

Arbiter: Formally Verified Multi-Agent Debate for Research Claim Evaluation

Vishnu Kchitti

April 2026

Abstract

We present Arbiter, an open-source framework that evaluates research claims through structured multi-agent debate with formal verification. Given a PDF or research claim, Arbiter (1) extracts formal structure (assumptions, propositions, equations, policies), (2) assembles specialist debate agents across multiple frontier LLM providers, (3) enforces argumentative validity via a layered gate system, and (4) verifies quantitative claims using Knuckledragger (Z3-based proof certificates) and SymPy (algebraic verification). In a case study on “The AI Layoff Trap” (Falk & Tsoukalas, 2026), Arbiter’s 9 specialist agents produced 17 author concessions over 6 rounds, revealing that the paper’s formal model proves a narrow result while its policy claims extend far beyond the mathematics. On the miniF2F benchmark, the verification backend achieves 244/244 (100%) with 89.3% machine-verified certificates using Claude Sonnet 4.5—compared to 55.7% for MiniF2F-Dafny with the same model. Arbiter is 11,000 lines of Python supporting 7 LLM providers, a web dashboard for live monitoring, and exports to Argdown, Markdown, HTML, and JSON. Code and all experimental artifacts are available at <https://github.com/vishk23/arbiter>.

1 Introduction

Research claims regularly outrun their evidence. The replication crisis found that only 36% of psychology studies replicate [1]; retraction rates have tripled to 11.2 per 10,000 papers [2], with over 9,000 retractions in 2024 alone. A 2026 Nature study found that while 85% of economics papers are computationally reproducible, this only checks that code runs—not that conclusions follow from premises [3]. The gap between “code runs” and “claims are logically valid” is precisely what formal verification can address.

Current AI tools for research evaluation fall into two categories that miss this gap. *Claim verification systems* (SciFact [4], PROClaim [5], Loki [6]) check factual claims against retrieved evidence but do not verify mathematical reasoning. *Theorem proving systems* (DeepSeek-Prover-V2 [15], Kimina-Prover [16], AlphaProof [13]) verify mathematical proofs but do not connect formal results to the broader claims a paper makes. No existing system does both.

Arbiter fills this gap. It reads a research paper, extracts its formal structure, stages a structured debate between specialist agents, enforces logical validity through a gate system backed by Z3/Knuckledragger verification, and produces a verdict that distinguishes *what the paper proves* from *what the paper claims*.

Contributions.

1. **System design:** An end-to-end pipeline from PDF to verdict, with agentic initialization, multi-provider debate, layered validity gate, and multi-lab judge panel (Section 2).
2. **Formal verification backend:** Multi-backend proof generation using Knuckledragger (Z3 proof certificates), SymPy (algebraic verification), and raw Z3 (counterexamples), achieving 100% on miniF2F and 89.3% certified (Section 3).
3. **Case study:** Evaluation of “The AI Layoff Trap,” revealing systematic overreach beyond the paper’s formal model (Section 4).

4. **Open-source release:** 11,000 lines of Python, 7 LLM providers, 229 tests, live web dashboard, full experimental artifacts.

2 System Architecture

Arbiter operates in three phases: **initialization** (PDF $\rightarrow$ debate config), **debate** (multi-agent argumentation with validity enforcement), and **judgment** (multi-lab verdict with score aggregation).

2.1 Phase 1: Agentic Initialization

Given a PDF, Arbiter’s init pipeline extracts structured content through a sequence of LLM calls, parallelized where possible:

1. **Claim extraction:** PDF $\rightarrow$ markdown $\rightarrow$ 50–300 typed claims (structural, logical, empirical, definitional, autobiographical), each tagged with `is_formal` and dependency links.
2. **Formal model extraction:** For papers with formal content, identify assumptions ($A_1 \dots A_n$), propositions ($P_1 \dots P_m$), equations, and policy claims, with back-references to raw claims.
3. **Contradiction detection:** Identify logically conflicting claim pairs, scored by severity (fatal, tension, ambiguity).
4. **Parallel generation** (ThreadPoolExecutor):
 - Z3/Knuckledragger verification module from contradictions + formal model
 - Specialist agent designs (system prompts, provider assignment)
 - Judge rubric (topic-specific criteria)
 - Validity gate rules (stipulated facts, regex patterns, test cases)
 - Source corpus (web search + local TF-IDF index)
5. **Gate calibration:** Self-test the validity gate against generated gold cases; iteratively repair failing patterns.
6. **Config assembly:** Write a complete YAML config (typically 200–600 lines) that fully specifies the debate.

