HUMANLEAN: FAITHFUL LEAN FORMALIZATION FOR HUMAN PROOF REVIEW

Yuhao Mao[1*], **Chenhao Sun**[1*], **Yani Zhang**[2], **Martin Vechev**[1]
[1]Department of Computer Science
[2]Department of Information Technology and Electrical Engineering
ETH Zürich, Switzerland
{yuhao.mao, chenhao.sun, martin.vechev}@inf.ethz.ch, yanizhang@mins.ee.ethz.ch

ABSTRACT

Reviewing math proofs is demanding. In our survey of NeurIPS, ICML, and ICLR in 2025, reviewers report checking at least part of a proof for only 28.4% of papers with proofs. Proof assistants such as Lean can check formal reasoning, but extending that guarantee to the original math requires *faithfulness*, the semantic alignment between the formalization and the original math. However, we found widespread unfaithfulness in existing Lean formalizations, including OpenAI's recent advances and Axiom's Stacks project. Moreover, faithfulness is algorithmically undecidable in general, making human review of faithfulness essential. We address this by building *HumanLean*, the first system that produces faithful and human-readable Lean formalizations. Across theoretical papers spanning multiple topics, HumanLean with GPT-6-Astra achieves 100% faithfulness, confirmed by two independent human reviewers, and with GPT-5.6-Sol improves faithfulness from 72.5% to 95.0%. With GPT-6-Astra, it also reduces custom definitions by 90% and explicit axioms by 64% compared with the purely agentic baseline. Beyond these benchmarks, we apply HumanLean to OpenAI's unit-distance proof, producing its first faithful public Lean formalization. We also show that HumanLean can aid proof authors to improve their proofs. In collaboration with authors of three additional publications, HumanLean helps identify and correct theorem-statement typos, ambiguous proof arguments, and incorrect results. After these corrections, it faithfully formalizes all goals in these publications without further human intervention. These results support HumanLean as a tool for authors to improve their proofs, with the potential to make proof review more efficient.

1 INTRODUCTION

Reviewing math proofs requires tracing assumptions, checking intermediate arguments, and verifying that cited results apply in the stated setting. This complexity may contribute to limited proof checking in peer review. In our survey of 4,286 theoretical publications accepted at NeurIPS, ICML, and ICLR in 2025, reviewers report checking or attempting to check at least part of a proof for only 28.4% of papers with proofs (Figure 1). This existing review burden is growing as AI-generated math gains prominence, with major advances such as the solution to the longstanding unit-distance problem (OpenAI, 2026e). Verifying these results also requires specialized expertise and substantial time, adding pressure to an already strained review process and increasing the risk that significant proof errors enter the mathematical literature.

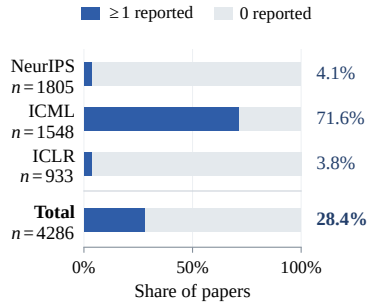


Figure 1: Reported proof-checking rates for top conference publications in 2025. ICML has a relatively high rate because its review form explicitly asks reviewers to report proof checking. The detailed protocol is provided in §A.

[*]Equal contribution.

Proof assistants such as Lean offer a way to make mathematical reasoning machine-checkable. Recent advances in autoformalization and proof generation use language models to translate informal math and construct formal proofs (Wu et al., 2022; Jiang et al., 2023; Liu et al., 2026). However, extending the formal guarantee to the original math requires *faithfulness*, the semantic alignment between the formalization and the original math. We found widespread unfaithfulness in existing Lean formalizations, including OpenAI's Navier–Stokes and Quasi-Riemann formalizations, and Axiom's Stacks project (§E). These differences prevent the formal verification guarantee from carrying over to the original math.

Moreover, faithfulness is algorithmically undecidable in general, making human review of faithfulness essential (Bastounis et al., 2026). However, large formalizations can be difficult to inspect. Formalizations of cited results, verbose definitions, and duplicate concepts complicate this review and limit the benefits of formalization. We address this challenge with *HumanLean*, the first system that produces faithful and human-readable Lean formalizations.

HumanLean uses *compositional formalization* to make source arguments machine-checkable while exposing cited external results and standard mathematical properties as explicit axioms. This boundary mirrors the division of responsibility in conventional proof review, where cited results are assumed correct, and allows each correct source argument to be formalized without recursively formalizing all external dependencies. We illustrate the benefit of compositional formalization in the context of formalizing randomized smoothing (Cohen et al., 2019): it combines a Neyman–Pearson argument with Clopper–Pearson confidence bounds to certify robustness, but neither result is yet formalized in libraries such as Mathlib. Formalizing these dependencies increases the size of the formalization and complicates human review beyond checking the original math.

Across nine theoretical papers spanning various topics, HumanLean with GPT-6-Astra achieves 100% faithfulness, confirmed by two independent human reviewers. With GPT-5.6-Sol, it improves faithfulness from 72.5% to 95.0% compared with the purely agentic baseline. It also increases the number of faithfully formalized goals from 55 to 60 with GPT-6-Astra and from 29 to 76 with GPT-5.6-Sol (§5). Beyond these faithfulness gains, HumanLean with GPT-6-Astra reduces custom definitions from 661 to 68 and explicit axioms from 11 to 4, reductions of approximately 90% and 64%, respectively, compared to the purely agentic baseline. These reductions make the formalizations more manageable for human inspection. Beyond these benchmarks, we apply HumanLean to OpenAI's unit-distance proof, producing its first faithful public Lean formalization (§F).

HumanLean also supports authors in improving their proofs. In collaborations with authors of three additional publications, it helps identify and correct theorem-statement typos, ambiguous proof arguments, and incorrect results. After these corrections, it faithfully formalizes all goals in the three publications without further human intervention (Table 4). These results support HumanLean as a tool for authors to improve their proofs, with the potential to make proof review more efficient.

2 Related Work

Autoformalization and Proof Generation. Autoformalization translates informal math into formal statements checkable by proof assistants such as Lean (Wu et al., 2022; Li et al., 2024). Existing benchmarks span olympiad, undergraduate, and research-level math (Zheng et al., 2022; Azerbayev et al., 2023; Yu et al., 2026; Poiroux et al., 2025; Firsching et al., 2026; Patel et al., 2026). Automated proof systems construct formal proofs for these statements (Jiang et al., 2023; First et al., 2023; Yang et al., 2023; Song et al., 2025; Liu et al., 2026).

Research-Level Autoformalization. For research-level theorem statements, Patel et al. (2023) benchmark translation into Lean while allowing unresolved references to informal definitions. At the project level, Wang et al. (2026) evaluate statement compilation and proof repair on real and convex analysis textbooks and a research paper, while Gloeckle et al. (2026) scale multi-agent formalization to a graduate textbook in algebraic combinatorics. Domain-specific efforts formalize the Vlasov-Maxwell-Landau equilibrium (Ilin, 2026), dimension-8 sphere packing (Hariharan et al., 2026), and neural-network semantics (George et al., 2026). These efforts also differ in how they handle external results and unfinished proofs. Ju et al. (2026) resolve conjectures and formalize their proofs together with supporting results from external papers, recursively formalizing those dependencies. Ren et al. (2026) allow axioms when formalizing three quantum-computation papers. They also

permit unresolved subgoals to become assumptions after a fixed number of attempts, leaving parts of the proofs unformalized. Our compositional formalization treats cited results as dependencies, while forbidding internal proof arguments from being assumed.

3 Unlocking Human Review of Lean Formalizations

Bastounis et al. (2026) prove that resolving ambiguities required for faithful translation can be strictly harder than the halting problem, ruling out a general algorithmic solution. Human review therefore remains essential for checking faithfulness. Yet formalizations can be difficult to review: large projects can contain millions of lines of Lean code, and complex abstractions can obscure their correspondence to the original math. Two criteria guide human readability. *(a) Minimality.* The formalization should focus on the arguments under review, avoiding recursive formalization of external results, which significantly complicates the formalization and incurs unnecessary review burden. Compositional formalization enforces this boundary by treating cited results as dependencies, while requiring internal proof arguments to be formalized. *(b) Correspondence.* Formal definitions, assumptions, and results should clearly correspond to the original math. The formalization should avoid redundant declarations for the same mathematical object and dependencies beyond those needed by the proof. Beyond human readability, initial faithfulness also matters. Reviewing a faithful formalization is much easier than reviewing an unfaithful one: the former corresponds directly to the original math, whereas the latter lacks this correspondence.

