Hakim. Weak baselines and reporting biases lead to overoptimism in machine learning for fluid-related partial differential equations. *Nature Machine Intelligence*, 6(10):1256–1269, 2024. doi: 10.1038/s42256-024-00897-5. URL https://doi.org/10.1038/s42256-024-00897-5.

[35] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard S. Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020. doi: 10.1038/s42256-020-00257-z. URL https://doi.org/10.1038/s42256-020-00257-z.

[36] Dennis Ulmer, Christian Hardmeier, and Jes Frellsen. deep-significance — easy and meaningful statistical significance testing in the age of neural networks, 2022. URL https://doi.org/10.48550/arXiv.2204.06815. arXiv preprint; also listed as a contribution to the ML Evaluation Standards Workshop at ICLR 2022 in institutional repositories.

[37] Charles Jones, Daniel C. Castro, Fabio De Sousa Ribeiro, Ozan Oktay, Melissa McCradden, and Ben Glocker. A causal perspective on dataset bias in machine learning for medical imaging. *Nature Machine Intelligence*, 6:138–146, 2024. doi: 10.1038/s42256-024-00797-8. URL https://doi.org/10.1038/s42256-024-00797-8.

[38] Alex Broadbent and Thomas Grote. Can robots do epidemiology? machine learning, causal inference, and predicting the outcomes of public health interventions. *Philosophy & Technology*, 35:14, 2022.

doi: 10.1007/s13347-022-00509-3. URL https://doi.org/10.1007/s13347-022-00509-3. Springer presents this as volume 35, article number 14; issue and expanded page-range metadata vary across indexes.

[39] Gary S. Collins and Karel G. M. Moons. Reporting of artificial intelligence prediction models. *The Lancet*, 393(10181):1577–1579, 2019. doi: 10.1016/S0140-6736(19)30037-6. URL https://doi.org/10.1016/S0140-6736(19)30037-6.

[40] Liz Fuller-Wright. "AI Snake Oil": A Conversation with Princeton AI Experts Arvind Narayanan and Sayash Kapoor. Princeton University News, December 2024. URL https://www.princeton.edu/news/2024/12/18/ai-snake-oil-conversation-princeton-ai-experts-arvind-narayanan-and-sayash-kapoor. Interview/article quoting Narayanan and Kapoor on AI that does not work as advertised and predictive-AI systems not backed by scientific evidence. Accessed 2026-03-21.

[41] Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first international conference on machine learning*, 2024.

[42] Martin Funkquist, Ilia Kuznetsov, Yufang Hou, and Iryna Gurevych. Citebench: A benchmark for scientific citation text generation, 2022. URL https://arxiv.org/abs/2212.09577. Using the arXiv submission year; later bibliographic records may surface under 2023 metadata updates.

[43] Yutong Li, Lu Chen, Aiwei Liu, Kai Yu, and Lijie Wen. Chatcite: LLM agent with human workflow guidance for comparative literature summary, 2024. URL https://arxiv.org/abs/2403.02574.

[44] Yubo Wang, Xueguang Ma, Ping Nie, Huaye Zeng, Zhiheng Lyu, Yuxuan Zhang, Benjamin Schneider, Yi Lu, Xiang Yue, and Wenhu Chen. Scholarcopilot: Training large language models for academic writing with accurate citations, 2025. URL https://arxiv.org/abs/2504.00824.

[45] Tong Bao, Mir Tafseer Nayeem, Davood Rafiei, and Chengzhi Zhang. Surveygen: Quality-aware scientific survey generation with large language models, 2025. URL https://arxiv.org/abs/2508.17647.

[46] Haomin Wen, Zhenjie Wei, Yan Lin, Jiyuan Wang, Yuxuan Liang, and Huaiyu Wan. Overleafcopilot: Empowering academic writing in Overleaf with large language models, 2024. URL https://arxiv.org/abs/2403.09733.

[47] Junyi Hou, Huikai Andre Lin, Nuo Chen, Yiwei Gong, and Bingsheng He. Paperdebugger: A plugin-based multi-agent system for in-editor academic writing, review, and editing, 2025. URL https://arxiv.org/abs/2512.02589.

[48] Tal Ifargan, Lukas Hafner, Maor Kern, Ori Alcalay, and Roy Kishony. Autonomous LLM-driven research — from data to human-verifiable research papers. *NEJM AI*, 2(1), 2025. doi: 10.1056/AIoa2400555. URL https://ai.nejm.org/doi/10.1056/AIoa2400555.

[49] Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist: Towards fully automated open-ended scientific discovery, 2024. URL https://arxiv.org/abs/2408.06292.

[50] Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist-v2: Workshop-level automated scientific discovery via agentic tree search, 2025. URL https://arxiv.org/abs/2504.08066.

[51] Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. Agent laboratory: Using LLM agents as research assistants. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 5977–6043.

Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.320. URL https://aclanthology.org/2025.findings-emnlp.320/.

[52] Yixuan Weng, Minjun Zhu, Guangsheng Bao, Hongbo Zhang, Jindong Wang, Yue Zhang, and Linyi Yang. Cycleresearcher: Improving automated research via automated review, 2024. URL https://arxiv.org/abs/2411.00816. First submitted in 2024; later revised in 2025.

[53] Jiabin Tang, Lianghao Xia, Zhonghang Li, and Chao Huang. AI-researcher: Autonomous scientific innovation, 2025. URL https://arxiv.org/abs/2505.18705.

