Lacuna: A Research Map for Machine Learning Problem Formulation

Martin Weiss [1 2 3] **Alejandro H. Artiles** [1] **Yacine Mkhinini** [1] **Miles Q. Li** [4 1] **Chris Pal** [2 3] **Hugo Larochelle** [5 2] **Nasim Rahaman** [1]

Abstract

Research problem formulation is the cognitive task of turning a vague scientific idea into a testable hypothesis. **Lacuna** is a research map that supports this process for machine learning, using LLMs to turn papers and scholarly metadata into markdown summaries, concept elements, research directions, author and venue pages, and research proposals. Each item keeps links to the primary source records and papers that support it. We release the map with web, markdown, and MCP interfaces, plus scripts for reproducing the agent run. On a theorem-proving seed idea, the agent reaches a research proposal in 85.5 seconds and 7 tool calls, vs. 289.2 seconds and 27 tool calls for a sequential-PDF baseline. On ScholarQA-CS-ML, an ML/AI subset of the OpenScholar benchmark, Lacuna-GPT-4o scores 0.694 under the ScholarQABench rubric judge, compared with 0.672 for the OpenScholar-GPT-4o baseline on the same questions.

1. Introduction

A common moment in machine learning (ML) research is starting from an underspecified seed topic, for example, "automated theorem proving with proof assistants", and refining it into a specific research proposal. We call this *research problem formulation*: moving from a broad topic to a scoped technical question, identifying supporting evidence, and stating what remains unknown. This is the step before retrieval-augmented question answering; it determines what questions are asked.

The name Lacuna refers to a gap or a missing place. We aim to show the paper-grounded neighborhood around such gaps: relevant findings, limitations, methods, and proposed hypotheses that make a new research question inspectable. Lacuna represents this neighborhood as a knowledge graph built from papers, authors, venues, concepts, research directions, and research proposals.

A common large language model (LLM) workflow is to upload papers one at a time and ask for summaries, related work, and next steps. This is slow and expensive: each interaction repeats reading and note-taking that could be shared across users. Lacuna precomputes that intermediate structure as a research map.

Lacuna hosts a research map[1] over 733,795 catalogued papers. It exposes web pages, markdown pages at `/md`, a site-wide schema, and an MCP interface. A model can fetch compact pages, follow typed links, and end at a research proposal page whose claims can be audited against the map. Figure 1 shows the workflow and names the main page types; Section 3 defines them.

Contributions.

1. *Lacuna*, a research map for ML problem formulation, covering 733,795 catalogued papers and a generated knowledge graph of papers, authors, venues, concept elements, research directions, and research proposals.
2. A pipeline and object model for generating research maps: scholarly record reconciliation, source acquisition, core-idea paper summaries, paper summaries with figures, concept elements, research directions, author and venue profiles, and research proposals, all with provenance to the papers and metadata used to create them.
3. Interfaces for humans and LLM tools over the hosted map: web pages, markdown variants, `/llms.txt`, and a reviewer MCP artifact released with the exact prompts, page sequence, transcript, and machine-readable summaries.
4. Empirical evidence that Lacuna supports problem formulation and literature synthesis: a theorem-proving run with a groundedness audit and a cost-and-time comparison against a sequential-PDF baseline, plus

[1]Tiptree Advanced Systems Corporation [2]Mila [3]Polytechnique Montreal [4]McGill University [5]Universite de Montreal. Correspondence to: Martin Weiss <martin@tiptreesystems.com>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

[1]Anonymous web review instance: http://34.8.208.118/home/anonymized.

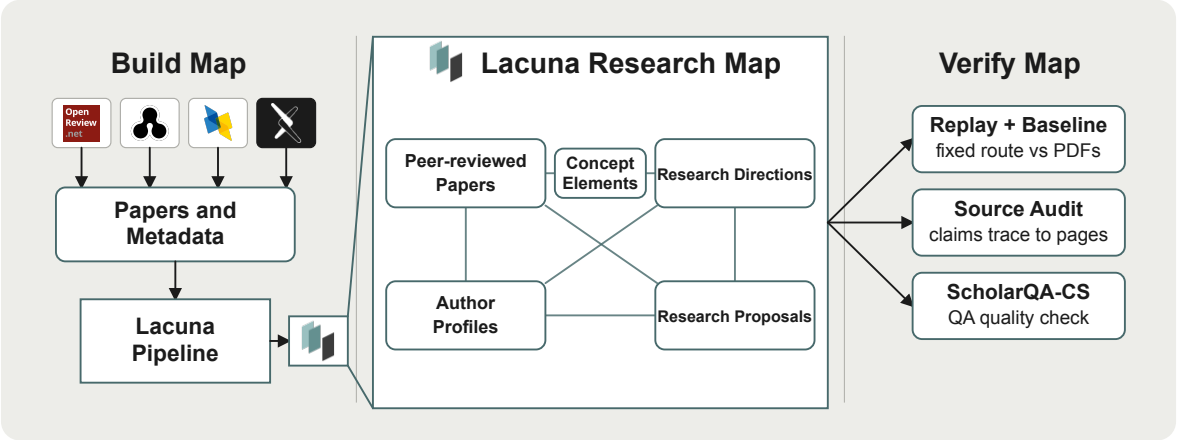

Figure 1. Lacuna builds a research map from scholarly records, then exposes linked paper, author, research direction, and research proposal pages for navigation, reproducible runs, and evidence-quality checks.

a ScholarQA-CS-ML result where Lacuna-GPT-4o scores 0.694 versus 0.672 for the released OpenScholar-GPT-4o answer file.

2. Related Work

Scholarly corpora and scholarly knowledge graphs. Open scholarly infrastructure such as OpenAlex (Priem et al., 2022) and S2ORC (Lo et al., 2020) provides large-scale metadata, identifiers, and machine-readable paper text. Document representation work such as SPECTER (Cohan et al., 2020) learns citation-informed embeddings for scholarly search and recommendation, while scholarly knowledge graph systems such as the Open Research Knowledge Graph (Jaradeh et al., 2019) represent scientific contributions in machine-actionable form. Lacuna builds on this infrastructure with generated, paper-grounded entries for problem formulation.

Information foraging and exploratory search. Information foraging theory models information seeking as a trade-off between information value and the costs of locating and consuming it (Pirolli & Card, 1999). Its notions of information patches and information scent fit scholarly navigation: researchers sample local cues, enter promising topic regions, and abandon weak paths as expected value changes. Lacuna uses this view by making research directions navigable topic regions with low-cost previews and grounded relationships.

Scientific QA and retrieval-augmented synthesis. PaperQA (Lála et al., 2023) and OpenScholar (Asai et al., 2026) answer well-formed scientific questions by retrieving evidence and generating citation-backed responses; SciFact verifies claims against papers (Wadden et al., 2020). Our workflow targets the upstream stage of *forming* the question, before retrieval systems generate answers. Section 5.3 evaluates whether Lacuna's compact evidence can also support citation-backed synthesis on ScholarQA-CS-ML.

Literature-based discovery. The notion of research problem formulation targeted here overlaps with literature-based discovery, which studies methods for finding implicit hypotheses in scientific text (Thilakaratne et al., 2019). Lacuna's research proposal pages are produced using Alien Science sampling (Artiles et al., 2026); we treat them as hypotheses for researcher review.

3. Task and the Lacuna Map

We target **research problem formulation**: given a seed query, produce a scoped research question, a small number of supporting observations, and a limitation statement in the form a researcher could circulate to a collaborator. Lacuna supports formulation by precomputing research directions and research proposals; the LLM's role is to choose, connect, and audit these for the researcher.