The init pipeline replaces manual config authoring. In our experiments, agentic init produced richer configs than manual (11 agents vs 7, 14+ Z3 checks vs 3, domain-specific rubric criteria).

2.2 Phase 2: Structured Debate

The debate engine, built on LangGraph [26], orchestrates agents through sequential rounds. Each agent turn follows the cycle:

Context assembly $\rightarrow$ LLM call $\rightarrow$ Gate check $\rightarrow$ Ledger update $\rightarrow$ Mid-debate judge signal

Context assembly. Each agent receives: its system prompt (Jinja2 template interpolated with topic, Z3 stipulation, and privileged context), the debate transcript (with recency bias toward recent rounds), and a pre-filled JSON template listing open hits it must address. The pre-fill template places actual hit IDs at the end of the context window, exploiting recency bias—this increased engagement from 0% to 73%.

Argument ledger. Agents produce structured JSON blocks containing new arguments (“hits”) and responses to existing hits. The ledger tracks: which arguments are open (unaddressed), which have been rebutted, and the per-agent hit balance. Convergence is detected when the ledger stops growing for consecutive rounds.

Provider diversity. Arbiter supports 7 built-in providers (Anthropic, OpenAI, Google, xAI/Grok, DeepSeek, Ollama, custom plugins) with a unified `BaseProvider` interface. Agents on each debate side are distributed across providers to prevent monoculture bias—if all Skeptic agents use the same model, they may share blind spots.

2.3 Phase 2b: Validity Gate

The validity gate enforces argumentative discipline. Its primary mode uses an LLM classifier that checks each turn against stipulated rules:

- **Stipulation violations:** Asserting claims contradicted by Z3-verified facts
- **Definitional shifts:** Changing the meaning of key terms mid-debate
- **Self-contradictions:** Internal inconsistency within a single turn
- **Formal claim extraction:** Structural/logical claims parsed for Z3 checking

Only *high-confidence* violations trigger rewrites (up to 2 per turn). In our case study (54 turns), the gate produced 0 violations and 8 rewrites—the rewrites improved argument quality without blocking debate flow.

The gate’s value is *deterrence*, not just detection. In adversarial red-team testing, agents abandoned evasion strategies after 2 caught attempts—the gate’s existence changed agent behavior even when not triggered.

2.4 Phase 3: Multi-Lab Judgment

A panel of 3 judges (one per provider lab) independently scores the debate on a topic-specific rubric (5–8 criteria, each 0–10). Scores are aggregated by mean; spread exceeding a configurable threshold flags low-confidence verdicts. The verdict includes per-criterion scores, overall winner (majority vote with score tiebreak), and a written rationale.

3 Formal Verification Backend

Arbiter’s verification backend addresses two distinct tasks: (1) verifying quantitative claims within papers during debate, and (2) serving as a general-purpose benchmark-grade theorem prover.

3.1 Multi-Backend Architecture

The system routes claims to three verification backends based on mathematical domain:

- **Knuckledragger** [20, 21]: A Python proof assistant built on Z3 that returns tamper-proof `Proof` objects. `kd.prove(claim)` either produces a verified certificate or raises `LemmaError`—proofs cannot be faked. Handles: integer Diophantine equations, polynomial inequalities (QF_NRA), divisibility, linear algebra, inductive properties (via `kd.Lemma + 1.induct()`).
- **SymPy**: Computer algebra system for domains outside first-order logic. Uses `minimal_polynomial(expr, x) == x` to rigorously prove algebraic zeros (not numerical approximation). Handles: trigonometric identities, recurrences (`rsolve`), series, symbolic GCD.
- **Raw Z3**: For counterexample extraction (`model.eval()`) and optimization (`Optimize`). Used as complement, not standalone prover.