[54] Samuel Schmidgall and Michael Moor. Agentrxiv: Towards collaborative autonomous research, 2025. URL https://arxiv.org/abs/2503.18102.

[55] Ed Li, Junyu Ren, Xintian Pan, Cat Yan, Chuanhao Li, Dirk Bergemann, and Zhuoran Yang. Build your personalized research group: A multiagent framework for continual and interactive science automation, 2025. URL https://arxiv.org/abs/2510.15624.

[56] Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, Khaled Saab, Dan Popovici, Jacob Blum, Fan Zhang, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Pushmeet Kohli, Yossi Matias, Andrew Carroll, Kavita Kulkarni, Nenad Tomasev, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R. D. Costa, José R. Penadés, Gary Peltz, Yunhan Xu, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Towards an AI co-scientist, 2025. URL https://arxiv.org/abs/2502.18864.

[57] InternAgent Team, Bo Zhang, Shiyang Feng, Xiangchao Yan, Jiakang Yuan, Runmin Ma, Yusong Hu, Zhiyin Yu, Xiaohan He, Songtao Huang, Shaowei Hou, Zheng Nie, Zhilong Wang, Jinyao Liu, Tianshuo Peng, Peng Ye, Dongzhan Zhou, Shufei Zhang, Xiaosong Wang, Yilan Zhang, Meng Li, Zhongying Tu, Xiangyu Yue, Wangli Ouyang, Bowen Zhou, and Lei Bai. Internagent: When agent becomes the scientist — building closed-loop system from hypothesis to verification, 2025. URL https://arxiv.org/abs/2505.16938.

[58] Shiyang Feng, Runmin Ma, Xiangchao Yan, Yue Fan, Yusong Hu, Songtao Huang, Shuaiyu Zhang, Zongsheng Cao, Tianshuo Peng, Jiakang Yuan, Zijie Guo, Zhijie Zhong, Shangheng Du, Weida Wang, Jinxin Shi, Yuhao Zhou, Xiaohan He, Zhiyin Yu, Fangchen Yu, Qihao Zheng, Jiamin Wu, Mianxin Liu, Chi Zhang, Shaowei Hou, Shuya Li, Yankai Jiang, Wenjie Lou, Lilong Wang, Zifu Wang, Jiong Wang, Wanghan Xu, Yue Deng, Dongrui Liu, Yiheng Wang, Wenlong Zhang, Fenghua Ling, Shufei Zhang, Xiaosong Wang, Shuangjia Zheng, Xun Huang, Siqi Sun, Shuyue Hu, Peng Ye, Chunfeng Song, Bin Wang, Conghui He, Yihao Liu, Xin Li, Qibin Hou, Tao Chen, Xiangyu Yue, Liang He, Dahua Lin, Bowen Zhou, Bo Zhang, and Lei Bai. Internagent-1.5: A unified agentic framework for long-horizon autonomous scientific discovery, 2026. URL https://arxiv.org/abs/2602.08990.

[59] 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.

[60] Bogdan Georgiev, Javier Gómez-Serrano, Terence Tao, and Adam Zsolt Wagner. Mathematical exploration and discovery at scale, 2025. URL https://arxiv.org/abs/2511.02864.

[61] Tianyu Fan, Fengji Zhang, Yuxiang Zheng, Bei Chen, Xinyao Niu, Chengen Huang, Junyang Lin, and Chao Huang. Deepinnovator: Triggering the innovative capabilities of LLMs, 2026. URL https://arxiv.org/abs/2602.18920.

[62] Lukas Weidener, Marko Brkić, Phillip Lee, Martin Karlsson, Kevin Noessler, and Paul Kohlhaas. From agent-only social networks to autonomous scientific research: Lessons from OpenClaw and Moltbook, and the architecture of ClawdLab and Beach.Science, 2026. URL https://arxiv.org/abs/2602.19810.

[63] Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. S2orc: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.447. URL https://aclanthology.org/2020.acl-main.447/.

[64] Jason Priem, Heather Piwowar, and Richard Orr. Openalex: A fully-open index of scholarly works, authors, venues, institutions, and concepts, 2022. URL https://arxiv.org/abs/2205.01833.

[65] Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6465–6488. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.398. URL https://aclanthology.org/2023.emnlp-main.398/.

[66] Michael D. Skarlinski, Sam Cox, Jon M. Laurent, James D. Braza, Michaela Hinks, Michael J. Hammerling, Manvitha Ponnapati, Samuel G. Rodriques, and Andrew D. White. Language agents achieve superhuman synthesis of scientific knowledge, 2024. URL https://arxiv.org/abs/2409.13740.

[67] Elicit. Elicit: AI for scientific research, n.d. URL https://orion.elicit.com/. Undated product site; accessed 2026-03-21.

[68] OpenClaw. Openclaw documentation, n.d. URL https://docs.openclaw.ai/. Undated documentation site; accessed 2026-03-21.

[69] ClawdLab. Clawdlab, n.d. URL https://www.clawdlab.xyz/. Undated project site; accessed 2026-03-21.

[70] Molecule Protocol. science.beach. GitHub repository, n.d. URL https://github.com/moleculeprotocol/science.beach. Undated repository citation; accessed 2026-03-21.