Lacuna Contents. Lacuna contains several generated page and record types. A *core-idea paper summary* is a markdown document generated from the paper text and metadata to capture its main technical ideas. A *paper summary with figures* is a compact paper summary that combines text with extracted and selected figures. A *concept element* is a one- or two-sentence statement of a core idea, method, limitation, or empirical observation extracted from a core-idea paper summary. Concept elements are embedded, linked to papers, and clustered. A *research direction* is an approximately two-page synthesis of a cluster of concept elements into a recurring problem, method family, or opportunity area. *Author* and *venue* profiles summarize researchers and publication venues from metadata plus linked papers and directions. A *research proposal* is a generated hypothesis sampled from research directions and supporting papers. The production map used Gemini-

family LLMs, with core-idea paper summaries, concept elements, research directions, and profiles generated with `gemini-flash-3.1-preview` and research proposals generated with `gemini-3.1-pro-preview` (Gemini Team, 2026).

Lacuna Interface. Lacuna exposes four interfaces: RRF (reciprocal-rank-fusion) search over research directions, papers, and authors; `/md` pages with 5–30 KB of cleaned markdown; typed links among map pages; and an MCP artifact documenting the page schema for LLM tools. Section 4 and Appendix D describe construction; Table 1 reports the live map scale.

Table 1. Dataset size. This table shows the scale of the Lacuna knowledge graph.

Quantity	Count
Papers catalogued	733,795
Author pages	176,661
Research direction pages	27,017
Research proposal pages	38,000
Venue pages	4,183
Institution pages	5,276
Research direction–paper relation rows	6,188,727
Research direction–author relation rows	8,554,741

4. Building Lacuna

Lacuna is built by a pipeline. It harvests and reconciles records from OpenAlex, OpenReview, DBLP, arXiv, and venue open-access pages, then attaches paper files or URL replicas to stable paper, author, and venue records. This stage establishes the identifiers that later generated content refers to. Further details available in Appendix D.

Within the catalog stage, Lacuna treats author bibliography construction as an authorship-anchored expansion problem. We use OpenReview as the primary identity anchor because its profiles are author-maintained and provide a cleaner signal than external author identifiers alone, which can conflate researchers with shared names. For each OpenReview-profiled author, the pipeline harvests that author's OpenReview papers, canonicalizes OpenReview profile aliases, and builds an OpenReview-only co-author neighborhood. External sources such as DBLP and OpenAlex are used as recall sources, not as authorities. Papers obtained through an author's external identifiers are retained for that author's bibliography only when they either reconcile to an existing OpenReview paper or when their author lists overlap the author's trusted OpenReview co-author neighborhood. This gives Lacuna broader bibliographies than OpenReview alone while reducing the risk that external-source disambiguation errors propagate into author profiles, venue pages, research-direction memberships, and generated proposals.

LLM calls then turn papers and metadata into reusable summaries, concepts, profiles, and proposals. From paper text and metadata, the pipeline generates core-idea paper summaries; from figures extracted with RT-DETR (Zhao et al., 2024) and selected for paper context, it generates paper summaries with figures. Concept-element extraction converts these summaries into one- or two-sentence units that capture methods, observations, limitations, and empirical findings. Author and venue profiles are generated from metadata plus linked paper and research direction context.

The graph-building stages embed concept elements, cluster them with HDBSCAN (McInnes et al., 2017) into research directions, and generate research proposal pages from research directions and supporting papers. Serving stages then build search indexes, relation tables, HTML pages, markdown variants, and MCP access. Each generated item keeps links to the papers, metadata records, concept elements, and research directions that produced it, which lets a researcher inspect evidence from a research proposal back to the underlying paper pages. Appendix D gives the full stage table and a paper-to-direction subpipeline diagram.

5. Evaluation

We evaluate whether Lacuna supports three operations: formulating a research question from a seed idea, auditing generated claims against source pages, and searching over literature to answer scientific questions.

We test these operations in three parts. First, we follow a fixed, reproducible run through Lacuna from a theorem-proving seed idea to a generated research proposal. Second, we audit the claims used in that run and compare time and cost against a sequential-PDF baseline on the same task. Third, we evaluate Lacuna's concise evidence outside the run on ScholarQA-CS-ML, a 42-question ML/AI subset of the ScholarQA-CS benchmark introduced with OpenScholar (Asai et al., 2026).

5.1. From the Seed of an Idea to a Research Proposal

We instantiate the run on the seed idea *automated theorem proving with proof assistants*. The run uses Lacuna's hosted markdown pages, which can be reached through the reviewer MCP artifact[2] or through direct HTTP `GET` requests. The full prompt, transcript, and machine-readable summaries are reproduced in Appendices A, B, and F.

The run visits two research direction pages, four paper pages, and one research proposal page. It starts at a research direction about the autoformalization gap between informal mathematical reasoning and proof-assistant verifi-

[2]Reviewer MCP artifact for querying the hosted map: `https://anonymous.4open.science/r/research-map-mcp-artifact-20260513-600C/README.md`.

cation, follows a neighboring research direction about iterative compiler feedback, and then inspects paper pages for ProofNet (Azerbayev et al., 2023), FIMO (Liu et al., 2023), StepFun-Prover (Shang et al., 2025), and VERINA (Ye et al., 2025). These papers provide quantitative anchors: low Codex success on ProofNet, GPT-4's 0% proof success on FIMO, StepFun-Prover's 70.0% pass@1 on miniF2F, and VERINA's gap between code generation and proof generation. The run ends at a research proposal page that turns this local neighborhood into a hypothesis about compiler-feedback granularity.

The synthesized final question is: *at what granularity of compiler feedback — tactic-block, subgoal, or token-level — does iterative LLM-REPL interaction most efficiently separate fixable translation errors from fundamental deductive gaps?* The run therefore turns the broad seed idea into a concrete, inspectable problem formulation: a specific independent variable, a mechanism to test, and a limitation of current one-shot and iterative proof-generation methods.

5.2. Grounding, Time, and Cost

We audit the claims used by the run against the markdown pages it cites, marking each as supported by a cited paper page, a research proposal hypothesis, a research direction limitation, or unsupported. The audit covers five claims: two are supported by cited paper pages, one is a research direction limitation, one is a research proposal hypothesis correctly treated as a proposed mechanism rather than a reported result, and one is a deliberately included false claim — that the route is perfectly clean — that Lacuna's paper pages and relation tabs let a reader catch and correct. The full per-claim audit with evidence quotes is in Appendix C.

We also compare the time and cost of consuming Lacuna pages with using raw PDFs for the same theorem-proving task. The sequential-PDF baseline has no access to Lacuna; it downloads four papers, extracts their first ten pages with `pdftotext`, writes a 650–900 word note per paper, and only then synthesizes the final question. Table 2 reports the resulting cost and tool use. This fixed run shows work that Lacuna avoids at inference time: paper notes and map context are served as existing content instead of being regenerated inside the run.

5.3. ScholarQA-CS-ML Evidence Quality

The third evaluation asks whether Lacuna's research map is useful for scientific question answering beyond the fixed problem-formulation run. We evaluate this on ScholarQA-CS-ML, a 42-question ML/AI subset of ScholarQA-CS. ScholarQA-CS is the 100-question computer-science literature-synthesis subset of ScholarQABench, introduced with OpenScholar in *Nature* (Asai et al., 2026); its questions were written by PhD-level domain experts and scored against detailed literature-grounded rubrics. We classified the full 100-prompt set and evaluated every prompt marked ML/AI: 42 prompts, or 42% of ScholarQA-CS, with no score-based filtering. Appendix E gives the subset-selection details and lists all questions in ScholarQA-CS-ML.

Table 3 reports the same-question comparison: Lacuna-GPT-4o scores 0.694, compared with 0.672 for the released OpenScholar-GPT-4o answer file. The gain is concentrated in content and evidence quality: Lacuna-GPT-4o is higher on rubric-item coverage, judged expertise, excerpt availability, and both must-have and nice-to-have rubric items. OpenScholar-GPT-4o remains higher on structural citation coverage, which measures whether claims carry inline bracket citations; it does not measure source entailment.

Table 2. Time, tool use, and cost for the fixed theorem-proving run, same model and task.