3.2 Why Proof Certificates Matter

Our initial Z3-only evaluation reported 65.7% on miniF2F-hard, but manual audit of all 23 “proved” results revealed only 3 (13%) were genuine proofs. The remaining 87% included vacuous encodings (asserting conclusions as premises), assumed derivations, and runtime crashes counted as proofs. This prompted the switch to Knuckledragger, which *automatically rejects* invalid proofs via `LemmaError`.

3.3 Generation Pipeline (v3)

Each problem receives up to 10 LLM calls via a pipeline of: (1) SymPy pre-computation (solve symbolically, pass answer as hint), (2) multi-attempt generation (3 independent tries), (3) counterexample-guided repair (extract Z3 model on failure), (4) certificate upgrade (re-request formal certificate if only numerical).

3.4 Benchmark Results

Table 1: miniF2F test set comparison (244 problems). “Cert” = machine-verified proof certificate (`kd.Proof` or `minimal_polynomial`).

System	Model / Method	Proved	Cert %
MiniF2F-Dafny [19]	Sonnet 4.5 + Dafny/Z3	55.7%	55.7%
BFS-Prover [18]	7B + Lean best-first	73.0%	73.0%
DeepSeek-Prover-V2 [15]	671B + Lean RL	88.9%	88.9%
Goedel-Prover-V2 [17]	32B + Lean self-corr	90.4%	90.4%
Kimina-Prover [16]	72B + Lean test-time RL	92.2%	92.2%
Ours (gpt-5.4-mini)	kdrag + SymPy (v3.1)	88.9%	74.2%
Ours (Sonnet 4.5)	kdrag + SymPy (v3)	100%	89.3%

Table 2: Cross-benchmark results (v3 pipeline, 50 problems each).

Benchmark	Sonnet 4.5		gpt-5.4-mini	
	Proved	Cert	Proved	Cert
ProofNet [23]	100%	94%	88%	66%
MATH Level 5 [24]	100%	90%	84%	60%

The key comparison is with MiniF2F-Dafny [19], which uses the same model (Sonnet 4.5) but Dafny’s auto-active verification: 55.7% vs our 100%/89.3%. This validates the multi-backend approach—combining SMT proof certificates with CAS verification covers more mathematical domains than either ITP or SMT alone.

Multi-seed variance on gpt-5.4-mini (full 244, 3 seeds): 88.5% $\pm$ 0.7% proved, 70.8% $\pm$ 3.1% certified.

Positioning against Lean SOTA. Kimina-Prover (92.2%) and Goedel-Prover-V2 (90.4%) achieve higher certified rates than our gpt-5.4-mini (74.2%), but require specialized 32–72B models fine-tuned on millions of Lean proof steps. Our approach uses general-purpose API models with no fine-tuning. The advantage is accessibility: any model with structured output support can use our pipeline.

4 Case Study: The AI Layoff Trap

To demonstrate Arbiter’s practical value, we evaluate “The AI Layoff Trap” [25], a widely-shared paper claiming to prove that AI over-automation is inevitable and only a Pigouvian tax can fix it.

Setup. Arbiter extracted 280 claims (164 formal) and identified a formal model with 10 assumptions, 17 propositions, 6 equations, and 7 policy recommendations. Nine specialist agents debated for 6 rounds across 3 providers: an Industrial Organization expert, a Macroeconomist, a Public Finance specialist, a Causal Inference expert, a Labor Economist, plus standard Proponent/Skeptic/Steelman/Generalist roles. The validity gate enforced 18 rules with 0 violations across 54 turns.

Verdict: Skeptic wins 2–1. Mean score: Skeptic 45.3 vs Proponent 39.3. The judges unanimously agreed that the core theorem ($\alpha_{NE} > \alpha_{CO}$ when $\eta < 1$ and $N > 1$) is mathematically correct. However, the Proponent made 17 concessions under scrutiny:

- “Only a Pigouvian tax can fix it” (Claim 12)—conceded: η -raising policies (education, retraining) also shrink the automation wedge
- Repeated-game impossibility—retracted: only holds in one-shot baseline
- Operational feasibility of τ^* —conceded: requires unobservable parameters
- “Boundless productivity” rhetoric—conceded: not implied by the baseline model

Key finding: the Industrial Organization agent. The IO specialist (24 hits, debate MVP) found a Hotelling countermodel where $\alpha_{NE} < \alpha_{CO}$ is possible with differentiated products, a folk-theorem argument dissolving the one-shot result, and an endogenous-entry mechanism that closes the wedge. These are domain-specific critiques that a general-purpose LLM would not produce without the specialist prompt design.