[71] Kaiyu Yang, Aidan M. Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. Leandojo: Theorem proving with retrieval-augmented language models, 2023. URL https://arxiv.org/abs/2306.15626.

[72] Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. minif2f: A cross-system benchmark for formal olympiad-level mathematics, 2021. URL https://arxiv.org/abs/2109.00110.

[73] Trieu H. Trinh, Yuhuai Wu, Quoc V. Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024. doi: 10.1038/s41586-023-06747-5. URL https://www.nature.com/articles/s41586-023-06747-5.

[74] AlphaProof and AlphaGeometry teams. AI achieves silver-medal standard solving international mathematical olympiad problems. Google DeepMind blog, July 2024. URL https://deepmind.google/blog/ai-solves-imo-problems-at-silver-medal-level/. Published 2024-07-25; accessed 2026-03-21.

[75] Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: A benchmark for general AI assistants. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=fibxvahvs3.

[76] 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, 2023. URL https://arxiv.org/abs/2311.12022.

[77] 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, 2025. URL https://arxiv.org/abs/2501.14249.

[78] OpenAI. Evaluating AI's ability to perform scientific research tasks. OpenAI research publication, 2025. URL https://openai.com/index/frontierscience/. Published 2025-12-16; accessed 2026-03-21.

## A Full Architecture Reference

This appendix provides a self-contained technical reference for the **PoggioAI/**`MSc` architecture: the execution graph, agent inventory, track-level details, optional rigor modules, campaign orchestration, and the project-level code layout. It consolidates material that was spread across the system-overview, design-choices, and implementation sections of an earlier draft.

### A.1 Full pipeline graph description

At the core of the release is a fixed single-run LangGraph workflow. "Fixed" refers to the execution topology: the graph always traverses the same high-level stages and routing logic, even though the generated content is stochastic.

The graph is organized into six phases.

**Phase 1: Discovery and feasibility screening.** The run begins with `persona_council`, followed by `literature_review_agent`, a literature-review gate, `brainstorm_agent`, and `formalize_goals_agent`. This phase grounds the task in prior work, checks feasibility, and converts a high-level objective into a concrete technical plan.

**Phase 2: Track planning and routing.** Planning writes a decomposition into theory, experiment, or mixed tracks, recorded in `paper_workspace/track_decomposition.json`. The track router uses this contract to decide which execution branches to launch.

**Phase 3: Parallel technical execution.** When theory work is selected and theorem-oriented modules are enabled, the graph runs the theory path through mathematical literature review, proposal, proving, rigorous verification, empirical verification, and proof transcription. When empirical work is selected, it runs experiment literature review, design, execution, verification, and transcription. Both tracks can run independently or in parallel.

**Phase 4: Completion verification and results formalization.** Track outputs are merged via `track_merge` and passed to `verify_completion`, which routes in three directions: *complete* (proceed to results synthesis), *incomplete* (return to goal formalization), or *rethink* (return to brainstorming). If the run proceeds, `formalize_results_agent` produces a structured assessment in `results_assessment.pdf`.

**Phase 5: Internal consistency and follow-up.** A duality check is applied after results formalization. Failure triggers follow-up literature work and renewed planning; success advances to paper production. The file `followup_decision.json` captures the routing outcome.

**Phase 6: Paper production and editorial quality assurance.** The pipeline runs `resource_preparation_agent`, `writeup_agent`, `proofreading_agent`, `reviewer_agent`, and `validation_gate`. Validation failure routes back to writeup for revision before a final success state.

**Loopbacks.** A failed literature-feasibility screen routes back to `persona_council`; `verify_completion` can send the run back to `formalize_goals_agent` or to `brainstorm_agent`; a failed duality gate triggers follow-up literature work; and a failed validation gate returns control to writeup.

**Human steering surfaces.** The system exposes: (i) an initial task specification passed to the launcher; (ii) live steering through a TCP control socket and HTTP API during execution; (iii) resume from a named stage; (iv) context injection via an `inputs/` directory; and (v) campaign-level control when the optional campaign engine is used.

## A.2 Agent inventory and specialist roles

The current release decomposes work into the following specialist agents, grouped by phase.

| **Phase** | **Agent** | **Role** |
|---|---|---|
| Discovery | `persona_council` | Three-perspective debate (practitioner, theorist, narrative) to evaluate direction |
| Discovery | `literature_review_agent` | Literature grounding and feasibility gate |
| Planning | `brainstorm_agent` | Convert research prompt to operational ideas |
| Planning | `formalize_goals_agent` | Structured goal state, milestone definition |
| Theory | `math_literature_agent` | Mathematical literature review for the theory track |
| Theory | `math_proposer_agent` | Propose claims, lemmas, and theorem statements |
| Theory | `math_prover_agent` | Construct proof drafts |
| Theory | `math_rigorous_verifier_agent` | Rigorous verification of proof drafts |
| Theory | `math_empirical_verifier_agent` | Empirical/numerical checks on mathematical claims |
| Theory | `proof_transcription_agent` | Transcribe accepted proofs into paper workspace |
| Experiment | `experiment_literature_agent` | Empirical literature review and baseline identification |
| Experiment | `experiment_design_agent` | Experimental design specification |
| Experiment | `experimentation_agent` | Execute experiments (local subprocess, Docker, or SLURM) |
| Experiment | `experiment_verification_agent` | Post-hoc verification of experimental results |
| Experiment | `experiment _transcription_agent` | Transcribe results into paper workspace |
| Synthesis | `track_merge` | Merge theory and experiment outputs |
| Synthesis | `verify_completion` | Three-way routing: complete / incomplete / rethink |
| Synthesis | `formalize_results_agent` | Structured results assessment |
| Editorial | `resource_preparation_agent` | Prepare assets, bibliography, resource inventory |
| Editorial | `writeup_agent` | Manuscript generation (markdown or LaTeX) |
| Editorial | `proofreading_agent` | Copyediting and revision artifacts |
| Editorial | `reviewer_agent` | Internal review with scoring |
| Editorial | `validation_gate` | Final artifact-contract and review-threshold check |