	Lacuna `/md`	PDF baseline
Wall time (s)	85.5	289.2
Claude turns	8	28
Tool calls	7	27
Tool text read	41.5 KB	167.3 KB
Output tokens	4,919	14,135
Blocking model calls	1	6
Cost (USD)	$0.171	$0.695

Table 3. ScholarQA-CS-ML literature-synthesis evaluation, covering 42 of the 100 ScholarQA-CS prompts. Scores are averages under the ScholarQABench rubric judge; OpenScholar-GPT-4o is the released answer + references condition, and Lacuna-GPT-4o is answer + cited evidence.

Metric	OpenScholar	Lacuna
Overall	0.672	**0.694**
Rubric items	0.607	**0.638**
Expertise	0.826	**0.879**
Citation coverage	**0.956**	0.924
Excerpts	0.747	**0.826**
Must-have items	0.577	**0.615**
Nice-to-have items	0.455	**0.469**

6. Conclusion

Lacuna shows how a served research map can support machine learning problem formulation. Its linked summaries, concepts, directions, profiles, and proposals let researchers and agents move from broad seed ideas to inspectable research proposals while keeping claims tied to source pages.

The current map is batch-built for ML, so new papers require rerunning pipeline stages and other fields will need additional source coverage and quality checks. Its upfront build costs are substantial, but they amortize across later navigation, page fetches, and agent runs.

References

Artiles, A. H., Weiss, M., Brinkmann, L., Goyal, A., and Rahaman, N. Alien science: Sampling coherent but cognitively unavailable research directions from idea atoms, 2026. URL https://arxiv.org/abs/2603.01092.

Asai, A., He, J., Shao, R., Shi, W., Singh, A., Chang, J. C., Lo, K., Soldaini, L., Feldman, S., D'Arcy, M., Wadden, D., Latzke, M., Sparks, J., Hwang, J. D., Kishore, V., Tian, M., Ji, P., Liu, S., Tong, H., Wu, B., Xiong, Y., Zettlemoyer, L., Neubig, G., Weld, D. S., Downey, D., Yih, W.-t., Koh, P. W., and Hajishirzi, H. Synthesizing scientific literature with retrieval-augmented language models. *Nature*, 650:857–863, 2026. doi: 10.1038/s41586-025-10072-4. URL https://doi.org/10.1038/s41586-025-10072-4.

Azerbayev, Z., Piotrowski, B., Schoelkopf, H., Ayers, E. W., Radev, D., and Avigad, J. ProofNet: Autoformalizing and formally proving undergraduate-level mathematics, 2023. URL https://arxiv.org/abs/2302.12433.

Cohan, A., Feldman, S., Beltagy, I., Downey, D., and Weld, D. S. SPECTER: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020. URL https://arxiv.org/abs/2004.07180.

Gemini Team. Gemini 3.1 Pro: A smarter model for your most complex tasks. Google Blog, 2026. URL https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-pro/.

Jaradeh, M. Y., Oelen, A., Farfar, K. E., Prinz, M., D'Souza, J., Kismihók, G., Stocker, M., and Auer, S. Open research knowledge graph: Next generation infrastructure for semantic scholarly knowledge, 2019. URL https://arxiv.org/abs/1901.10816.

Lála, J., O'Donoghue, O., Shtedritski, A., Cox, S., Rodriques, S. G., and White, A. D. PaperQA: Retrieval-augmented generative agent for scientific research, 2023. URL https://arxiv.org/abs/2312.07559.

Liu, C., Shen, J., Xin, H., Liu, Z., Yuan, Y., Wang, H., Ju, W., Zheng, C., Yin, Y., Li, L., Zhang, M., and Liu, Q. FIMO: A challenge formal dataset for automated theorem proving, 2023. URL https://arxiv.org/abs/2309.04295.

Lo, K., Wang, L. L., Neumann, M., Kinney, R., and Weld, D. S. S2ORC: The semantic scholar open research corpus, 2020. URL https://arxiv.org/abs/1911.02782.

McInnes, L., Healy, J., and Astels, S. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11):205, 2017. doi: 10.21105/joss.00205. URL https://doi.org/10.21105/joss.00205.

Pirolli, P. and Card, S. K. Information foraging. *Psychological Review*, 106(4):643–675, 1999. doi: 10.1037/0033-295X.106.4.643.

Priem, J., Piwowar, H., and Orr, R. Openalex: A fully-open index of scholarly works, authors, venues, institutions, and concepts, 2022. URL https://arxiv.org/abs/2205.01833.

Shang, S., Wan, R., Peng, Y., Wu, Y., Chen, X.-h., Yan, J., and Zhang, X. StepFun-Prover preview: Let's think and verify step by step, 2025. URL https://arxiv.org/abs/2507.20199.

Thilakaratne, M., Falkner, K., and Atapattu, T. A systematic review on literature-based discovery workflow. *PeerJ Computer Science*, 5:e235, 2019. doi: 10.7717/peerj-cs.235. URL https://doi.org/10.7717/peerj-cs.235.

Wadden, D., Lin, S., Lo, K., Wang, L. L., van Zuylen, M., Cohan, A., and Hajishirzi, H. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020. URL https://arxiv.org/abs/2004.14974.

Ye, Z., Yan, Z., He, J., Kasriel, T., Yang, K., and Song, D. VERINA: Benchmarking verifiable code generation, 2025. URL https://arxiv.org/abs/2505.23135.

Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., Liu, Y., and Chen, J. DETRs beat YOLOs on real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2304.08069.

Acknowledgements

Martin Weiss acknowledges support from IVADO through its postdoctoral entrepreneur program. Lacuna builds on public scholarly infrastructure, including OpenReview, OpenAlex, DBLP, and arXiv; we thank the maintainers of these services for making large-scale research discovery systems possible.

A. Full Agent and Baseline Prompts

The prompts below are reproduced verbatim from the supplementary archive. They can also be found at `agent_workflow_eval/lacuna_navigation_prompt.md` and `agent_workflow_eval/pdf_to_chat_baseline_prompt.md`.

A.1. Lacuna Navigation Prompt

```
# Condition A: Lacuna Wiki Navigation

You are running a timed research-navigation trace for a paper evaluation.

## Task

Starting from the topic `automated theorem proving proof assistants`, use Lacuna markdown wiki pages to produce:

1. one scoped research question;
2. three supporting source-linked observations;
3. one limitation or failure mode;
4. a concise route summary.

## Allowed Source

Use only pages under `http://34.8.208.118`, and use the `/md/render/...` reviewer routes. Do not use external
     search engines, source PDFs, arXiv pages, or non-Lacuna paper pages.

## Navigation Instructions

Use Bash with `curl -sS` to fetch pages. Keep each fetch bounded by piping through `head -c 30000` when the page
     may be long. Do not write files.

Start here:

`http://34.8.208.118/md/render/search?q=automated%20theorem%20proving%20proof%20assistants`

If the search HTML is too noisy for a command-line trace, continue with the known search result:

`http://34.8.208.118/md/render/direction/the-autoformalization-gap-in-theorem-proving-23293`

Required route unless a page is unreachable:

- `http://34.8.208.118/md/render/direction/the-autoformalization-gap-in-theorem-proving-23293`
- `http://34.8.208.118/md/render/direction/iterative-compiler-feedback-for-formal-theorem-proving-12169`
- `http://34.8.208.118/md/render/paper/proofnet-autoformalizing-and-formally-proving-undergraduate-level-
     mathematics/art_f5a5f3551f6641598e578328d5771b3b`
- `http://34.8.208.118/md/render/paper/fimo-a-challenge-formal-dataset-for-automated-theorem-proving/
     art_d65417c966cc4525ab8bd63248394361`
- `http://34.8.208.118/md/render/paper/stepfun-prover-preview-let-s-think-and-verify-step-by-step/
     art_804f191b837940a1a2b73568d30d29b0`
- `http://34.8.208.118/md/render/paper/verina-benchmarking-verifiable-code-generation/
     art_ae1f0fb500a44279ba999f4c6df4ed6e`
- `http://34.8.208.118/md/render/hypothesis/token-level-alignment-of-informal-mathematics-to-formal-compiler-
     states-5f6a6e04c0bdbac4`

## Output Format

Return only valid JSON with these fields:

```json
{
  "condition": "lacuna_navigation",
  "route": [{"url": "...", "reason": "..."}],
  "pages_read_count": 7,
  "branch_decisions": ["..."],
  "source_checks": [{"claim": "...", "checked_against": "...", "status": "supported|proposal_level|limitation|
     unclear"}],
  "final_question": "...",
  "observations": ["...", "...", "..."],
  "limitation": "...",
  "blocking_model_calls_needed": 1,
  "notes": "..."
}
```

`blocking_model_calls_needed` should count the current Claude Code run as one blocking model call. Do not count
     local page fetches as model calls.
```

A.2. Sequential PDF-to-Chat Baseline Prompt

```
# Condition B: Sequential PDF-to-Chat Baseline

You are running a timed baseline trace for a paper evaluation.

## Task

Starting from the topic `automated theorem proving proof assistants`, produce the same output as the Lacuna
     condition:

1. one scoped research question;
2. three supporting source-linked observations;
3. one limitation or failure mode;
4. a concise route summary.

## Baseline Constraint

Do not use Lacuna pages, Lacuna `/md` routes, or Lacuna-generated summaries.

This baseline approximates a PDF-to-chat workflow where the agent must summarize each academic paper in turn
     before deciding whether it helps the route. To make the run reproducible, use the following fixed source-
     paper order. For each item:

1. fetch the public source paper or source article;
2. for arXiv PDFs, extract and read the first 10 pages with `pdftotext`;
3. for the StepFun source article, extract as much main article text as is available from the public page;
4. write a Lacuna-style markdown note before moving to the next source;
5. decide whether it supports the emerging research question;
6. only then move to the next item.

Use Bash with `curl -L -sS`, `pdftotext`, and ordinary shell tools. You may write temporary files in the current
     working directory. Keep extracted text bounded with `head -c 50000`.

## Source-Paper Order

- ProofNet PDF: `https://arxiv.org/pdf/2302.12433`
- FIMO PDF: `https://arxiv.org/pdf/2309.04295`
- StepFun-Prover source article: `https://stepfun.ai/research/en/stepfun-prover-preview`
- VERINA PDF: `https://arxiv.org/pdf/2505.23135`

If a source page or PDF is unavailable, record the failure and continue to the next source page. Do not replace it
      with a Lacuna page.

For each arXiv PDF, use a command of this form before summarizing it:

```bash
curl -L -sS -o proofnet.pdf https://arxiv.org/pdf/2302.12433
pdftotext -f 1 -l 10 -layout proofnet.pdf - | head -c 50000
```

After reading each source, immediately create a markdown note with a Bash here-doc before fetching the next source.
      Use these filenames:

- `proofnet_note.md`
- `fimo_note.md`
- `stepfun_note.md`
- `verina_note.md`

Each note should be 650-900 words and structured like a compact Lacuna paper page:

```markdown
# Paper Title

## Summary
...

## Method / System
...

## Key Results
...

## Limitations And Open Questions
...
```

Do not include the full note text in the final JSON; the trace will capture that you generated it.

Output Format
```

```
Return only valid JSON with these fields:

```json
{
  "condition": "pdf_to_chat_baseline",
  "route": [{"url": "...", "reason": "..."}],
  "pages_read_count": 4,
  "notes_written": [{"paper": "...", "file": "...", "approx_words": 750, "decision": "use|skip|unclear"}],
  "source_checks": [{"claim": "...", "checked_against": "...", "status": "supported|proposal_level|limitation|
     unclear"}],
  "final_question": "...",
  "observations": ["...", "...", "..."],
  "limitation": "...",
  "blocking_model_calls_needed": 6,
  "notes": "..."
}
```

`blocking_model_calls_needed` should be a conservative estimate for an ordinary sequential PDF-to-chat workflow:
     one call to plan plus one call for each paper summary before final synthesis.
```

B. Full Conversation Transcript

We ran both conditions once on 2026-05-08 with Claude Sonnet via Claude Code in non-interactive mode. The transcript below is the sanitized run log generated by the harness, including the measured summary, tool commands, and final JSON outputs. The same file is released in the supplementary archive at `agent_workflow_eval/outputs/2026-05-08_pdf10_notes_baseline/results.md`; the corresponding machine-readable summaries are `lacuna_navigation.summary.json` and `pdf_to_chat_baseline.summary.json` in the same directory.

```
# Claude Code Workflow Evaluation

Run directory: `outputs/2026-05-08_pdf10_notes_baseline`

This is an operational trace, not a human-subjects study. Both conditions used Claude Code in non-interactive mode
      with the same model and a fixed task.

## Summary Table

| Condition | Wall time | Claude turns | Tool calls | Tool text | Output tokens | Estimated blocking model calls |
      Cost | Outcome |
|---|---:|---:|---:|---:|---:|---:|---:|---|
| `lacuna_navigation` | 85.5s | 8 | 7 | 41.5KB | 4919 | 1 | $0.171 | At what granularity of compiler feedback ---
     tactic-block, subgoal, or token-level --- does iterative LLM-REPL interactio... |
| `pdf_to_chat_baseline` | 289.2s | 28 | 27 | 167.3KB | 14135 | 6 | $0.695 | Does integrating real-time formal
     verifier feedback into the LLM training loop via reinforcement learning---rather than... |

## Tool Commands

### lacuna_navigation

- `curl -sS "http://34.8.208.118/md/render/direction/the-autoformalization-gap-in-theorem-proving-23293" | head -c
      30000`
- `curl -sS "http://34.8.208.118/md/render/direction/iterative-compiler-feedback-for-formal-theorem-proving-12169"
      | head -c 30000`
- `curl -sS "http://34.8.208.118/md/render/paper/proofnet-autoformalizing-and-formally-proving-undergraduate-level
     -mathematics/art_f5a5f3551f6641598e578328d5771b3b" | head -c 30000`
- `curl -sS "http://34.8.208.118/md/render/paper/fimo-a-challenge-formal-dataset-for-automated-theorem-proving/
     art_d65417c966cc4525ab8bd63248394361" | head -c 30000`
- `curl -sS "http://34.8.208.118/md/render/paper/stepfun-prover-preview-let-s-think-and-verify-step-by-step/
     art_804f191b837940a1a2b73568d30d29b0" | head -c 30000`
- `curl -sS "http://34.8.208.118/md/render/paper/verina-benchmarking-verifiable-code-generation/
     art_ae1f0fb500a44279ba999f4c6df4ed6e" | head -c 30000`
- `curl -sS "http://34.8.208.118/md/render/hypothesis/token-level-alignment-of-informal-mathematics-to-formal-
     compiler-states-5f6a6e04c0bdbac4" | head -c 30000`

### pdf_to_chat_baseline

- `curl -L -sS -o /private/tmp/lacuna_workflow_eval/proofnet.pdf https://arxiv.org/pdf/2302.12433 && echo "
     Downloaded proofnet.pdf"`
- `pdftotext -f 1 -l 10 -layout /private/tmp/lacuna_workflow_eval/proofnet.pdf - | head -c 50000`
- `pdftotext -f 1 -l 10 -layout /private/tmp/lacuna_workflow_eval/proofnet.pdf - | head -c 50000 > /private/tmp/
     lacuna_workflow_eval/proofnet_text.txt && wc -c /private/tmp/lacuna_workflow_eval/proofnet_text.txt`
- `cat /private/tmp/lacuna_workflow_eval/proofnet_text.txt`
- `sed -n '1,200p' /private/tmp/lacuna_workflow_eval/proofnet_text.txt`
- `sed -n '200,400p' /private/tmp/lacuna_workflow_eval/proofnet_text.txt`
```

```
- `cat > /private/tmp/lacuna_workflow_eval/proofnet_note.md << 'EOF'
<omitted 4985 chars>`
- `cat >> /private/tmp/lacuna_workflow_eval/proofnet_note.md << 'EOF'
<omitted 649 chars>`
- `curl -L -sS -o /private/tmp/lacuna_workflow_eval/fimo.pdf https://arxiv.org/pdf/2309.04295 && echo "Downloaded
    fimo.pdf"`
- `pdftotext -f 1 -l 10 -layout /private/tmp/lacuna_workflow_eval/fimo.pdf - | head -c 50000`
- `pdftotext -f 1 -l 10 -layout /private/tmp/lacuna_workflow_eval/fimo.pdf - | head -c 50000 > /private/tmp/
    lacuna_workflow_eval/fimo_text.txt && sed -n '1,350p' /private/tmp/lacuna_workflow_eval/fimo_text.txt`
- `sed -n '1,150p' /private/tmp/lacuna_workflow_eval/fimo_text.txt 2>/dev/null || head -150 /private/tmp/
    lacuna_workflow_eval/fimo_text.txt`
- `sed -n '150,350p' /private/tmp/lacuna_workflow_eval/fimo_text.txt`
- `sed -n '350,500p' /private/tmp/lacuna_workflow_eval/fimo_text.txt`
- `cat > /private/tmp/lacuna_workflow_eval/fimo_note.md << 'EOF'
<omitted 5619 chars>`
- `curl -L -sS "https://stepfun.ai/research/en/stepfun-prover-preview" | head -c 50000`
- `curl -L -sS "https://stepfun.ai/research/en/stepfun-prover-preview" | python3 -c "
import sys, re, html
content = sys.stdin.read()
# Remove script and style tags
content = re.sub(r'<script[^>]*>.*?</script>', '', content, flags=re.DOTALL)
content = re.sub(r'<style[^>]*>.*?</style>', '', content, flags=re.DOTALL)
# Extract text from HTML
text = re.sub(r'<[^>]+>', ' ', content)
text = html.unescape(text)
text = re.sub(r'\s+', ' ', text).strip()
print(text[:20000])
"`
- `cat > /private/tmp/lacuna_workflow_eval/stepfun_note.md << 'EOF'
<omitted 6149 chars>`
- `curl -L -sS -o /private/tmp/lacuna_workflow_eval/verina.pdf https://arxiv.org/pdf/2505.23135 && echo "
    Downloaded verina.pdf"`
- `pdftotext -f 1 -l 10 -layout /private/tmp/lacuna_workflow_eval/verina.pdf - | head -c 50000`
- `pdftotext -f 1 -l 10 -layout /private/tmp/lacuna_workflow_eval/verina.pdf - | head -c 50000 > /private/tmp/
    lacuna_workflow_eval/verina_text.txt && sed -n '1,350p' /private/tmp/lacuna_workflow_eval/verina_text.txt`
- `sed -n '350,550p' /private/tmp/lacuna_workflow_eval/verina_text.txt`
- `sed -n '550,720p' /private/tmp/lacuna_workflow_eval/verina_text.txt`
- `sed -n '720,850p' /private/tmp/lacuna_workflow_eval/verina_text.txt`
- `wc -l /private/tmp/lacuna_workflow_eval/verina_text.txt && tail -100 /private/tmp/lacuna_workflow_eval/
    verina_text.txt`
- `cat > /private/tmp/lacuna_workflow_eval/verina_note.md << 'EOF'
<omitted 6450 chars>`
- `ls -la /private/tmp/lacuna_workflow_eval/*.md /private/tmp/lacuna_workflow_eval/*.pdf 2>/dev/null`

## Final Outputs

### lacuna_navigation