These criteria motivate four evaluation dimensions: goal coverage, cost, dependency exposure, and formalization verbosity (Figure 2).

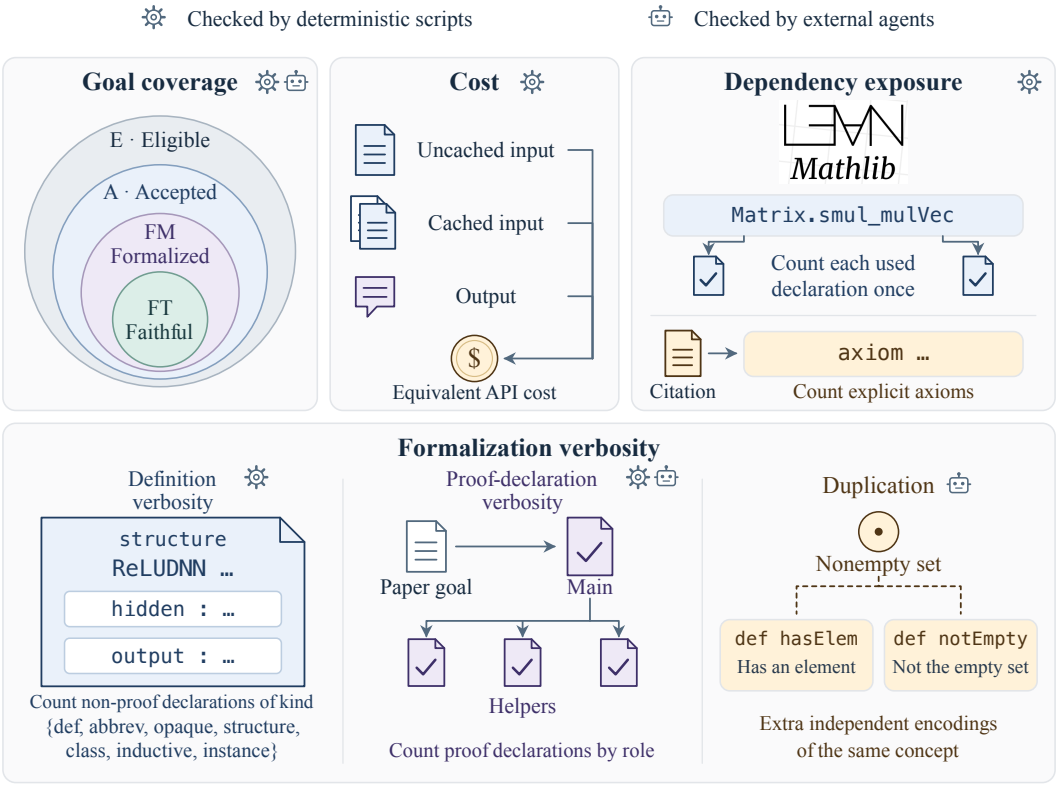


Figure 2: Evaluation metrics for compositional formalization.

Goal coverage and cost. Goal coverage measures how much of the original math is successfully and faithfully formalized. An explicitly numbered result with a self-contained proof is an *eligible* (E) goal. Before attempting formalization, the system informally assesses each goal and marks it as *accepted* (A) if it identifies no known mathematical issue, such as an incorrect proof step. It then

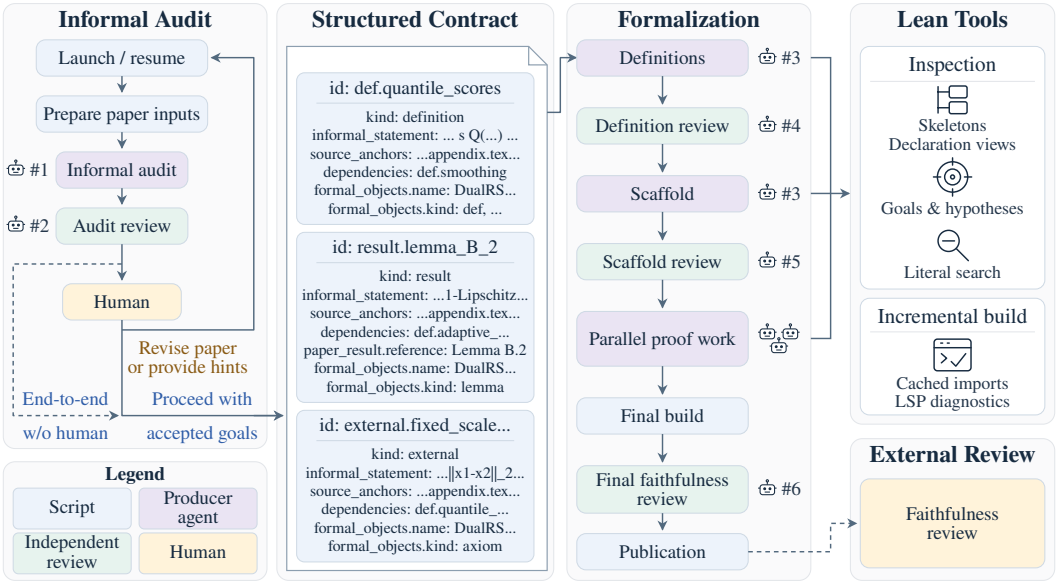


Figure 3: HumanLean workflow. Dashed arrows represent optional paths.

attempts to formalize each accepted goal's statement, dependencies, and proof. A goal is *formalized* (FM) if the final artifact compiles, and *faithfully formalized* (FT) if it also preserves the original math. Formalizations with substantive changes to the intended objects, assumptions, conclusions, or central proof argument count as unfaithful, excluding routine changes in notation or implementation. These stages yield three rates: acceptance (A/E), completion (FM/A), and faithfulness (FT/FM). Acceptance has no absolute target because concerns about the original arguments can justify excluding a goal, while higher completion and faithfulness are preferable. Alongside coverage, cost measures token consumption by type and the equivalent API cost of producing the formalization.

Dependency exposure. Dependency exposure counts the Mathlib declarations used and the custom axioms. The former records reuse of existing formal results, with no universal preference for higher or lower counts. The latter counts assumptions requiring external validation, so fewer custom axioms are preferable for equivalent coverage.

Formalization verbosity. Formalization verbosity counts declarations and redundant representations through three components: (a) definition verbosity counts definition-like declarations, (b) proof-declaration verbosity counts main and helper theorems, and (c) duplication counts overlapping definitions of the same mathematical concept. For equivalent coverage and faithfulness, lower values indicate less material to inspect and fewer redundant representations to reconcile.

4 HumanLean

In this section, we introduce HumanLean, a system for producing faithful, human-readable Lean formalizations. It organizes compositional formalization around a structured contract linking formal objects and proof routes to the original math. Figure 3 presents how it works.

4.1 Design Principles

Scripted workflow control. HumanLean assigns workflow control to scripts, a role often given to orchestrator agents (Wu et al., 2024; Fourney et al., 2024; Gloeckle et al., 2026). Scripts enforce layout and compilation requirements and handle task scheduling, delegation, permission enforcement, output validation, and integration. Agents receive scoped assignments and neither schedule other agents nor approve their own integration.

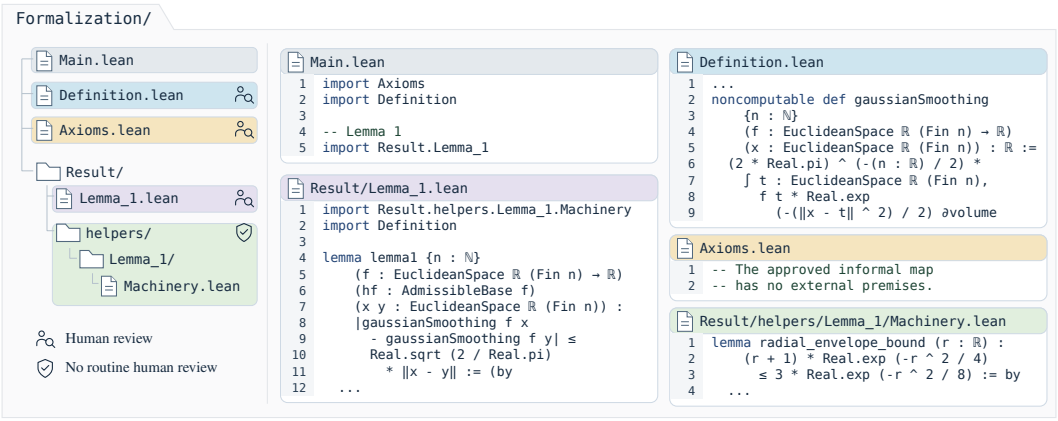


Figure 4: Enforced formalization layout, illustrated with Salman et al. (2019).