Table 4: Agent inventory in the current **PoggioAI**/`MSc` release, grouped by pipeline phase.

## A.3 Theory track details

The theorem-oriented path is optional and enabled only when the planner selects theory work and math agents are available (`-enable-math-agents`). The path includes six agents: `math_literature_agent`, `math_proposer_agent`, `math_prover_agent`, `math_rigorous_verifier_agent`, `math_empirical_verifier_agent`, and `proof_transcription_agent`.

The corresponding artifact surface is explicit: the math workspace stores a claim graph in `claim_graph.json`, proof files under `proofs/`, verification traces under `checks/`, and a reusable lemma store in `lemma_library.md`. The mathematical path is treated as a workspace with claims, dependencies, and verification traces rather than as one long free-form proof draft.

These agents provide structure for theorem-oriented work and for integrating accepted claims into the paper, but they do not establish mathematical truth by oracle. The rigorous and empirical verifiers are supportive checks, not substitutes for expert proof review.

## A.4 Experiment track details

When empirical work is selected, the graph runs `experiment_literature_agent`, `experiment_design_agent`, `experimentation_agent`, `experiment_verification_agent`, and `experiment_transcription_agent`.

This path writes its own workspace: `experiment_literature.md`, `experiment_baselines.json`, `experiment_design.json`, `experiment_rationale.md`, `execution_log.json`, `verification_report.md`, and `verification_results.json`. When experiment transcription runs, the paper workspace additionally receives `experiment_report.tex` and `experiment_report.pdf`.

Experiment execution is delegated to an external execution layer. In the current release this includes AI-Scientist-v2 integration and optional SLURM submission. Verification occurs *after* execution and serves as a post-hoc assessment, not a pre-execution code safety guarantee.

## A.5 Counsel protocol specification

When `-enable-counsel` is active, selected specialist stages become a multi-model protocol with sandboxing, debate, synthesis, and artifact promotion.

**Model pool.** In the default configuration, the debate pool includes Anthropic Claude Opus 4.6, OpenAI GPT-5.4, and Google Gemini 3 Pro Preview, with Claude Sonnet 4.6 as the synthesis model. These choices are configurable through the model configuration file.

**Protocol.** Each model first works in an isolated sandbox copy of the workspace, producing an independent candidate output. Models then critique the candidate solutions over several debate rounds. A synthesis model selects and promotes a single authoritative output back to the main workspace.

**Target stages.** Counsel targets stages where disagreement is informative: literature review, experiment design, and manuscript writing.

**Cost profile.** Counsel increases per-stage cost substantially but changes the failure mode from "one model silently committed to one line of reasoning" to "multiple models exposed and critiqued alternative drafts before promotion."

## A.6 Tree search algorithm

When `-enable-tree-search` is active, the linear proof stage in the theory track is replaced by a DAG-layered best-first controller that explores multiple proof strategies in parallel.

**Algorithm.**

1. Frontier claims are selected from the claim graph.
2. Alternative proof strategies are generated for each frontier claim.
3. Branches are scored using promise, graph impact, cost efficiency, depth, and diversity.
4. Top candidates are executed in forked workspaces.
5. Failed branches can spawn debugging children up to a configurable depth.
6. Low-scoring branches are pruned.

**Artifacts.** The search state is persisted in `tree_search_state.json` and branch-level workspaces are stored under `tree_branches/`, making the search process inspectable rather than hidden inside one monolithic prover call.

## A.7 Campaign orchestration

The single-run graph is the primary execution unit. On top of it, the project provides an optional campaign layer driven by heartbeat scripts, cron, or OpenClaw.

Campaigns repeatedly launch or resume runs, validate stage artifacts, distill stage memory, track budget, attempt autonomous repair when artifacts are missing, and notify the user. Campaign state is recorded explicitly, and circuit breakers prevent runaway budget consumption.

The launcher supports full-workspace resume, resume from a named stage, and addition of context files under `inputs/`. During execution, the interaction layer exposes both a TCP control socket and an HTTP steering API, allowing pause, instruction injection, and status inspection without manually reconstructing context.

## A.8 Mathematical reasoning pipeline

The theory track implements a structured mathematical reasoning pipeline rather than a single free-form proof attempt:

1. **Mathematical literature review** (`math_literature_agent`): surveys existing results relevant to the theory claims.
2. **Claim proposal** (`math_proposer_agent`): proposes claims, lemmas, and theorem statements, populating the claim graph.
3. **Proof construction** (`math_prover_agent`): builds proof drafts for claims in the graph. When tree search is enabled, this stage expands into parallel branch exploration.
4. **Rigorous verification** (`math_rigorous_verifier_agent`): checks proof drafts for logical gaps, missing cases, and dependency errors.