```json
{
  "condition": "lacuna_navigation",
  "route": [
    {
      "url": "http://34.8.208.118/md/render/direction/the-autoformalization-gap-in-theorem-proving-23293",
      "reason": "Entry point: establishes the core autoformalization problem \u2014 LLMs have strategic intuition
     but fail formal compiler precision; introduced key concepts of granularity mismatch, neuro-symbolic bridges,
     and semantic misalignment."
    },
    {
      "url": "http://34.8.208.118/md/render/direction/iterative-compiler-feedback-for-formal-theorem-proving
     -12169",
      "reason": "Second direction page: details the REPL-based closed-loop feedback paradigm as the dominant
     response to one-shot fragility; introduces the 'semantic wall' limitation where LLMs thrash rather than
     converge on hard problems."
    },
    {
      "url": "http://34.8.208.118/md/render/paper/proofnet-autoformalizing-and-formally-proving-undergraduate-
     level-mathematics/art_f5a5f3551f6641598e578328d5771b3b",
      "reason": "Empirical grounding: ProofNet benchmark (371 undergraduate-level problems); Codex baseline
     formalization 13.4%, retrieval-augmented 16.1%; documented implicit-hypothesis failure mode (inner product
     space example)."
    },
    {
      "url": "http://34.8.208.118/md/render/paper/fimo-a-challenge-formal-dataset-for-automated-theorem-proving/
     art_d65417c966cc4525ab8bd63248394361",
      "reason": "Difficulty ceiling: 149 IMO-Shortlist problems in Lean; GPT-4 achieves 0% proof success;
     iterative feedback pipeline raises formalization success from 32.6% to 60.8%, establishing the gap between
     formalization and proof generation."
    },
    {
      "url": "http://34.8.208.118/md/render/paper/stepfun-prover-preview-let-s-think-and-verify-step-by-step/
```
```

```
    art_804f191b837940a1a2b73568d30d29b0",
      "reason": "State-of-the-art result: StepFun-Prover-32B achieves 70.0% pass@1 on miniF2F via GRPO RL with
    Lean 4 REPL tool integration, outperforming 671B models \u2014 demonstrates feedback quality dominates
    parameter count."
    },
    {
      "url": "http://34.8.208.118/md/render/paper/verina-benchmarking-verifiable-code-generation/
    art_ae1f0fb500a44279ba999f4c6df4ed6e",
      "reason": "Critical bottleneck quantification: o3 achieves 72.6% CodeGen but only 4.9% ProofGen; end-to-end
    (code+spec+proof) success is 3.2%; 64-round iterative refinement only reaches ~20% ProofGen at massive
    compute cost."
    },
    {
      "url": "http://34.8.208.118/md/render/hypothesis/token-level-alignment-of-informal-mathematics-to-formal-
    compiler-states-5f6a6e04c0bdbac4",
      "reason": "Proposed solution: token-to-compiler-state alignment using CTC-inspired contrastive loss and
    Dynamic Time Warping to address credit assignment in iterative proof search; connects speech alignment
    methods to theorem proving."
    }
  ],
  "pages_read_count": 7,
  "branch_decisions": [
    "Followed the mandatory required route without deviation \u2014 no branching was available or needed.",
    "Fetched pages in two parallel batches (2 then 5) to minimize wall-clock time while respecting sequential data
      availability.",
    "Used head -c 30000 on all fetches to bound output size; all pages returned well under the limit so no content
      was truncated in a loss-critical way."
  ],
  "source_checks": [
    {
      "claim": "Codex achieves 13.4% baseline formalization accuracy on ProofNet, rising to 16.1% with prompt
    retrieval",
      "checked_against": "http://34.8.208.118/md/render/paper/proofnet-autoformalizing-and-formally-proving-
    undergraduate-level-mathematics/art_f5a5f3551f6641598e578328d5771b3b \u2014 Key Results section",
      "status": "supported"
    },
    {
      "claim": "GPT-4 achieves 0% proof success on FIMO IMO-level problems even when provided the informal proof
    as a guide",
      "checked_against": "http://34.8.208.118/md/render/paper/fimo-a-challenge-formal-dataset-for-automated-
    theorem-proving/art_d65417c966cc4525ab8bd63248394361 \u2014 Key Insights section",
      "status": "supported"
    },
    {
      "claim": "StepFun-Prover-Preview-32B achieves 70.0% pass@1 on miniF2F, outperforming DeepSeek-Prover-V2-671B
      at 61.9%",
      "checked_against": "http://34.8.208.118/md/render/paper/stepfun-prover-preview-let-s-think-and-verify-step-
    by-step/art_804f191b837940a1a2b73568d30d29b0 \u2014 Experimental Results section",
      "status": "supported"
    },
    {
      "claim": "o3 achieves 72.6% CodeGen but only 4.9% ProofGen; end-to-end success drops to 3.2%; 64-round
    refinement reaches ~20% ProofGen",
      "checked_against": "http://34.8.208.118/md/render/paper/verina-benchmarking-verifiable-code-generation/
    art_ae1f0fb500a44279ba999f4c6df4ed6e \u2014 Key Insights section",
      "status": "supported"
    },
    {
      "claim": "Token-level alignment via DTW and contrastive loss over compiler state deltas can solve the credit
      assignment problem in iterative proof search",
      "checked_against": "http://34.8.208.118/md/render/hypothesis/token-level-alignment-of-informal-mathematics-
    to-formal-compiler-states-5f6a6e04c0bdbac4 \u2014 Core Claim and Proposed Experiments",
      "status": "proposal_level"
    },
    {
      "claim": "FIMO iterative-feedback pipeline raises formalization success from 32.6% to 60.8%",
      "checked_against": "http://34.8.208.118/md/render/paper/fimo-a-challenge-formal-dataset-for-automated-
    theorem-proving/art_d65417c966cc4525ab8bd63248394361 \u2014 Methodology section",
      "status": "supported"
    },
    {
      "claim": "Iterative refinement causes thrashing rather than convergence when the model lacks underlying
    deductive capability",
      "checked_against": "http://34.8.208.118/md/render/direction/iterative-compiler-feedback-for-formal-theorem-
    proving-12169 \u2014 The Semantic Wall section; corroborated by VERINA paper",
      "status": "limitation"
    }
  ],
  "final_question": "At what granularity of compiler feedback \u2014 tactic-block, subgoal, or token-level \u2014
    does iterative LLM-REPL interaction most efficiently separate fixable translation errors from fundamental
    deductive gaps, and can token-to-compiler-state alignment (e.g., via DTW and contrastive loss on state deltas
```

```
    ) measurably reduce thrashing on IMO-level benchmarks like FIMO?",
  "observations": [
    "Formalization and proof generation are sharply separable tasks with different difficulty ceilings: FIMO shows
     iterative LLM feedback raises formalization success from 32.6% to 60.8%, yet GPT-4 still achieves 0% proof
    success on the same IMO problems, and VERINA confirms this asymmetry at scale \u2014 o3 scores 72.6% on
    CodeGen but only 4.9% on ProofGen (sources: FIMO paper /md; VERINA paper /md).",
    "Tool-integrated RL with REPL feedback dominates raw parameter count: StepFun-Prover-Preview-32B achieves
    70.0% pass@1 on miniF2F by training on live Lean 4 error trajectories via GRPO, outperforming DeepSeek-Prover-
    V2-671B (61.9%), suggesting that the quality of feedback integration \u2014 not model scale \u2014 is the
    current performance frontier (source: StepFun-Prover paper /md).",
    "Implicit mathematical hypotheses are a structural failure mode that compiler feedback cannot resolve:
    ProofNet documents that models consistently fail to infer unstated prerequisites required by formal libraries
     (e.g., that an orthogonal complement presupposes an inner product space), because these assumptions are
    present in human shared context but absent from both the problem prompt and the compiler's error messages (
    source: ProofNet paper /md; autoformalization gap direction /md)."
  ],
  "limitation": "Iterative compiler feedback exhibits diminishing returns and eventual thrashing on problems that
    exceed the model's deductive capability: VERINA shows that 64 rounds of Lean feedback raise ProofGen success
    only from ~5% to ~20% at massive token and inference cost, while the iterative compiler feedback direction
    page frames this as a 'semantic wall' \u2014 the compiler's binary error signals lack the semantic content
    needed to guide logical recovery, causing models to cycle through different incorrect proofs rather than
    converging. This means the technique is largely a surface-error fixer and cannot substitute for missing
    mathematical reasoning depth.",
  "blocking_model_calls_needed": 1,
  "notes": "All 7 required pages were reachable and returned full markdown content. Pages were fetched in two
    parallel Bash batches (2 + 5) within a single Claude Code session. No external search engines, PDFs, or non-
    Lacuna sources were consulted. The token-level alignment hypothesis page is categorized as proposal-level: it
     describes a proposed method with planned experiments, not yet empirically validated results. VERINA is an
    ICLR 2026 paper, consistent with the current date of 2026-05-08."
}
```