Central structured contract. HumanLean converts the original math into a structured JSON contract that gives producers and reviewers a shared specification linked to the source. The contract records definitions, goals, external dependencies, and shared helpers, together with source locations, dependency relations, and intended formal objects. For each goal, a *proof route* records the original reasoning, assumptions, dependency uses, and authorized corrections. Scripts use this contract to prepare scoped assignments and determine task prerequisites.

Isolated workspaces and publication. Each agent receives task-specific inputs in an isolated workspace and can write only assigned outputs and temporary files. This separates concurrent edits and preserves shared inputs for validation. Scripts validate outputs before integration. A separate publication stage gathers internal artifacts, enforces layout restrictions, and improves statement readability through semantically equivalent conversions. The layout separates shared definitions and axioms, result entries grouped by source reference, and private technical helpers (Figure 4). This separation helps human reviewers focus on axiom reliability and main-result faithfulness. Scripts check that implementation-only helpers contained in `Machinery.lean` are all lemma declarations, while human faithfulness review checks whether their use preserves the original proof route.

4.2 From Unstructured Source to Structured Contract

The system takes the original math, a Lean and Mathlib environment, and optional human guidance as input. During informal audit, an agent inventories eligible goals, reconstructs their effective assumptions, and assesses their arguments and dependencies to prepare the structured contract. An independent reviewer checks the contract's faithfulness to the original math. The contract contains accepted goals and their required dependencies. A separate inventory retains unaccepted goals and their concerns for human review. By default, the workflow pauses after audit review for a human to inspect concerns, provide clarifications, revise the original math, or authorize formalization of the currently accepted goals. When applied end-to-end, the workflow proceeds from audit to formalization without a mandatory human pause. To retain goals requiring minimal clarifications, e.g., explicitly assuming non-emptiness for a set clearly intended to be non-empty, it allows source-supported auto-clarifications that preserve the intended setting, boundary cases, parameter regimes, quantitative guarantees, and proof route. The audit review agent checks these clarifications for faithfulness to the intended meaning of the original math.

4.3 From Contract to Formalization

Formalization begins with an agent implementing the definitions identified in the contract. An independent reviewer checks their faithfulness to the contract before statement formalization, so errors can be caught before they propagate across goals. After approval, the workflow resumes the producer agent for scaffolding: formalizing theorem statements with `sorry` proof placeholders and the identified external facts. The agent searches Mathlib for existing formalizations of these facts and declares them as axioms when none are found. A separate review checks the scaffold's faithfulness and the feasibility of the proof routes under it.

Table 1: Benchmark results across nine publications on various topics.

Paper	Topic	System	Goals	Cost				Dependencies		Complexity		
			E/A/FM/FT	API ($)	In (M)	Cache (M)	Out (M)	Mathlib	Axioms	Defs	Main/ Helper	Dup.
Kairouz et al. (2015)	Differential privacy	GPT-5.6-Sol	15/3/3/3	15.4	0.64	25.42	0.14	408	0	48	4/27	3
		Ours w/ GPT-5.6-Sol	15/3/3/3	50.3	4.06	64.87	0.41	302	0	8	4/8	0
		GPT-6-Astra	15/6/6/6	149.1	1.73	113.05	0.38	1381	2	177	10/539	11
		Ours w/ GPT-6-Astra	15/5/5/5	22.8	0.77	10.93	0.08	644	0	9	6/9	0
Telgarsky (2015)	Depth separation	GPT-5.6-Sol	6/2/2/2	8.0	0.42	11.29	0.09	526	0	25	2/43	1
		Ours w/ GPT-5.6-Sol	6/6/6/6	19.9	1.18	26.52	0.23	632	0	8	6/21	0
		GPT-6-Astra	6/3/3/3	6.3	0.14	3.31	0.03	388	0	9	3/24	0
		Ours w/ GPT-6-Astra	6/4/4/4	11.4	0.51	4.14	0.04	477	0	8	4/15	0
Yarotsky (2017)	Approximation	GPT-5.6-Sol	11/1/1/1	20.4	0.81	33.95	0.18	635	0	53	2/82	1
		Ours w/ GPT-5.6-Sol	11/10/10/10	288.0	16.85	479.57	1.44	1625	2	19	10/161	3
		GPT-6-Astra	11/4/4/4	69.6	0.71	51.10	0.23	1489	2	122	6/265	8
		Ours w/ GPT-6-Astra	11/10/10/10	216.0	3.91	149.61	0.55	2015	1	7	10/294	0
Arora et al. (2018)	Neural expressivity	GPT-5.6-Sol	20/6/6/4	9.3	0.40	14.59	0.09	442	0	58	8/64	2
		Ours w/ GPT-5.6-Sol	20/8/8/8	64.6	3.18	101.95	0.55	919	1	16	8/97	4
		GPT-6-Astra	20/9/9/9	24.1	0.38	15.14	0.10	812	2	108	10/113	1
		Ours w/ GPT-6-Astra	20/7/7/7	23.6	0.91	9.71	0.10	743	1	17	7/0	0
Zhang et al. (2018)	Tropical geometry	GPT-5.6-Sol	16/6/6/5	11.9	0.56	17.74	0.13	397	1	114	7/71	6
		Ours w/ GPT-5.6-Sol	16/10/10/10	83.7	5.40	126.08	0.58	1307	2	19	12/155	1
		GPT-6-Astra	16/6/6/6	29.7	0.50	18.63	0.12	862	4	101	8/146	2
		Ours w/ GPT-6-Astra	16/6/6/6	29.1	0.93	14.21	0.11	554	1	4	6/50	0
Cohen et al. (2019)	Randomized smoothing	GPT-5.6-Sol	13/5/5/3	17.1	0.73	28.91	0.13	651	5	23	8/36	1
		Ours w/ GPT-5.6-Sol	13/12/12/10	60.7	3.10	97.57	0.46	1510	3	8	13/67	2
		GPT-6-Astra	13/6/6/6	27.4	0.40	19.08	0.09	957	0	20	8/70	3
		Ours w/ GPT-6-Astra	13/4/4/4	20.0	0.72	9.33	0.07	858	0	4	4/26	0
Salman et al. (2019)	Adversarial robustness	GPT-5.6-Sol	2/1/1/1	5.0	0.29	7.03	0.05	416	4	4	1/8	0
		Ours w/ GPT-5.6-Sol	2/2/2/2	53.6	2.62	93.09	0.29	1128	0	2	2/49	0
		GPT-6-Astra	2/0/0/0	2.6	0.10	0.98	0.01	0	0	0	0/0	0
		Ours w/ GPT-6-Astra	2/1/1/1	10.7	0.36	5.31	0.04	807	0	2	1/16	0
Abbe et al. (2021)	Hierarchical learning	GPT-5.6-Sol	48/5/5/1	10.3	0.80	13.21	0.09	203	0	14	5/2	0
		Ours w/ GPT-5.6-Sol	48/12/12/12	79.3	6.68	103.19	0.57	653	0	7	12/43	2
		GPT-6-Astra	48/7/7/6	25.0	0.67	14.92	0.07	473	1	13	7/15	0
		Ours w/ GPT-6-Astra	48/9/9/9	29.9	1.18	13.29	0.10	584	1	11	9/17	0
Baader et al. (2024)	Neural verification	GPT-5.6-Sol	19/11/11/9	17.2	0.63	27.56	0.18	510	0	70	13/80	6
		Ours w/ GPT-5.6-Sol	19/17/17/15	201.0	15.31	285.41	1.28	878	0	12	16/167	3
		GPT-6-Astra	19/15/15/15	40.0	0.48	27.01	0.16	775	0	111	26/178	9
		Ours w/ GPT-6-Astra	19/14/14/14	159.3	2.80	109.43	0.44	971	0	6	14/125	1

Once scaffolding is complete, the workflow delegates proof construction to at most three agents in parallel. Each proof agent can write only to the corresponding result file containing the proof placeholder and an initially empty private helper file. The agent is instructed to retain the original proof's steps in the result file and place technical lemmas needed for formalization but absent from the original proof in the helper file. Scripts check that the helper file contains only lemmas, with no new definitions or axioms, and that only the proof placeholder is modified in the result file. The proof is merged only after all checks pass.

Agents use Lean tools to inspect declarations and proof states, search modules and declarations, and incrementally compile assigned outputs and their imports. Proof agents also check package scope, statements, proof completeness, and the approved axiom boundary. These tools serve purposes similar to those of Lean LSP MCP (Dressler, 2025).