5. **Empirical verification** (`math_empirical_verifier_agent`): runs numerical or constructive checks against mathematical claims.
6. **Transcription** (`proof_transcription_agent`): promotes accepted proofs into the paper workspace as LaTeX-ready theorem environments.

The claim graph (`claim_graph.json`) tracks claim status (proposed, proved, verified, transcribed, failed) and inter-claim dependencies. The lemma library (`lemma_library.md`) accumulates reusable intermediate results across proof attempts.

## A.9 Project structure and module map

The repository is organized around a thin launcher and a core package. The top-level entry point `launch_multiagent.py` is lightweight; most workflow logic lives inside the `consortium` package.

| **Component** | **Location** | **Role** |
|---|---|---|
| Launcher and run lifecycle | `launch_multiagent.py`, `consortium/runner.py` | CLI entry point, config loading, workspace creation, checkpoint setup, execution lifecycle |
| Workflow topology and state | `consortium/graph.py`, `consortium/state.py`, `consortium/workflow_utils.py` | LangGraph definition, track routing, gates, shared validation helpers, state schema |
| Model and counsel layer | `consortium/models.py`, `consortium/utils.py`, `consortium/counsel.py` | Model registry, provider mappings, multi-model sandbox/debate/synthesis |
| Specialist agents | `consortium/agents/` | Discovery agents, theory agents, experiment agents, merge, writeup, proofreading, reviewer |
| Domain toolkits | `consortium/toolkits/*` | Search, experimentation, writeup, math, filesystem, ideation, communication |
| Validation and supervision | `consortium/supervision/` | Artifact validation, review checks, traceability checks |
| Optional advanced modules | `consortium/tree_search/`, `consortium/campaign/`, `consortium/interaction/`, `consortium/external_tools/` | Tree search, campaign engine, steering APIs, experiment execution integrations |

Table 5: Implementation boundary. **PoggioAI**/`MSc` is the project-level system; `consortium` is the package that implements the workflow.

## A.10 Run directory layout

Each run writes to a timestamped directory under `results/consortium_<timestamp>/`. The run directory is the main interface through which the system exposes its internal state.

- **Run root.** Top-level deliverables: `final_paper.tex`, optionally `final_paper.pdf`, plus `run_token_usage.json`, budget files, `checkpoints.db`, inter-agent messages, and memory backup.
- **Paper workspace** (`paper_workspace/`). Literature review, research plan, risk register, track decomposition, results assessment, follow-up decision, follow-up literature notes, experiment report transcription, resource inventory, and bibliography.
- **Experiment workspace** (`experiment_workspace/`). Baselines, design specification, rationale, execution log, and verification reports.

- **Experiment runs** (`experiment_runs/<uuid>/`). Concrete experiment executions launched by the external execution layer.
- **Math workspace** (`math_workspace/`). Claim graph, proof drafts, checks, and lemma library (when theorem-oriented agents are enabled).
- **Counsel sandboxes** (`counsel_sandboxes/`). Per-agent, per-model sandbox outputs before synthesis (when counsel is enabled).
- **Tree-search state** (`tree_search_state.json` and `tree_branches/`). Branch-level search state and forked workspaces (when tree search is enabled).

## B Extended Related Work

The literature closest to **PoggioAI/**`MSc` does not form a single clean lineage. It spans manuscript-local writing assistants, literature-review and survey systems, manuscript-first research-to-paper agents, broader autonomous discovery systems, and open multi-agent scientific platforms. We find it more useful to compare these systems by three questions than by raw agent count: *what scholarly artifact is produced*, *what substrate the system can actually act on*, and *what verification loop constrains error accumulation.* Under that lens, **PoggioAI/**`MSc` is best understood as a manuscript-first, artifact-driven research-to-paper system for quantitative research, rather than as a generic autonomous-science platform.

### B.1 Historical precursors: scholarly infrastructure, citation grounding, and editor-native assistance

The pre-agent history of research automation is largely a history of stronger anchoring. Before multi-agent research workflows became feasible, the main bottlenecks were access to machine-readable scholarly corpora, citation graphs, and interfaces for grounding generated text in evidence. At the corpus level, Lo et al. [63] introduced S2ORC, a large-scale structured corpus that made citation-aware retrieval and document-level text mining practical at scale. At the catalog and graph level, Priem et al. [64] introduced OpenAlex as a fully open index of works, authors, institutions, venues, and concepts, giving later systems a programmable substrate for literature discovery, citation traversal, and bibliometric filtering. These resources matter because many later "research agents" inherit their apparent capability from the quality of the scholarly substrate underneath them.

A first wave of LLM-era work then focused on citation-grounded text generation rather than full research automation. Gao et al. [65] introduced ALCE, a benchmark and evaluation framework for end-to-end generation of text with citations, emphasizing fluency, correctness, and citation quality rather than stylistic quality alone. Funkquist et al. [42] similarly proposed CiteBench as a benchmark for citation text generation, helping expose the gap between merely *having* citations and citing evidence that actually supports a claim. On top of this evaluation infrastructure, project-level systems such as ChatCite [43], ScholarCopilot [44], SurveyGen [45], PaperQA2 [66], and Elicit [67] pushed increasingly far into literature synthesis, comparative summary, citation-grounded writing, and systematic-review-style workflows. These systems are highly relevant to **PoggioAI/**`MSc` because they establish an important baseline lesson: in scholarly writing, retrieval quality and claim–evidence linkage often matter more than surface fluency.