pdf_to_chat_baseline

```json
{
  "condition": "pdf_to_chat_baseline",
  "route": [
    {
      "url": "https://arxiv.org/pdf/2302.12433",
      "reason": "ProofNet establishes the benchmark landscape for autoformalization and formal proving at
    undergraduate-mathematics level, revealing that parallel data scarcity is the foundational bottleneck (Codex
    13.4% accuracy) and motivating the research question about what training-signal mechanisms can close this gap
    ."
    },
    {
      "url": "https://arxiv.org/pdf/2309.04295",
      "reason": "FIMO shows that IMO-shortlisted problems define a qualitative difficulty cliff: GPT-4 achieves 0%
     Pass@8 on all 149 problems despite 21.3% on miniF2F, demonstrating that raw LLM capability without richer
    verifier interaction cannot bridge the gap to hard mathematical reasoning."
    },
    {
      "url": "https://stepfun.ai/research/en/stepfun-prover-preview",
      "reason": "StepFun-Prover-Preview provides the current SOTA result (70% pass@1 on miniF2F-test with a 32B
    model, surpassing a 671B competitor) and makes the causal argument: free-form, iterative Lean REPL
    interaction during both training (tool-integrated RL via GRPO) and inference is the key architectural lever,
    not model scale."
    },
    {
      "url": "https://arxiv.org/pdf/2505.23135",
      "reason": "VERINA extends the proof-generation bottleneck finding to software verification: even the best
    general-purpose LLM (o3) achieves only 4.9% proof success on structured coding tasks, and iterative
    refinement with Lean feedback raises this to only 20.1% at 64 steps\u2014confirming the bottleneck
    generalizes beyond mathematical benchmarks."
    }
  ],
  "pages_read_count": 4,
  "notes_written": [
    {
      "paper": "ProofNet: Autoformalizing and Formally Proving Undergraduate-Level Mathematics",
      "file": "proofnet_note.md",
      "approx_words": 713,
      "decision": "use"
    },
    {
      "paper": "FIMO: A Challenge Formal Dataset for Automated Theorem Proving",
      "file": "fimo_note.md",
      "approx_words": 763,
      "decision": "use"
    },
    {
```

```
      "paper": "StepFun-Prover Preview: Let's Think and Verify Step by Step",
      "file": "stepfun_note.md",
      "approx_words": 798,
      "decision": "use"
    },
    {
      "paper": "VERINA: Benchmarking Verifiable Code Generation",
      "file": "verina_note.md",
      "approx_words": 853,
      "decision": "use"
    }
  ],
  "source_checks": [
    {
      "claim": "Code-davinci-002 achieves 13.4% autoformalization accuracy on ProofNet; prompt retrieval raises it
     to 16.1% with a +21.5 percentage-point typecheck rate improvement",
      "checked_against": "proofnet_note.md / Table 3 in arXiv:2302.12433",
      "status": "supported"
    },
    {
      "claim": "GPT-4 achieves 0% Pass@1 and 0% Pass@8 on all 149 FIMO problems, compared to 21.3% Pass@8 on
    miniF2F validation set",
      "checked_against": "fimo_note.md / Table 9 in arXiv:2309.04295",
      "status": "supported"
    },
    {
      "claim": "StepFun-Prover-Preview-32B achieves 70.0% pass@1 on miniF2F-test, surpassing Kimina-Prover-72B
     (63.9%) and DeepSeek-Prover-V2-671B (61.9%) with a smaller model",
      "checked_against": "stepfun_note.md / Table 1 at stepfun.ai/research/en/stepfun-prover-preview",
      "status": "supported"
    },
    {
      "claim": "OpenAI o3 achieves 72.6% code correctness, 52.3% spec soundness+completeness, but only 4.9% proof
    success on VERINA; iterative refinement with Lean feedback reaches 20.1% at 64 steps",
      "checked_against": "verina_note.md / Figures 5\u20136 in arXiv:2505.23135",
      "status": "supported"
    },
    {
      "claim": "Tool-integrated RL with free-form Lean REPL interaction outperforms systems that allow only a
    single retry of verifier feedback",
      "checked_against": "stepfun_note.md / Introduction and Methodology sections at stepfun.ai",
      "status": "supported"
    },
    {
      "claim": "Proof generation is the bottleneck for end-to-end verifiable code generation: o3 achieves 41.2%
    code+spec jointly but only ~3.2% end-to-end with proof",
      "checked_against": "verina_note.md / Figure 6 in arXiv:2505.23135",
      "status": "supported"
    }
  ],
  "final_question": "Does integrating real-time formal verifier feedback into the LLM training loop via
    reinforcement learning\u2014rather than scaling model parameters or enlarging parallel informal-formal
    corpora\u2014represent the primary lever for advancing neural theorem proving systems across the full
    difficulty spectrum from undergraduate mathematics (ProofNet) to IMO-level problems (FIMO) and software
    verification (VERINA)?",
  "observations": [
    "Verifier-feedback integration during training, not model scale, drives the largest performance gains: StepFun
    -Prover-Preview-32B achieves 70% pass@1 on miniF2F by allowing the model to freely interact with the Lean 4
    REPL during rollout and training via GRPO, surpassing DeepSeek-Prover-V2-671B (61.9%)\u2014a model more than
    20\u00d7 larger\u2014by nearly 8 percentage points (source: stepfun_note.md / StepFun-Prover article, Table
    1).",
    "A hard difficulty cliff separates the competition-math regime from research-level formal proving: GPT-4
    achieves 21.3% Pass@8 on miniF2F but 0% Pass@8 on all 149 FIMO IMO-shortlisted problems, and ProofNet's Codex
     baseline of 13.4% on undergraduate mathematics similarly does not transfer\u2014indicating that benchmark-
    specific saturation in miniF2F masks genuine inability to generalize to harder formal reasoning tasks (source:
     fimo_note.md / FIMO Table 9; proofnet_note.md / ProofNet Table 3).",
    "Proof generation is the decisive bottleneck across both mathematical and software verification domains:
    VERINA shows that o3\u2014which achieves 72.6% code correctness\u2014falls to 4.9% proof success on
    structured programming verification tasks, and even with 64 Lean-guided refinement steps only reaches 20.1%,
    while the best specialized theorem-proving model (Goedel Prover V2 32B) reaches only 11.2%\u2014suggesting
    the formal proof generation problem is a domain-general failure mode, not specific to mathematical
    abstraction (source: verina_note.md / VERINA Figures 5\u20136)."
  ],
  "limitation": "The miniF2F benchmark where tool-integrated RL systems like StepFun-Prover achieve 70% pass@1 may
     be approaching saturation and reflects a narrow distribution of competition-math statement types; the same
    class of systems scores 0% on FIMO's IMO-shortlisted problems and under 12% on VERINA's software verification
     proofs, indicating that verifier-feedback training induces benchmark-specific rather than generalizable
    formal reasoning. The scarcity of high-quality parallel informal-formal data (flagged by ProofNet as a
    fundamental constraint) compounds this: without diverse hard proof examples to train on, RL-based systems
    cannot self-improve beyond the difficulty frontier of their training distribution.",
  "blocking_model_calls_needed": 6,
```

```
  "notes": "All four sources were successfully fetched and processed. The StepFun article is a Next.js server-
     rendered page; text was extracted by stripping HTML tags from the raw response\u2014the main article body was
      fully recoverable (~4,000 words). FIMO and ProofNet PDFs were extracted via pdftotext on pages 1\u201310 (
     approx. 40\u201349KB each). VERINA PDF is 10 pages at ICLR 2026 camera-ready density; the results section was
      cut off just after Figure 6 but the abstract and key quantitative results (code 72.6%, spec 52.3%, proof
     4.9%, end-to-end 3.2%) were captured in the available text. No sources were unavailable. The
     blocking_model_calls_needed estimate of 6 reflects: 1 planning call + 4 sequential paper-summary calls (one
     per source, each requiring a full model response to process extracted text) + 1 final synthesis call."
}
```