5 Experiments

End-to-end evaluation setup and baseline. For the end-to-end evaluation, we select nine papers with significant theoretical implications across various fields, none of which has a known public Lean formalization. We compare HumanLean against baseline agents using goal coverage, faithfulness, cost, dependency exposure, and formalization verbosity to assess support for human review. HumanLean's agentic components and the baselines both use Codex, with GPT-6-Astra at medium reasoning and GPT-5.6-Sol at high reasoning. Baselines are Codex sessions with task prompts and

Table 2: Aggregate benchmark results across all nine publications.

	Goals	Goal coverage (%)			Total (per FM)					
System	E/A/FM/FT	Acceptance	Completion	Faithfulness	API ($)	Mathlib	Axioms	Defs	Main/Helper	Dup.
GPT-5.6-Sol	150/40/40/29	26.7	**100.0**	72.5	**114.6** (**2.9**)	4188 (104.70)	10 (0.25)	409 (10.23)	50/413 (1.25/10.33)	20 (0.50)
Ours w/ GPT-5.6-Sol	150/80/80/76	**53.3**	**100.0**	**95.0**	901.1 (11.3)	8954 (111.93)	**8** (**0.10**)	**99** (**1.24**)	83/768 (**1.04/9.60**)	**15** (**0.19**)
GPT-6-Astra	150/56/56/55	37.3	**100.0**	98.2	**373.8** (**6.7**)	7137 (127.45)	11 (0.20)	661 (11.80)	78/1350 (1.39/24.11)	34 (0.61)
Ours w/ GPT-6-Astra	150/60/60/60	**40.0**	**100.0**	**100.0**	522.7 (8.7)	7653 (127.55)	**4** (**0.07**)	**68** (**1.13**)	61/552 (**1.02/9.20**)	**1** (**0.02**)

receive the same environment and permissions as HumanLean, including web search and multi-agent collaboration. All systems perform the same compositional formalization task: preserve faithfulness to the original paper without formalizing external dependencies (full baseline prompts in §B). They operate without human intervention and may exclude goals because of unresolved concerns. To assess the resulting formalizations, environment scripts measure cost and count dependencies and declarations, while an external GPT-6-Astra agent with xhigh reasoning evaluates the formalizations using the metrics in §3. Goals with no identified unfaithfulness are counted as faithful. Because the external agent may miss deviations, the resulting faithfulness rates are upper-bound estimates. HumanLean with GPT-6-Astra additionally undergoes review by independent human experts, who confirm its reported 100% faithfulness.

End-to-end evaluation results. Tables 1 and 2 report per-paper and aggregate results across nine papers. HumanLean improves acceptance and faithfulness with both models. With GPT-5.6-Sol, acceptance rises from 26.7% to 53.3% and faithfulness from 72.5% to 95.0%. With GPT-6-Astra, HumanLean achieves 100% faithfulness. Although the GPT-5.6-Sol baseline accepts the fewest goals, HumanLean with GPT-5.6-Sol accepts the most. The stronger GPT-6-Astra remains more conservative even with HumanLean. As discussed in §3, higher acceptance is not a target: stronger systems may identify problems in the informal assessment that weaker systems miss.

These faithfulness gains hold even though baselines also review their formalizations in eight of nine papers with GPT-6-Astra and all nine with GPT-5.6-Sol. The sole unfaithful goal from the GPT-6-Astra baseline occurs in a reviewed formalization, showing that generic review can leave semantic deviations undetected. HumanLean instead applies dedicated reviews at multiple stages, catching 121 instances of unfaithfulness with GPT-5.6-Sol and three with GPT-6-Astra across nine papers (Table 3). It achieves higher faithfulness with both models, although some deviations still escape review with GPT-5.6-Sol. Review and repeated revisions add cost. Cost per formalized goal rises from $2.9 to $11.3 with GPT-5.6-Sol and from $6.7 to $8.7 with GPT-6-Astra.

Table 3: Unfaithfulness caught by each review stage, summed across nine papers.

Stage	Sol	Astra
Source	26	2
Definition	42	1
Scaffold	37	0
Final proof	16	0
Total	121	3

HumanLean also reduces the material exposed for inspection. Axiom exposure falls from 0.25 to 0.10 per FM for GPT-5.6-Sol and from 0.20 to 0.07 for GPT-6-Astra, reductions of 60% and 65%, respectively. Definition verbosity falls from 10.23 to 1.24 per FM for GPT-5.6-Sol and from 11.80 to 1.13 for GPT-6-Astra, an approximately 90% reduction. Duplicates fall from 0.50 to 0.19 per FM with GPT-5.6-Sol and from 0.61 to 0.02 with GPT-6-Astra, reductions of 62.5% and 97%, respectively. Explicitly prompting the GPT-6-Astra baseline to minimize its formalization reduces some verbosity but leaves a substantial gap with HumanLean (§C). These differences directly affect the review burden: the baselines introduce more than ten custom definitions per formalized goal on average, each potentially requiring recursive dependency tracing. HumanLean further simplifies locating the material to review through its enforced, modular layout, which separates the review interface from technical helpers (Figure 4). It typically uses a single main theorem declaration per goal, while baselines tend to split a goal into multiple main theorems and produce less structured layouts, e.g., creating 70 Lean files to formalize Kairouz et al. (2015).

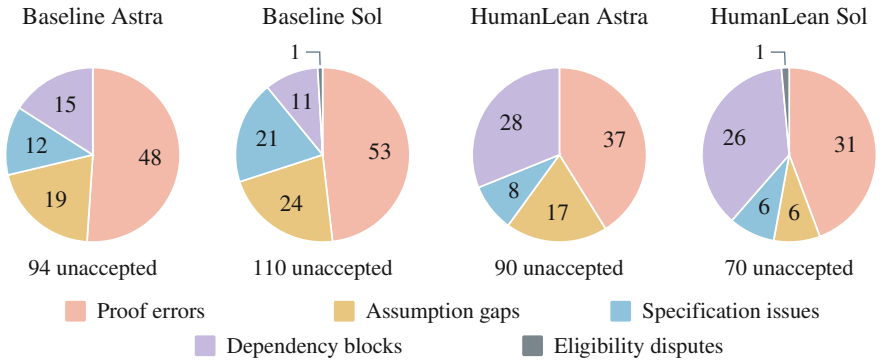


Figure 5: Concerns associated with unaccepted goals across the nine benchmark publications.

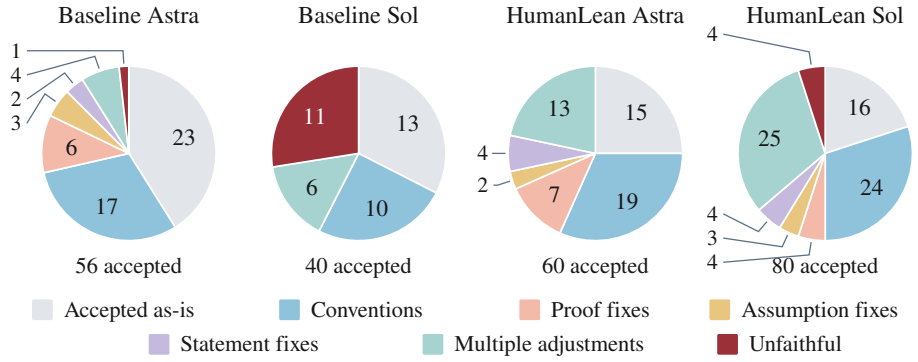


Figure 6: Auto-clarifications made for accepted goals across the nine benchmark publications.

Two independent human reviewers with expertise in Lean and math proofs confirm the 100% faithfulness result for HumanLean with GPT-6-Astra. The reviewers report spending 5–20 minutes per goal, mostly gathering relevant details from the original paper rather than checking their correspondence with the formalization. In contrast, reviewers of baseline formalizations must first identify and gather the relevant definitions, assumptions, and proof steps scattered throughout the Lean code, a task made particularly difficult by the size and verbosity of these formalizations. We are therefore unable to conduct human reviews of the baseline formalizations.

Figures 5 and 6 show concerns about unaccepted goals and auto-clarifications for accepted goals. HumanLean reports fewer proof errors, assumption gaps, and specification issues with both models. By contrast, dependency blocks increase from 15 to 28 with GPT-6-Astra and from 11 to 26 with GPT-5.6-Sol. HumanLean excludes goals whose proofs require defective dependencies, while baselines tend to use a correct subpart of a dependency even when the full dependency is defective. These concerns do not establish errors in the original paper unless confirmed by its authors. HumanLean makes source-supported auto-clarifications with both models, retaining goals that require minimal clarification. The GPT-6-Astra baseline also makes such clarifications, while the GPT-5.6-Sol baseline makes many unfaithful adjustments. §D analyzes overlap among accepted goal sets.