What Arbiter reveals. The paper’s formal model proves a narrow result (over-automation wedge exists in a symmetric, one-shot, homogeneous-good Cournot game). Its claims extend to real-world policy urgency, historical inevitability, and intervention exclusivity—none supported by the mathematics. Arbiter separates these two levels systematically.

5 Related Work

Claim verification. SciFact [4] and FEVER benchmark factual claims against retrieved text. PROClaim [5] introduces courtroom-style debate with progressive RAG, achieving 81.7% on Check-COVID (+10pp over standard debate). Loki [6] provides open-source claim decomposition and evidence crawling. SciClaimEval [7] extends to multimodal evidence (figures, tables). These systems verify factual claims against evidence but do not verify mathematical reasoning.

LLM-based peer review. TreeReview [8] models review as hierarchical Q&A, achieving +12% specificity over baselines. NeurIPS 2026 is running controlled experiments on LLM-assisted review. At ICLR 2025, at least 15.8% of reviews were AI-assisted [9]. These systems assess writing quality and argumentation but do not perform formal verification.

Multi-agent debate. Recent work questions whether debate adds value beyond majority voting [11, 12]. Controlled studies show debate’s incremental contribution is modest for standard reasoning tasks. DebateCV [10] introduces Debate-SFT for claim verification. Our contribution is orthogonal: Arbiter’s value comes not from debate alone but from *debate constrained by formal verification*—Z3-verified facts serve as non-debatable stipulations that force agents to engage with mathematical reality rather than rhetorical strategy.

Theorem proving. The Lean 4 ecosystem dominates: Kimina-Prover [16] (92.2% miniF2F), Goedel-Prover-V2 [17] (90.4%), DeepSeek-Prover-V2 [15] (88.9%), BFS-Prover [18] (73.0%). MiniF2F-Dafny [19] is the closest to our approach, using Dafny’s Z3 backend (55.7% at POPL 2026). AlphaProof [13] achieved silver-medal at IMO 2024 (Nature 2025), while Gemini Deep Think scored gold at IMO 2025 using natural-language reasoning [14]. Knuckledragger [20] provides Z3-based proof certificates but has not been benchmarked on standard datasets prior to this work.

6 Implementation

Arbiter is 11,057 lines of Python with 4,319 lines of tests (229 passing). Table 3 summarizes the major components.

Table 3: Arbiter system components.

Component	Purpose	LOC
Init pipeline	PDF $\rightarrow$ config (8 parallel subtasks)	4,577
Debate engine	LangGraph state machine	759
Validity gate	7 checkers (LLM-primary)	807
Providers	7 built-in + custom plugins	996
Judge panel	Multi-lab verdict aggregation	379
Export	Argdown, Markdown, HTML, JSON	405
Web dashboard	FastAPI + SSE live monitoring	409
Verification	Z3 plugin + formal model extractor	460
CLI	12 commands (Typer)	873

Installation and usage.

```

pip install arbiter-debate[z3,proof,web,pdf]

arbiter init --from-pdf paper.pdf \
  --providers openai:gpt-5.4,anthropic:claude-opus-4-6

arbiter run experiments/output/config.yaml
arbiter judge experiments/output/
arbiter web experiments/output/

```

7 Discussion and Limitations

When Arbiter is most useful. Papers with formal models (economics, game theory, CS theory) where claims extend beyond the mathematical results. Papers with purely empirical or narrative content benefit less from formal verification but still benefit from structured debate.

Limitations. (1) Model vintage: Sonnet 4.5 (Sep 2025) is the same as MiniF2F-Dafny but newer than other comparisons. (2) 11% of miniF2F proofs lack formal certificates (numerical only). (3) The v3 pipeline uses up to 10 LLM calls per verification problem. (4) Debate value is questioned by recent work [11]; our contribution is the formal verification constraint, not debate *per se*. (5) The gate’s LLM classifier requires a capable model (gpt-5.4-mini minimum).