C. Per-Claim Audit

Table 4 is based on the detailed audit in `autoformalization_route_audit.md`. Each row pairs an extracted claim with the `/md` page (and section) where its support is located.

Claim	Evidence checked	Status	Notes
Autoformalization is hard because informal mathematics leaves assumptions implicit while proof assistants require explicit formal structure.	Autoformalization-gap research direction page; ProofNet paper page (typechecking, implicit hypotheses, retrieval).	Supported	Concrete example: orthogonal complement presupposes inner product space.
Compiler feedback turns formal theorem proving from one-shot generation into iterative correction.	Iterative-compiler-feedback research direction page; FIMO; StepFun-Prover; VERINA.	Supported	FIMO and StepFun document the feedback loop; VERINA documents costs.
Iterative feedback alone does not solve deep logical gaps.	VERINA; iterative-compiler-feedback research direction page (Semantic Wall section).	Supported (limitation)	64-round VERINA refinement plateaus near 20% ProofGen.
Token-level alignment of informal mathematics to formal compiler states is a validated result.	Token-level alignment research proposal page.	Not supported (research-proposal-level)	The research proposal page describes a planned method, not a result.
The run is perfectly clean and contains only theorem-proving papers.	Research direction context and related-paper tabs.	False (useful failure case)	The run includes some adjacent or noisy papers from broader verifiable-reasoning neighborhoods.

Table 4. Full per-claim audit on the autoformalization run. The detailed audit memo is in `autoformalization_route_audit.md`.

D. Map Construction Pipeline

Lacuna's hosted map is built by a pipeline running on production infrastructure. The pipeline starts from scholarly records, paper files, author metadata, and venue metadata. LLM stages then create core-idea paper summaries, paper summaries with figures, concept elements, research directions, profiles, and research proposals, while preserving the database links needed to follow generated content back to the records that produced it. The production map used the Gemini-family models named in Section 3. Figure 2 shows the paper-to-direction generation subpipeline; Table 5 summarizes the full construction pipeline.

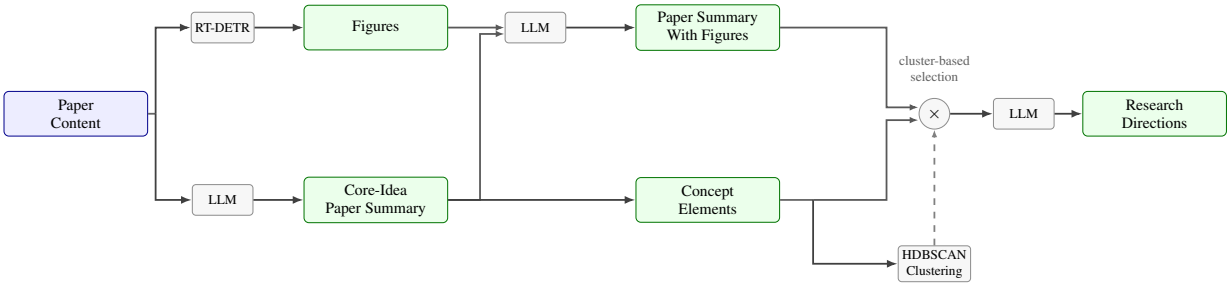

Figure 2. Core paper-to-direction generation in Lacuna. Paper content is processed into figures and core-idea summaries; summaries and figures are merged into paper summaries with figures, core summaries produce concept elements, and cluster-based selection supplies summaries and concepts to an LLM synthesis stage that writes research directions.

Table 5. Major construction stages and durable outputs used by the hosted Lacuna map.