Application to the unit-distance proof. We further address an important question: can HumanLean help check AI-generated math in which no informal error has been found? We study OpenAI's unit-distance proof (OpenAI, 2026e). To date, it has three other public formalizations, all of which we find unfaithful. HumanLean produces its first faithful public Lean formalization, with human experts checking the main theorem's faithfulness and GPT-6-Astra at xhigh reasoning checking the rest. §F details the unfaithfulness findings and compares the sizes of the formalizations.

Human-in-the-loop evaluation. To assess whether HumanLean helps authors improve their proofs, we work with the authors of three additional anonymized publications using HumanLean with GPT-6-Astra. Its informal concerns lead the authors to clarify assumptions, fix theorem-

Table 4: Author evaluation on three publications, anonymized for peer review.

Paper	E/A/FM/FT	Goal coverage (%)			Improvements		
		Acceptance	Completion	Faithfulness	Refined Assumptions	Corrected Results	Fixed Proofs
Paper A	12/12/12/12	**100.0**	**100.0**	**100.0**	1	3	1
Paper B	3/3/3/3	**100.0**	**100.0**	**100.0**	0	1	1
Paper C	18/18/18/18	**100.0**	**100.0**	**100.0**	4	4	2
Total	33/33/33/33	**100.0**	**100.0**	**100.0**	5	8	4

statement typos, replace ambiguous or erroneous proofs, and correct false theorem statements. After these revisions, HumanLean faithfully formalizes all 33 goals without further human intervention, achieving 100% acceptance, completion, and faithfulness (Table 4). The original authors confirm the faithfulness of the final formalizations. This study demonstrates that HumanLean can help proof authors significantly improve proof rigor through valuable informal concerns. It also shows that HumanLean can achieve 100% faithful goal coverage when it finds no remaining concerns in the original math.

6 Discussion

Implications for AI-assisted formalization. We find widespread unfaithfulness in existing AI-assisted Lean formalizations. When the formalization changes the original math, its correctness guarantee no longer establishes the correctness of the original argument. This limits the confidence that formalization can provide in the original math. AI-assisted formalization should therefore prioritize faithfulness and make the correspondence easy for humans to review.

Implications for proof review. HumanLean supports proof authors and reviewers in complementary ways. For authors, it raises informal concerns that help identify and correct problems, along with providing a faithful formalization, as demonstrated by our author collaborations. For reviewers, its faithful, minimal formalizations make unfamiliar math easier to review by keeping the original argument explicit and reducing the material they need to inspect. The compact, modular interface helps them check correspondence with the original math and the reliability of external assumptions. The unit-distance application extends this use to AI-generated math. These complementary roles suggest a practical workflow for peer review: authors resolve concerns and submit formalizations alongside their proofs; reviewers check faithfulness and external assumptions while also assessing the results' significance and presentation.

Implications for LLM training. HumanLean could provide research-level training pairs that align original math with faithful Lean formalizations. This alignment could help models learn to preserve the original proof argument while producing compiling Lean code. Training specialized provers, such as those developed by Ren et al. (2025), on these pairs could improve their suitability for HumanLean's formalization tasks and reduce its reliance on advanced general-purpose LLMs.

7 Conclusion

The limited human review of math proofs motivates automated support for checking their correctness. Yet the widespread unfaithfulness we find in existing Lean formalizations shows that formalization alone cannot provide the intended guarantee for the original math. HumanLean addresses this gap by producing faithful, human-readable Lean formalizations for efficient proof review. Across nine research papers, it improves faithfulness and substantially reduces formalization verbosity, with human reviewers confirming 100% faithfulness for GPT-6-Astra. Its applications extend these benefits to both AI-generated math and author-led proof improvement: HumanLean produces the first faithful public Lean formalization of OpenAI's unit-distance proof and helps authors of three additional papers identify and correct problems in their proofs. After these revisions, it faithfully formalizes all theoretical results. Together, these results demonstrate how HumanLean supports efficient human proof review through faithful, human-readable Lean formalization.

BROADER IMPACT

HumanLean could strengthen research reliability by helping authors correct proof errors and making proof review more accessible. Faithful, readable formalizations could also make research arguments easier to reuse and provide training data for mathematical reasoning models.

Using formalizations in publication decisions requires care. Although HumanLean achieves high faithfulness in our evaluations, each formalization still requires explicit faithfulness checking before it can be accepted as a valid formalization of the original math. Its external assumptions must also be checked: compilation alone does not establish their reliability. Conversely, a failed formalization attempt or an unconfirmed agent concern does not establish a proof error. Reviewers should examine these concerns before drawing conclusions about an author's work, preserving human responsibility for publication decisions.

Wider adoption should also account for access and confidentiality. Requiring formalizations at submission could burden researchers with limited computational resources. Processing unpublished manuscripts through external model services must respect authors' consent and venue confidentiality policies.

REPRODUCIBILITY STATEMENT

We provide the formalization artifacts from the end-to-end evaluations, the code, and the Docker environment in the supplementary material. Survey protocol for Figure 1 is included in §A.

LLM DISCLOSURE

This paper was assisted by PIRA (PIRA Project, 2026), a research-assistant agent powered by GPT-6-Astra. The assistance included implementation, writing, result analysis, and figure preparation. The LLMs used in each experiment are documented separately in the corresponding experimental setup. The authors are fully responsible for the presented final content.

REFERENCES

Emmanuel Abbe, Enric Boix-Adsera, Matthew S. Brennan, Guy Bresler, and Dheeraj Nagaraj. The staircase property: How hierarchical structure can guide deep learning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 26989–27002. Curran Associates, Inc., 2021.

Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. Understanding deep neural networks with rectified linear units. In *International Conference on Learning Representations*, 2018.

Axiom. The Stacks Project, formalized in Lean, November 2026. URL `https://axiommath.ai/research/the-stacks-project-formalized/`.

Axiom Stacks repository. The Stacks Project in Lean, 2026. URL `https://github.com/AxiomMath/Stacks/tree/5ed82d14634e8d9576bb844c7bd4e3da66d450e7`. Revision 5ed82d1.

Zhangir Azerbayev, Bartosz Piotrowski, Hailey Schoelkopf, Edward W. Ayers, Dragomir Radev, and Jeremy Avigad. ProofNet: Autoformalizing and formally proving undergraduate-level mathematics. *arXiv preprint arXiv:2302.12433*, 2023.

Maximilian Baader, Mark Niklas Müller, Yuhao Mao, and Martin Vechev. Expressivity of ReLU-networks under convex relaxations. In *International Conference on Learning Representations*, 2024.

Alexander Bastounis, Fabian Circelli, and Anders C. Hansen. Navier–Stokes lost in translation: Why Lean verification of AI autoformalisation does not guarantee correct natural language proofs, October 2026. URL `https://arxiv.org/abs/2610.08144`. Version 1, October 6, 2026.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1310–1320. PMLR, 2019.

Oliver Dressler. Lean LSP MCP: Tools for agentic interaction with the Lean theorem prover. GitHub repository, 2025. URL https://github.com/oOo0oOo/lean-lsp-mcp.

fbundle/erdos90 repository. ERDOS90: A Lean formalization of Erdős problem 90, 2026. URL https://github.com/fbundle/erdos90/tree/7197bb6c2d2ef0180bde588b3c350a404278d7c4. Formalization C; revision 7197bb6.

Moritz Firsching, Paul Lezeau, Salvatore Mercuri, Miklós Z. Horváth, Yaël Dillies, Calle Sönne, Eric Wieser, Fred Zhang, Thomas Hubert, Blaise Agüera y Arcas, and Pushmeet Kohli. Formal conjectures: An open and evolving benchmark for verified discovery in mathematics. *arXiv preprint arXiv:2605.13171*, 2026.

Emily First, Markus N. Rabe, Talia Ringer, and Yuriy Brun. Baldur: Whole-proof generation and repair with large language models. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 1229–1241. Association for Computing Machinery, 2023.

Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Erkang Zhu, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, Peter Chang, Ricky Loynd, Robert West, Victor Dibia, Ahmed Awadallah, Ece Kamar, Rafah Hosn, and Saleema Amershi. Magentic-One: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*, 2024.

Robert Joseph George, Jennifer Cruden, Will Adkisson, Xiangru Zhong, Huan Zhang, and Anima Anandkumar. TorchLean: Formalizing neural networks in Lean. *arXiv preprint arXiv:2602.22631*, 2026.