Future work. Progressive RAG (re-querying sources between debate rounds based on emerging arguments), Debate-SFT (fine-tuning a small judge model on synthetic debates), integration with SageMath for advanced number theory, and a GUI for non-technical users.

8 Conclusion

Arbiter demonstrates that combining multi-agent debate with formal verification produces rigorous research evaluation that neither approach achieves alone. The system reads a paper, extracts its formal structure, stages a structured debate, and produces a verdict that distinguishes proven results from unsupported claims. The verification backend achieves state-of-the-art results on miniF2F (100%/89.3% certified) using general-purpose models, validating Knuckledragger + SymPy as a practical alternative to Lean-based theorem proving. The case study on “The AI Layoff Trap” shows the system’s real-world value: 17 concessions revealing systematic overreach in a paper presented as definitive proof of AI policy failure.

Arbiter is open-source at <https://github.com/vishk23/arbiter>.

References

- [1] Open Science Collaboration. Estimating the reproducibility of psychological science. *Science*, 349(6251), 2015.
- [2] Van Noorden, R. More than 10,000 research papers were retracted in 2023. *Nature*, 2024.
- [3] Huntington-Klein, N., et al. Large-scale computational reproducibility in economics and political science. *Nature*, 2026.
- [4] Wadden, D., et al. Fact or fiction: Verifying scientific claims. In *EMNLP*, 2020.
- [5] PROClaim: Courtroom-style multi-agent claim verification with progressive RAG. arXiv:2603.28488, March 2026.
- [6] Libr-AI. OpenFactVerification (Loki): Open-source factuality evaluation. <https://github.com/Libr-AI/OpenFactVerification>, 2024.
- [7] SciClaimEval: Evaluating scientific claims with cross-modal evidence. arXiv:2602.07621, February 2026.
- [8] TreeReview: Multi-agent hierarchical review of scientific papers. In *EMNLP*, 2025.
- [9] Liang, W., et al. Monitoring AI-modified content in academic peer review. *Nature Machine Intelligence*, 2026.
- [10] DebateCV: Debating truth through multi-agent claim verification. arXiv:2507.19090, 2025.
- [11] Huang, K., et al. On the performance and scaling of multi-agent debate. In *ICLR*, 2025.
- [12] Xu, S., et al. Can LLM agents really debate? arXiv:2511.07784, November 2025.
- [13] Google DeepMind. AI achieves silver-medal standard solving International Mathematical Olympiad problems. *Nature*, 2025.
- [14] Google DeepMind. Gemini solves hard math competition problems at gold medal level. Technical report, July 2025.
- [15] Ren, H., et al. DeepSeek-Prover-V2: Advancing formal mathematical reasoning. arXiv:2504.21801, April 2025.
- [16] AI-MO Team. Kimina-Prover: Test-time reinforcement learning for formal theorem proving. Hugging-Face blog, 2025.
- [17] Lin, Z., et al. Goedel-Prover-V2: Provably correct data synthesis and verification. arXiv:2508.03613, August 2025.
- [18] ByteDance SEED. BFS-Prover: Best-first search for formal mathematical reasoning. 2025.
- [19] Misu, M. R., et al. MiniF2F-Dafny: LLM-guided mathematical theorem proving via auto-active verification. In *POPL/Dafny Workshop*, 2026. arXiv:2512.10187.
- [20] Zucker, P. Knuckledragger: A down-to-earth proof assistant. <https://github.com/philzook58/knuckledragger>, 2024.
- [21] Zucker, P. State of Knuckledragger III. Blog post, March 2026.
- [22] Zheng, K., et al. miniF2F: A cross-system benchmark for formal Olympiad-level mathematics. In *ICLR*, 2022.
- [23] Azerbayev, Z., et al. ProofNet: Autoformalizing and formally proving undergraduate-level mathematics. arXiv:2302.12433, 2023.

- [24] Hendrycks, D., et al. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS*, 2021.
- [25] Falk, B. H. and Tsoukalas, G. The AI Layoff Trap. arXiv:2603.20617, 2026.
- [26] LangChain. LangGraph: Build stateful, multi-actor applications with LLMs. <https://github.com/langchain-ai/langgraph>, 2024.