A parallel line binds generation directly to live manuscript state rather than to a free-form chat session. OverleafCopilot [46] embeds assistance in an Overleaf/LaTeX workflow, while PaperDebugger [47] adds plugin-based multi-agent review and patching in-editor. This editor-native family is adjacent rather than identical to hypothesis-to-paper systems such as ours, but it highlights a design principle that we also adopt: manuscript state, intermediate artifacts, and revision traces are valuable external objects and should not be collapsed into ephemeral conversational context.

## B.2 Manuscript-first research-to-paper agents

The closest prior line to **PoggioAI**/`MSc` is the family of systems that explicitly target end-to-end or near end-to-end manuscript production. Within this family, a useful internal distinction is between *provenance-first* systems, which emphasize traceability from evidence to prose, and *exploration-first* systems, which emphasize autonomous idea and experiment search before paper synthesis.

Among provenance-first systems, data-to-paper is the clearest reference point. Ifargan et al. [48] describe a stepwise framework that starts from structured data, generates hypotheses, produces analyses and code, and then writes a human-verifiable manuscript with explicit backward traceability from claims to executed steps. The main affinity with **PoggioAI**/`MSc` is therefore not simply "end-to-end research," but the insistence that a manuscript should be supported by inspectable intermediate artifacts rather than by polished prose alone.

A second cluster emphasizes broader autonomous pipeline coverage. The AI Scientist [49] popularized the idea of a repeated loop in which a system generates research ideas, writes code, runs experiments, drafts a full paper, and then critiques its own output through a reviewer-style stage. AI Scientist-v2 [50] extends this exploration-first pattern through agentic tree search and stronger workshop-level paper generation. Agent Laboratory [51] structures the workflow into literature review, experimentation, and report writing, while explicitly showing that human feedback at stage boundaries materially improves outcomes. AI-Researcher [53] frames autonomous scientific innovation itself as the core task, and CycleResearcher [52] explicitly couples automated generation to automated review in a revision loop. AgentRxiv [54] adds a collaborative preprint-server abstraction on top of these laboratory-style agents, enabling cross-run accumulation and retrieval of prior reports rather than treating every run as an isolated research episode.

freephdlabor [55] is especially relevant because it departs from rigid fixed pipelines and instead argues for highly customizable, fully dynamic workflows determined by real-time agent reasoning. It is therefore much closer than earlier systems to a "research operating system" for multi-agent science. Relative to freephdlabor, however, **PoggioAI**/`MSc` intentionally remains more constrained: our emphasis is not maximum architectural freedom, but a tighter artifact contract, clearer theory–experiment coupling, and stronger paper-facing validation gates.

Taken together, these systems establish the manuscript-first research-agent frontier into which **PoggioAI**/`MSc` enters. Our system is closest to this family in artifact contract, but it differs in emphasis. Relative to exploration-first systems, we place more weight on bounded workflow control, explicit intermediate artifacts, and the coordination of theory and experiments under a single paper trajectory. Relative to more data-first systems, we begin from a human-specified hypothesis and support both mathematical and empirical development of that idea.

## B.3 Broader autonomous scientific discovery systems

A nearby but distinct literature optimizes for autonomous scientific discovery more broadly rather than for manuscript production as the primary output. The difference matters: these systems are often stronger on ideation, search breadth, or domain-specific discovery, yet weaker on paper-facing artifact contracts and manuscript governance.

At one end of this spectrum is AI co-scientist. Gottweis et al. [56] describe a multi-agent system that proposes, debates, and iteratively refines scientific hypotheses under scientist guidance, with biomedical validation examples. The center of gravity here is hypothesis generation and research reasoning, not manuscript production. InternAgent [57] and InternAgent-1.5 [58] push further toward long-horizon, closed-loop autonomous discovery across multiple scientific domains, with architectures explicitly organized around generation, verification, and evolution. DeepInnovator [61] addresses a different bottleneck still: not orchestration of the full pipeline, but training models for innovative idea generation through structured research memories and iterative next-idea prediction. In mathematics and algorithmic discovery, AlphaEvolve [59] uses evaluator-guided coding agents for scientific and algorithmic discovery, while Mathematical Exploration and Discovery at Scale [60] studies large-scale machine-assisted mathematical exploration more directly.

These systems are important to our positioning because they enlarge the meaning of "research automation." They suggest that the space is not exhausted by paper-writing systems, and that one can build powerful discovery engines without centering the final manuscript artifact. **PoggioAI**/`MSc`, by contrast, is not intended as a general theory of autonomous scientific discovery. It is narrower: a system whose organizing question is how far one can push a hypothesis-to-paper workflow while keeping human steering sparse and scientific artifacts explicit.

## B.4 Open multi-agent scientific platforms and decentralized research commons

A newer line of work moves away from both bounded single-run pipelines and monolithic laboratory agents toward persistent, open, multi-agent scientific ecosystems. The most important reference here is the OpenClaw/Moltbook ecosystem analyzed by Weidener et al. [62]. OpenClaw itself is not a manuscript-first research agent in the narrow sense; its official documentation presents it as a self-hosted gateway and plugin platform for AI agents across messaging channels and skills [68]. What makes it relevant to research automation is that it became a substrate on top of which scientific-agent ecosystems were rapidly built.