Fabian Gloeckle, Ahmad Rammal, Charles Arnal, Remi Munos, Vivien Cabannes, Gabriel Synnaeve, and Amaury Hayat. Automatic textbook formalization. *arXiv preprint arXiv:2604.03071*, 2026.

Sidharth Hariharan, Christopher Birkbeck, Seewoo Lee, Ho Kiu Gareth Ma, Bhavik Mehta, Auguste Poiroux, and Maryna Viazovska. Progress in formalizing sphere packing in dimension 8. *arXiv preprint arXiv:2604.23468*, 2026.

Vasily Ilin. Semi-autonomous formalization of the Vlasov-Maxwell-Landau equilibrium. *arXiv preprint arXiv:2603.15929*, 2026.

Albert Q. Jiang, Sean Welleck, Jin Peng Zhou, Wenda Li, Jiacheng Liu, Mateja Jamnik, Timothée Lacroix, Yuhuai Wu, and Guillaume Lample. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. In *International Conference on Learning Representations*, 2023.

Haocheng Ju, Guoxiong Gao, Jiedong Jiang, Bin Wu, Zeming Sun, Shurui Liu, Leheng Chen, Yutong Wang, Yuefeng Wang, Zichen Wang, Wanyi He, Peihao Wu, Liang Xiao, Ruochuan Liu, Bryan Dai, and Bin Dong. Automated conjecture resolution with formal verification. *arXiv preprint arXiv:2604.03789*, 2026.

Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1376–1385. PMLR, 2015.

Zenan Li, Yifan Wu, Zhaoyu Li, Xinming Wei, Fan Yang, Xian Zhang, and Xiaoxing Ma. Autoformalize mathematical statements by symbolic equivalence and semantic consistency. In *Advances in Neural Information Processing Systems*, volume 37, pp. 53598–53625. Curran Associates, Inc., 2024.

Junqi Liu, Zihao Zhou, Zekai Zhu, Marco Dos Santos, Weikun He, Jiawei Liu, Ran Wang, Yunzhou Xie, Junqiao Zhao, Qiufeng Wang, Lihong Zhi, Jia Li, and Wenda Li. Numina-Lean-Agent: An open and general agentic reasoning system for formal mathematics. *arXiv preprint arXiv:2601.14027*, 2026.

logical-intelligence/erdos-unit-distance repository. Erdős's planar unit-distance conjecture: Lean 4 formalization, 2026. URL https://github.com/logical-intelligence/erdos-unit-distance/tree/b6493074dd103ca32ea4f5e9b0bc9cb3a0379f2e. Formalization B; revision b649307.

OpenAI. openai/math: Mathematical manuscripts and Lean formalizations, 2026a. URL https://github.com/openai/math/tree/4eaee87548250b2682affda7d05d947c05554bae.

OpenAI. Finite time blowup for Navier–Stokes, 2026b. URL https://cdn.openai.com/pdf/32d9f210-8b73-45e0-91bc-82a30aef8a9a/navier-stokes.pdf.

OpenAI. NavierStokesAndEuler: Lean formalizations, 2026c. URL https://github.com/openai/NavierStokesAndEuler/tree/f9e8bc5b38b6e212696e8a30e3e91517af887bbd.

OpenAI. The Quasi-Riemann hypothesis: A zero-free half-plane $\mathrm{Re}(s) > 7/8$, 2026d. URL https://github.com/openai/math/blob/4eaee87548250b2682affda7d05d947c05554bae/preprints/The-Quasi-Riemann-Hypothesis-September-30-2026/paper.pdf.

OpenAI. Planar point sets with many unit distances, 2026e. URL https://cdn.openai.com/pdf/74c24085-19b0-4534-9c90-465b8e29ad73/unit-distance-proof.pdf.

Nilay Patel, Rahul Saha, and Jeffrey Flanigan. A new approach towards autoformalization. *arXiv preprint arXiv:2310.07957*, 2023.

Nilay Patel, Noah Arias, Davit Babayan, Victoria Cochran, Timothy Libman, Hafsah Mahmood, Liam McCarty, Soli Munoz, Laurel Willey, and Jeffrey Flanigan. MathAtlas: A benchmark for autoformalization in the wild. *arXiv preprint arXiv:2605.14061*, 2026.

PIRA Project. PIRA: A research assistant. https://github.com/AlgebraLoveme/PIRA, 2026.

plby/Erdos90 repository. Erdos90: A Lean formalization of the unit-distance proof, 2026. URL https://github.com/plby/Erdos90/tree/2062b0e6c9770c81e397bfe41148e90b92ca0567. Formalization A; revision 2062b0e.

Auguste Poiroux, Antoine Bosselut, and Viktor Kunčak. RLMEval: Evaluating research-level neural theorem proving. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 10946–10957. Association for Computational Linguistics, 2025.

Yuanjie Ren, Jinzheng Li, and Yidi Qi. MerLean: An agentic framework for autoformalization in quantum computation. *arXiv preprint arXiv:2602.16554*, 2026.

Z. Z. Ren, Zhihong Shao, Junxiao Song, Huajian Xin, Haocheng Wang, Wanjia Zhao, Liyue Zhang, Zhe Fu, Qihao Zhu, Dejian Yang, Z. F. Wu, Zhibin Gou, Shirong Ma, Hongxuan Tang, Yuxuan Liu, Wenjun Gao, Daya Guo, and Chong Ruan. DeepSeek-Prover-V2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. *arXiv preprint arXiv:2504.21801*, 2025.

Hadi Salman, Greg Yang, Jerry Li, Pengchuan Zhang, Huan Zhang, Ilya Razenshteyn, and Sébastien Bubeck. Provably robust deep learning via adversarially trained smoothed classifiers. In *Advances in Neural Information Processing Systems*, volume 32, pp. 11289–11300. Curran Associates, Inc., 2019.

Peiyang Song, Kaiyu Yang, and Anima Anandkumar. Lean Copilot: Large language models as copilots for theorem proving in Lean. In *Proceedings of the International Conference on Neuro-symbolic Systems*, volume 288 of *Proceedings of Machine Learning Research*, pp. 144–169. PMLR, 2025.

Stacks Project. The Stacks Project, 2026. URL https://github.com/stacks/stacks-project/tree/a04446e57ec1fbc252a871afcec7752fb2807b14. Revision a04446e.

Matus Telgarsky. Representation benefits of deep feedforward networks. *arXiv preprint arXiv:1509.08101*, 2015.

Zichen Wang, Wanli Ma, Zhenyu Ming, Gong Zhang, Kun Yuan, and Zaiwen Wen. M2F: Automated formalization of mathematical literature at scale. *arXiv preprint arXiv:2602.17016*, 2026.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=BAakY1hNKS.

Yuhuai Wu, Albert Q. Jiang, Wenda Li, Markus N. Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. Autoformalization with large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 32353–32368. Curran Associates, Inc., 2022.

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. In *Advances in Neural Information Processing Systems*, volume 36, pp. 21573–21612. Curran Associates, Inc., 2023.

Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.

Zhouliang Yu, Ruotian Peng, Keyi Ding, Yizhe Li, Zhongyuan Peng, Minghao Liu, Yifan Zhang, Zheng Yuan, Huajian Xin, Wenhao Huang, Yandong Wen, Ge Zhang, and Weiyang Liu. FormalMATH: Benchmarking formal mathematical reasoning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2026*, pp. 12204–12241. Association for Computational Linguistics, 2026. doi: 10.18653/2026.findings-emnlp.579. URL https://aclanthology.org/2026.findings-emnlp.579/.

Liwen Zhang, Gregory Naitzat, and Lek-Heng Lim. Tropical geometry of deep neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5824–5832. PMLR, 2018.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. MiniF2F: a cross-system benchmark for formal olympiad-level mathematics. In *International Conference on Learning Representations*, 2022.

A Survey Protocol for Figure 1

Measurement. Figure 1 measures explicit reviewer reports of checking or attempting to verify part of a detailed mathematical proof, including partial and unsuccessful attempts. These reports do not establish the rigor or completeness of checking. Missing reports do not necessarily imply an absence of checking.

Paper selection. We study accepted main-track papers from NeurIPS, ICML, and ICLR 2025 with public OpenReview records matched to official proceedings, excluding workshops, position papers, and the NeurIPS Datasets and Benchmarks track. Screening uses case-insensitive regular expressions on text extracted from final manuscripts and linked supplementary PDFs. Eligibility requires a line beginning with theorem, lemma, proposition, corollary, or claim followed by a numeric or letter-based identifier, together with a proof heading and a nonempty body. We exclude table-of-contents entries and proof passages marked as sketches, outlines, omitted or deferred proofs, or references elsewhere. The final sample sizes n in Figure 1 are 1,805 for NeurIPS, 1,548 for ICML, and 933 for ICLR, totaling 4,286 papers. This selection process is intentionally conservative.

Classification. We use the latest public versions of official reviews, excluding author responses, meta-reviews, decisions, and follow-up discussions. After retaining narrative fields and removing template text, we select sentences containing keywords such as proof, theorem, lemma, proposition, or corollary, including plurals and common abbreviations. These sentences serve as input to `gpt-5.6-luna` with low reasoning effort in a fresh agent session for each paper. The model identifies whether any sentence suggests checking any part of a proof during review. The numbers of papers with at least one review reporting proof checking are 74 for NeurIPS, 1,108 for ICML, and 35 for ICLR, totaling 1,217 papers.

B Baseline Instructions

The baseline receives the following launch prompt, with parameters substituted at runtime.

```
Set this as the active persistent goal and keep working until it is complete:

Formalize every informal goal in ${arxiv_url} in LEAN4.

An informal goal is an explicitly stated theorem, proposition, lemma, corollary, or
comparable mathematical result with its own explicit numbering in the original paper
and an explicit paper-provided proof argument in the main text, an appendix, or
supplementary proofs.  Section numbers, equation numbers, and agent-invented labels do
not count as result numbering.  Unnumbered claims, unproved assertions, conjectures,
and empirical observations are not informal goals.  A cited result qualifies only if
it meets these criteria and the paper itself supplies a self-contained proof; a
citation to an external proof is insufficient.  Definitions and supporting helpers are
not additional informal goals.

You are allowed to websearch and use subagents properly.  You are allowed to write an
escalation.md to explain concrete cases where formalization is impossible.  Cited
results and cited properties that are not proved in the paper may be accepted as
axioms; make their sources, precise statements, and applicability explicit.  Do not
use axioms to replace the paper's own results or arguments.

Be faithful to the paper's argument; do not invent new proofs or silently fix proofs.
Preserve the original statements, assumptions, and scope.  Identify each informal goal
by its original reference, and report unresolved gaps rather than silently changing or
omitting affected goals.

Keep all writable work and final artifacts under ${workspace_root}.  The matching Lean
4/Mathlib cache is mounted read-only at ${mathlib_root} and is available for use.

Include the following completion criterion in the saved goal.  Mark the goal complete
only when every informal goal is either formalized or escalated due to paper
source-level problems.  Implementation difficulties or usage limits do not count as
paper source-level problems.

Wait for all child agents to finish before completing the goal or exiting.
```

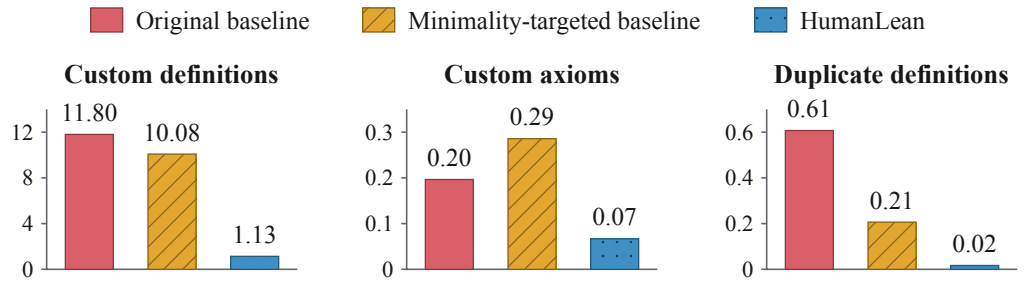


Figure 7: Custom definitions, custom axioms, and extra duplicate-definition families per formalized goal (FM), using GPT-6-Astra (medium reasoning) on nine papers. Scales are independent and zero-based.

C MINIMALITY-TARGETED BASELINE

Interestingly, in the end-to-end evaluation presented in §5, the stronger GPT-6-Astra baseline does not consistently produce smaller formalizations and exhibits higher formalization verbosity than the GPT-5.6-Sol baseline. This raises the question studied below: can explicitly instructing the stronger model to minimize its formalization achieve the same minimality as HumanLean?

We append the following prompt to the baseline prompts (§B) as a minimality-targeted baseline and compare its formalization verbosity with the existing baseline and HumanLean:

```
Prefer reusing applicable Mathlib and project definitions and theorems.  Share
definitions across goals and subagents, avoiding duplicate encodings and redundant
declarations.

Introduce new definitions, helper lemmas, or permitted external axioms only when
needed.

Preserve all required goals, original statements, assumptions, and the paper's proof
route.
```

Figure 7 shows that explicitly requesting minimality reduces some forms of verbosity but leaves a substantial gap with HumanLean. Custom definitions decrease from 11.80 to 10.08 per formalized goal, compared with HumanLean's 1.13, and duplicates fall from 0.61 to 0.21, compared with HumanLean's 0.02. Custom axioms instead increase from 0.20 to 0.29 per goal. These results suggest that an explicit request alone is insufficient to achieve HumanLean's minimality. HumanLean supports minimality through a central contract that agents must follow consistently throughout formalization (§4).

The minimality-targeted baseline also falls short of HumanLean in faithfulness. It reaches 96.8% estimated faithfulness, slightly below the original baseline's 98.2% and still below HumanLean's 100%.

D ACCEPTED GOAL OVERLAP

The systems accept overlapping but distinct goal sets. Of 150 eligible goals, 90 are accepted by at least one system, 32 by all four, and 60 by none. To account for unequal acceptance counts, Figure 8 uses the overlap coefficient $|A_i \cap A_j| / \min(|A_i|, |A_j|)$, where A_i is system i's accepted set pooled across all nine papers. This measures the fraction of the smaller set contained in the larger. A value of 100% indicates containment and does not require identical accepted sets.

Pairwise overlap ranges from 83.9% to 92.5%. HumanLean retains 37 of the 40 baseline goals with GPT-5.6-Sol (92.5%) and 47 of the 56 baseline goals with GPT-6-Astra (83.9%). The two HumanLean configurations share 55 of the smaller set's 60 goals (91.7%). Thus, their accepted sets overlap substantially despite differences in total acceptance.

High aggregate overlap can coexist with local differences. For Kairouz et al. (2015), the GPT-6-Astra baseline and HumanLean accept six and five goals, respectively, but share only three, yielding

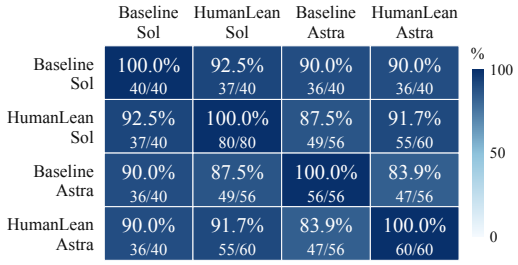


Figure 8: Pairwise accepted-goal overlap pooled across nine papers. Each cell shows the overlap coefficient as a percentage and the exact ratio of shared goals to the smaller accepted-set size. Color uses a 0–100% scale.

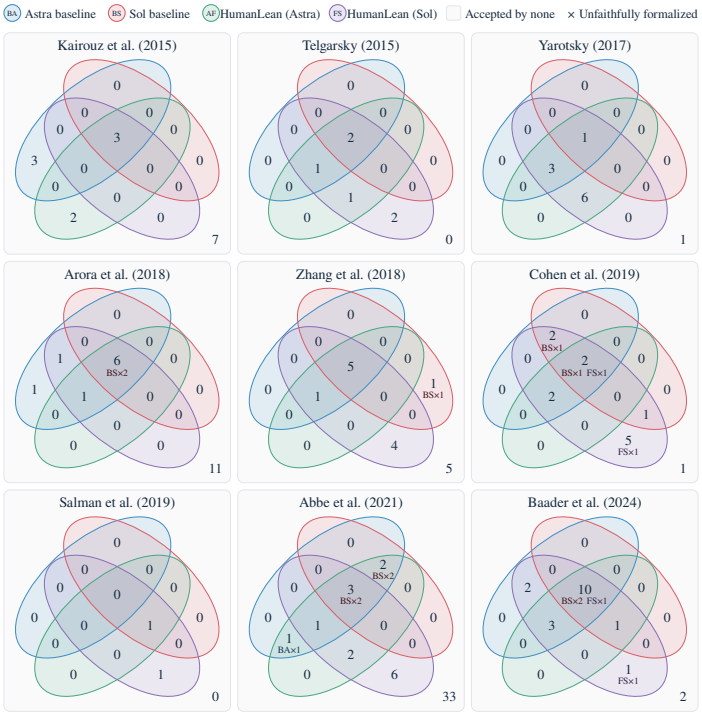


Figure 9: Analysis of goals accepted for each paper. BS, BA, and FS denote the Sol baseline, Astra baseline, and HumanLean with Sol, respectively.