Within that ecosystem, ClawdLab and Beach.Science instantiate two distinct architectural responses. The ClawdLab paper of Weidener et al. [62] presents ClawdLab as a structured laboratory platform with hard role restrictions, PI-led governance, explicit adversarial critique, and externally verifiable evidence requirements. The public ClawdLab site [69] reinforces this emphasis through specialized roles, permanent reports, and cryptographically auditable logs. Beach.Science, by contrast, is presented both in Weidener et al. [62] and in the public repository [70] as an open research commons in which heterogeneous agents and humans exchange, evaluate, and refine hypotheses in a more decentralized environment.

This platform-oriented line is important because it introduces design questions that bounded workflow papers often understate: agent identity, governance, auditability, reward design, role restrictions, and persistent public interaction. Relative to ClawdLab or Beach.Science, **PoggioAI**/`MSc` is deliberately less open-ended and less ecosystem-oriented. It is designed as a governed research-to-paper pipeline rather than as a general-purpose research commons. But these platforms are still highly relevant because they reveal what breaks when one relaxes workflow constraints, and because they show that future scientific-agent systems may need governance and auditability mechanisms that exceed what isolated pipeline papers currently provide.

## B.5 Verification substrates for machine learning theory and applied mathematics

Because **PoggioAI**/`MSc` is built with a current emphasis on machine learning theory and adjacent quantitative fields, it should also be positioned relative to work on formal mathematical verification, not only relative to manuscript agents. Quantitative disciplines differ from many applied domains in that they sometimes admit verification loops much stronger than human critique alone.

LeanDojo [71] is a central enabling platform in this regard: it turns Lean repositories into programmatically accessible training and evaluation environments for theorem proving, with structured proof states, premises, and proof traces. miniF2F [72] provides a cross-system benchmark for Olympiad-style formal mathematics, allowing direct comparison of theorem provers across formal systems. At a more public frontier, AlphaGeometry [73] demonstrated that synthetic-data-driven neuro-symbolic reasoning can reach olympiad-level geometry performance, and AlphaProof [74] brought reinforcement-learning-based formal reasoning into the IMO setting through Lean-verified proofs.

These systems are not research-to-paper agents in the ordinary sense. However, they are directly relevant to theorem-heavy workflows because they illustrate a stronger class of verification loop: when full or partial formalization is feasible, the most meaningful verifier is not another critic agent but a compiler-like checker. This is one reason **PoggioAI/**`MSc` treats theorem-oriented reasoning as an optional rigor-enhancing extension rather than as a cosmetic extra.

### B.6 Evaluation context: capability benchmarks versus manuscript benchmarks

Finally, several recent benchmarks matter for positioning capability claims, even though they are not manuscript-quality benchmarks. GAIA [75] evaluates general AI assistants on real-world, tool-using tasks. GPQA [76] probes graduate-level "Google-proof" scientific question answering. Humanity's Last Exam [77] seeks a broad frontier academic benchmark that remains difficult even as models saturate older tests. FrontierScience [78] evaluates expert-level scientific reasoning on olympiad-style and research-style scientific subtasks.

These benchmarks are useful, but they should not be confused with evaluation of manuscript-centered research systems. A system can score well on GAIA, GPQA, or FrontierScience and still fail to produce a traceable, coherent, properly evidenced paper. Conversely, a system can be strong on structured manuscript production while being narrow in open-ended scientific reasoning. This is precisely why we position **PoggioAI/**`MSc` not by a single scalar "research capability" score, but by its artifact contract, its grounding substrate, and its verification design.

## C Operational Reference

### C.1 Cost and runtime summary

Table 2 in the main body summarizes cost and runtime envelopes for representative configurations. For convenience, the key ranges are: quickstart runs cost \$2–10 in 15–40 min; base pipeline with LaTeX/PDF costs \$10–40 in 30–90 min; math-agent runs cost \$20–60 in 60–150 min; counsel mode costs \$50–200 in 2–5 hrs; tree search without counsel costs \$60–180 in 2–4 hrs; tree search with counsel costs \$200–600 in 4–10 hrs; and full paper campaigns cost \$100–400 in 6–12 hrs. Actual values vary with task scope, model choice, revision loops, and experiment intensity.

The configuration system resolves model settings through a precedence order: built-in defaults, then `.llm_config.yaml`, then explicit CLI overrides. Budget tracking uses workspace-local files: `budget_state.json`, `budget_ledger.jsonl`, and `run_token_usage.json`.

### C.2 Experiment safety model

The experiment path is deliberately usable, but its safety model remains limited. Table 6 summarizes what is and is not in place.

The correct interpretation: the system is suitable for controlled research environments where the operator accepts these risks and uses Docker or SLURM when appropriate, but it should not be described as a hardened execution sandbox.