60% overlap (Figure 9). The aggregate matrix therefore complements the per-paper analysis. Neither overlap nor acceptance alone establishes correctness or faithfulness.

E Widespread Unfaithfulness in Lean Formalizations

We found widespread unfaithfulness in OpenAI's Navier–Stokes (OpenAI, 2026b) and Quasi-Riemann (OpenAI, 2026d) Lean formalizations and Axiom's Stacks project Lean formalization (Axiom, 2026). This section presents a few examples.

E.1 OpenAI's Navier–Stokes Formalization

Axis values (Proposition 4.2).

Original: at the axis $X = 0$, the profiles take the prescribed values

$$N_s(0, \eta) = S_n(0, \eta), \qquad Q_s(0, \eta) = S_q(0, \eta)/2,$$

which need not be zero (OpenAI, 2026b, pp. 26–27, equations (4.9)–(4.10)).

Formalization: at $X = 0$, both corresponding profiles equal 0. Their smoothness lemmas assume $X \neq 0$ (OpenAI, 2026c, `ProfileHistories.lean`, lines 584–617).

Pressure-bump placement (Proposition 8.3).

Original: place a smooth bump of integral one inside the *reserved mean interval* I_m (OpenAI, 2026b, pp. 92–93, equation (8.12)).

Formalization: choose the bump using the *full active-shell interval*, with endpoints a, b, centering it at $(a+b)/2$ and normalizing its integral to one (OpenAI, 2026c, `PressureStream.lean`, lines 28–59; `CorrectionInitialization.lean`, lines 4255–4258).

Moment-matrix invertibility (Lemma 4.7).

Original: expand $\det B$ as an integral of determinants, then show that the integrand has *constant nonzero sign* (OpenAI, 2026b, p. 34, Lemma 4.7).

Formalization: obtain roots from weighted integrals that equal zero, then use a *bound on the number of roots* to prove $cB = dB \Rightarrow c = d$ (OpenAI, 2026c, `PowerMomentMatrix.lean`, lines 69–117; `PowerMomentMatrix.lean`, lines 216–249).

E.2 OpenAI's Quasi-Riemann Formalization

Probe scales (Proposition 11.2).

Original: the balanced probe uses

$$(X, Y) = (Z^{1/2}, Z^{1/2}), \qquad C_{\mathrm{I}}(s) = s - 2/3,$$

with signal shift $-2/3$ (OpenAI, 2026d, Proposition 11.2 and the definition of $J_{\mathrm{I},\eta}$).

Formalization: the compensated probe uses

$$(X, Y) = (Z^{17/48}, Z^{23/48}),$$

with signal shift $-11/16$ in the corresponding high estimate (OpenAI, 2026a, `FinalAssemblyHigh.lean`, lines 102–105).

Scope (Corollary 17.4).

Original: bound *any nonnegative family* $\mathcal{N}_v$ satisfying the stated recursive bounds (OpenAI, 2026d, pp. 118–119, Corollary 17.4).

Formalization: bound a *specific family of arithmetic energies*, with prescribed prime pools and rank caps (OpenAI, 2026a, `CanonicalRankExistenceData.lean`, lines 15–49).

The low-side bound (Lemma 6.1).

Original: use a bandlimited majorant and *Poisson summation*, then compare disk areas in a *packing argument* (OpenAI, 2026d, pp. 47–48, proof of Lemma 6.1).

Formalization: split arithmetic correlations into *exceptional and nonexceptional frequencies*, then bound the resulting ideal sums (OpenAI, 2026a, `GramSourceEstimate.lean`, lines 12–41).

E.3 Axiom's Stacks Formalization

Identification (Tag 003Q).

Original: between any two comparison maps, require *exactly one* compatible 2-isomorphism (Stacks Project, Tags 003P–003Q).

Formalization: identify maps when there is *at least one* such 2-isomorphism. Require exactly one class of comparison maps after this identification, without requiring uniqueness of the 2-isomorphism itself (Axiom Stacks repository, Tag 003Q; `Defs.lean`, lines 162–165; `Defs.lean`, lines 240–270).

Table 5: Comparison of HumanLean with Formalizations A–C of the unit-distance proof.

Metric	HumanLean	A	B	C
Axioms	11[a]	0	2[b]	1
Custom definitions	29	12,425	239	59
Proof declarations	151	24,935	692	78
Supplementary: lines of code				
Internal argument formalization	4,618	14,647	8,615	4,021
Total	4,618	975,620	31,501	5,805

[a]Seven axiom declarations encode explicitly cited external results; four state standard mathematical facts used in the manuscript (OpenAI, 2026e).
[b]B encodes its two external assumptions as proposition definitions passed as theorem parameters (logical-intelligence/erdos-unit-distance repository, `Assumptions.lean`, lines 68–132).

Base compatibility (Tag 06N8).

Original: for the two source factorizations, require an equivalence h over $\mathcal{Y}$, giving the *equality* $f \circ h = g$, with the required comparison over the underlying base $\mathcal{C}$ (Stacks Project, Tag 06N8).

Formalization: the conclusion gives *isomorphisms* $f \circ h \cong g$ and $h \circ b \cong a$, without requiring exact agreement over the base or an identity comparison on the base (Axiom Stacks repository, Tag 06N8; `AmeliorationUniqueness.lean`, lines 41–53).

Choosing a removable component (Tag 0GM3).

Original: in the component-intersection graph, choose an endpoint of a pair at *maximal distance* (Stacks Project, Tag 0GM3).

Formalization: grow a connected union one component at a time, stopping when exactly one component remains outside (Axiom Stacks repository, Tag 0GM3; `IrreducibleComponentRemoval.lean`, lines 86–140).

F Applying HumanLean on unit distance proof

There are three public AI-assisted formalizations of OpenAI's unit-distance proof (OpenAI, 2026e), denoted below as Formalization A (plby/Erdos90 repository), B (logical-intelligence/erdos-unit-distance repository) and C (fbundle/erdos90 repository). We first present the unfaithfulness uncovered in these formalizations, then compare our formalization against them.

Formalization A: a stronger threshold (Theorem 2.3).

Original: For a fixed collection of t split primes and the paper's class-number parameter H, require

$$\gamma = t \log 2 - \log H > 0, \qquad \text{equivalently} \qquad C_U := 2^t/H > 1$$

(OpenAI, 2026e, p. 7, Theorem 2.3).

Formalization: For fixed prime data, the construction instead requires

$$C_U > 2\rho/\pi > 1,$$

where $\rho > 0$ is its discriminant-scale bound. The assembly selects enough split primes to meet this stronger threshold (plby/Erdos90 repository, `UnitDistance.lean`, lines 57–76; lines 134–165).

Formalization B: a substituted proof (Lemma 2.6).

Original: For a nonzero algebraic integer β, use $|N_{K/\mathbb{Q}}(\beta)| \geq 1$, since the norm is a nonzero integer (OpenAI, 2026e, p. 9, Lemma 2.6).

Formalization: For a nonzero algebraic integer β, assume every conjugate has absolute value < 1. Invoke the root-of-unity theorem to obtain $\beta^m = 1$ for some $m > 0$, then contradict the strict bound (logical-intelligence/erdos-unit-distance repository, `Section2.lean`, lines 3582–3596).

Formalization C: a construction becomes an axiom (Proposition 3.8).

Original: Construct the field tower, prove prescribed splitting at its levels and derive the discriminant and class-number bounds (OpenAI, 2026e, pp. 12–14, Proposition 3.8).

Formalization: Assume arbitrarily large field levels, fixed split-prime data and class-number control through the declaration `axiom brd_tower_data` (fbundle/erdos90 repository, `Axioms.lean`, lines 25–43; line 100).

Our HumanLean formalization is faithful to the paper. Human experts checked the main theorem, and Astra at `xhigh` effort checked the rest of the formalization.

Metric comparison. Table 5 compares the declarations and lines of code in HumanLean and Formalizations A–C. We count only the axioms, custom definitions and proof declarations used to formalize the main theorem, including custom background results but excluding dependency libraries and unimported drafts. We supplement these declaration counts with lines of Lean code, excluding blank and comment-only lines. C's smaller internal argument formalization reflects its replacement of an internal proof with an axiom.