### C.3 Submission checklist

Before submitting any **PoggioAI/**`MSc`-produced manuscript to a venue (NeurIPS, ICML, ICLR, or similar), a human owner should at minimum:

| Aspect | What is in place | What is not in place |
|---|---|---|
| Process separation | Experiments run in dedicated `experiment_runs/<uuid>/` directories; local execution wrapped in a child process | Local experiments run as the same OS user when not containerized |
| Time control | Local timeout via `CONSORTIUM _EXPERIMENT_TIMEOUT`; SLURM wall-time in cluster mode | No general-purpose per-experiment CPU or memory isolation in local mode |
| Environment isolation | Optional Docker for whole-pipeline execution; optional SLURM for cluster enforcement | No mandatory per-experiment container, namespace, or chroot |
| Network and resource control | Budget cap on API spend | No network isolation; no local cgroup/ulimit enforcement |
| Code safety | Input-schema validation and post-hoc experiment verification | No human code-review gate before execution of AI-generated code |

Table 6: Experiment-execution safety model. The release includes pragmatic controls but not strong sandbox guarantees.

1. Verify novelty and related-work positioning against recent proceedings, not only arXiv.
2. Confirm that baselines are appropriate, figures are not misleading, and at least one key experiment has been rerun independently.
3. Spot-check citations and bibliography entries, including retraction status and faithfulness of attributed claims.
4. Manually revise high-stakes sections: abstract, ethics or broader-impact discussion, and venue-specific formatting or anonymization details.
5. Compile the manuscript locally and obtain at least one domain-expert read before submission.
6. Check venue-specific requirements: page limits, supplementary material format, code-availability statements, and reproducibility checklists.

This checklist is part of the scientific boundary of the system. **PoggioAI**/`MSc` reduces orchestration burden but does not remove the need for expert ownership of the final claims.

## D Evaluation Framework

This appendix specifies the evaluation structure for **PoggioAI**/`MSc`: the core evaluation questions, the metrics taxonomy, and the known failure modes. The current release should be evaluated primarily as a *workflow and artifact system*, not as a benchmarked scientific autopilot.

### D.1 Evaluation questions

For the present release, the central evaluation questions are:

- **Q1 – Structural completion.** Does the system reliably execute the fixed workflow graph and materialize the required stage artifacts under different operating modes?
- **Q2 – Editorial completion.** Under stricter settings, does the run reach a paper-facing state in which the manuscript artifacts, review artifacts, and optional PDF outputs are all present and structurally valid?

- **Q3 – Auditability.** Do the emitted files make it possible for a human supervisor to understand what the system planned, what it executed, what it decided to follow up on, and where additional verification is still required?
- **Q4 – Operational efficiency.** What runtime and spend envelopes are induced by the major execution modes, and how do optional modules such as counsel and tree search change that profile?
- **Q5 – Scientific quality.** Which aspects of output quality can already be instrumented automatically, and which still require external auditing, reruns, or expert review?

These questions deliberately separate *operational success* (Q1–Q4, already measurable from run outputs) from *scientific success* (Q5, only partially observable in the current release).

## D.2 Metrics taxonomy

Evaluation metrics fall into three families with different evidentiary status.

**Structural metrics.** Directly observable from workspace and validator outputs: artifact completion rate, strict editorial-gate pass rate, PDF compile rate when required, resume success after interruption, branch- and stage-level completion statistics.

**Operational metrics.** Already instrumented by run-level files (`run_token_usage.json`, `budget_ledger.jsonl`, completion summaries): total runtime, per-stage runtime, total spend, token counts by model, number of reviewer or rebuttal loops, branch counts in tree-search mode.

**Scientific-audit metrics.** Require a separate evaluation protocol beyond the current package: number of human steering interventions, citation-faithfulness spot checks, independent reruns of experiments, proof review outcomes, external expert judgments of manuscript quality.

Metrics in the first two families support release characterization today; metrics in the third family are the target for a future benchmark paper.

## D.3 Failure modes and ablation axes

Table 7 catalogs the recurring failure modes, current mitigations, and unresolved boundaries.

**Suggested ablation axes.** The most informative future studies would isolate: (i) the effect of counsel on editorial quality; (ii) the effect of tree search on theorem-track productivity; (iii) the effect of strict artifact enforcement on completion rates; and (iv) the effect of campaign repair on long-horizon task success.

| **Failure mode** | **Current mitigation** | **What remains unresolved** |
|---|---|---|
| Late-stage churn and overlong finishing loops | Validation gate, reviewer loop, configurable rebuttal iterations, milestone gates, campaign circuit breakers | A run can still spend substantial time polishing a weak draft instead of improving core content |
| Theory/experiment drift | Explicit `track_decomposition .json`, shared merge stage, completion verification, duality check, follow-up routing | Global coherence depends on intermediate reasoning quality, not only route structure |
| Weak manuscript quality in fully autonomous mode | Artifact enforcement, proofreading, reviewer stage, optional counsel, optional math agents | The system stops short of claiming submission-ready quality without human verification |
| Citation brittleness | Literature review stage, bibliography generation, submission checklist requiring manual spot checks | No internal oracle for citation truthfulness or complete related-work coverage |
| LaTeX fragility | Writeup reflection loop, fail-fast prerequisite checks, optional markdown mode | Complex papers may require manual debugging and local compilation |
| Budget explosion in higher-rigor modes | Budget caps, token ledgers, cost summaries, counsel/tree-search controls | Counsel and tree search remain expensive; quality gains need empirical justification |
| Experiment safety and correctness risk | Subprocess timeout, optional Docker, optional SLURM, post-hoc verification | No per-experiment sandbox, no network isolation, no pre-execution code review gate |

Table 7: Release-stage failure modes, current mitigations, and unresolved boundaries.