Stage group	Primary durable outputs	Role in the map
Catalog harvest and reconciliation	Paper, author, venue, identifier, and authorship records; duplicate and alias resolutions	Establishes stable paper, author, and venue records before any generated synthesis is attached.
Source acquisition	`blob_metadata` file identities and `blob_location` replicas for PDFs or URLs	Separates the identity of a paper file from where it can be fetched, so the system can attach new replicas without changing the paper object.
Paper text processing	Core-idea paper summaries, paper tags, and explanation records	Produces the text used for concept extraction and for readable paper pages.
Figure-rich paper summary processing	Extracted figures, selected figures, captions, and paper summaries with figures	Produces compact paper summaries that combine text and selected visual evidence.
Concept extraction	Concept elements, embeddings, paper assignments, and provenance snippets	Converts paper-level observations into one- or two-sentence units that can be clustered, searched, and linked back to papers.
Research direction clustering	`taxonomy_node` rows and `concept_element_assignment` links	Groups recurring technical problems, methods, and opportunity areas into research direction pages with representative papers and neighboring research directions.
Profile and research proposal generation	Author profiles, venue profiles, Alien Science samples, and research proposal pages	Uses author, venue, paper, paper summaries with figures, and research direction context to generate readable profiles and candidate hypotheses.
Serving indexes and exports	Search indexes, lookup indexes, relation tables, HTML pages, `/md` variants, and MCP access	Makes the map usable through search, tabs, hover previews, pagination, typed links, markdown pages, and tool calls.

Authorship-anchored bibliography expansion. Resolved author pages are built around OpenReview profile identities. For each OpenReview-profiled author a, the catalog pipeline first harvests the author's OpenReview papers, including records available only through legacy OpenReview APIs, and stores their positional author lists. The pipeline canonicalizes OpenReview profile aliases and aligns raw author identifiers to author names before using them as person-level links, so older records with aliases, emails, or other legacy identifiers do not fragment a single author into multiple resolved profiles.

The pipeline then constructs a trusted co-author neighborhood C_a from OpenReview papers only. Each edge records a normalized co-author name, an optional resolved co-author identifier, and the number of shared OpenReview papers. This OpenReview-only neighborhood is used as an authorship signal for external bibliography expansion. External identifiers declared by the OpenReview profile are followed into DBLP and OpenAlex to obtain candidate papers E_a. For each candidate paper, the pipeline reads the source's author list, normalizes names, and applies the following rule: retain the paper for a if it reconciles to an existing OpenReview paper for a, or if at least one candidate co-author matches C_a; otherwise the

candidate is left unattached to the resolved author profile unless later reconciliation provides stronger evidence. Candidates from authors with too little OpenReview co-author evidence are deferred, not treated as high-confidence rejections.

After this filtering step, paper reconciliation merges duplicate source records using identifiers, normalized titles, years, and author overlap, preferring canonical records that preserve OpenReview provenance when available. The resulting bibliography for an author is therefore the union of OpenReview papers and externally harvested papers whose authorship is supported by the OpenReview co-author graph. This gives author and venue profiles broader coverage than OpenReview alone while also keeping external-source disambiguation errors from propagating into generated summaries, research-direction memberships, and proposal context.

Records. Lacuna treats source records, source files, metadata entities, and generated content as separate records. A paper page is keyed by a canonical artifact identifier; its files are keyed by content-hash-like blob metadata; its core-idea paper summary, paper summary with figures, extracted figures, concept elements, research direction memberships, and research proposal links are stored as derived records. This separation matters in practice: a paper can remain searchable if a PDF URL dies, a new open-access replica can be attached without changing the paper identity, and generated synthesis can be regenerated while preserving links to the underlying records.

Generated Content and Links. The pipeline does not treat model-written text as free-floating prose. Concept elements are assigned back to papers; research direction articles are built from clusters of concepts and representative papers; research proposal pages list the research directions and papers they use as context; author and venue pages are generated from metadata plus linked papers and directions. The serving layer preserves these relationships as visible tabs, evidence sections, and markdown links. This is why the claim audit in Section 5.2 can check generated claims against specific page sections.

Pipeline organization. Pipeline stages are numbered, rerunnable, and idempotent at the stage level: harvest and reconciliation stages build the normalized catalog; PDF/blob stages acquire and verify paper files; batch LLM stages write core-idea paper summaries, concept elements, research direction articles, research proposals, paper summaries with figures, and profiles; embedding and clustering stages build the research direction graph; serving stages export lookup tables and search indexes. Large batch jobs run on production GCE infrastructure and write durable counters and manifests.

E. ScholarQA-CS-ML Evaluation Details

ScholarQA-CS (called Scholar-CS in the OpenScholar paper) is the computer-science subset of ScholarQABench (Asai et al., 2026). It contains 100 literature-review questions written by expert annotators holding PhDs; each question has detailed answer rubrics built from literature evidence and supporting quotes. The public test configuration we used contains 100 unique prompts.

To define the ML/AI slice, Codex GPT-5.5 with xhigh reasoning effort was instructed to act as a classifier and select prompts about machine learning, artificial intelligence, NLP, or robot learning. This produced 42 questions, or 42% of ScholarQA-CS. We evaluated all 42 selected prompts as-is, without tuning or filtering by Lacuna or OpenScholar performance.

Lacuna-GPT-4o used abstract-only Elasticsearch evidence from Lacuna, a facet-conditioned retrieval plan, top-15 paper evidence, concept evidence packets, and compact cited evidence appended to the answer with original citation IDs preserved. It reached an average rubric score of 0.694 over the 42 selected questions. On the same 42 questions, the released OpenScholar-GPT-4o answer file scored 0.672, while the same OpenScholar answers with appended references stripped scored 0.589. Lacuna-GPT-4o without the appended cited-evidence block scored 0.572. We report both answer-only and answer-plus-evidence formats because the released OpenScholar file itself includes appended reference material; the comparison is therefore both a synthesis check and a check that Lacuna can serve useful source context to the answerer. The component breakdown in Table 3 shows that Lacuna-GPT-4o improves the content-side rubric metrics while OpenScholar-GPT-4o retains higher structural citation coverage.

Table 6. ScholarQA-CS-ML question IDs and prompts.

Question ID	Prompt
00bdd80debc8549198001289188c6bea	How can I use an hybridization of ontology and machine learning for text summarization ?
018bff91263ab3be5f8ad5bade76b030	In robotics, what are the leading methods for learning terrain traversibility costs automatically from robot experience?
0348920a58979e759af9081a6225ee0d	What are the advantages and limitations of applying bias mitigation algorithms during pre-processing, training and inference stages of a model?
0919a8528cd20166163de3fdcb089efa	Have large langauge models been applied to the schema matching problem in databases and are they effective?

Question ID	Prompt
11e71107ddfdc824b8b87d4f5a2ef843	What data preprocessing steps are most important for point cloud datasets before performing surface reconstruction?
15dec998cf77887f870ebf9a55bb7e89	How effective are language models at automatically generating textual descriptions of scientific concepts?
1f384c4d3942c46b676b5bfc66447192	How good are LLMs at solving traditional tabular ML datasets using ICL?
2bb40aa93ac3a6a673c839bd660718ac	Have specialized approaches been developed for providing LLM assistance when people author SQL queries?
2dc0ff181a621680dde0a48e0d63f0d9	How has the citation graph been used to improve neural language models for scientific papers?
323e85c9052082358fc0c045fe20a537	How do different fairness metrics correlate with each other across various datasets and model architectures?
4534bd4b99ea2bfd1efd8c656e9264c7	What datasets and methods are used to pre-train models on table specific tasks?
5acd0e1d36af3c52c3159b4b230bcc2f	What interfaces have researchers developed to help people perform behavioral evaluation of ML models, and how do they accelerate annotation efforts?
6f526f72804ce3eb59feb7046f319ccc	What are good practices for detecting AI-generated texts in situations where false positives are extremely expensive?
803bc7891917f823a52948ebee89cf9d	How does in-context learning for LLMs differ from traditional machine learning model training?
928646729f5d824dd9a52e9ddff70e58	Describe what is known about overfitting in in-context learning.
948b6cb986a5d7732722975dbed9d420	What are leading approaches for evaluating complex scientific question answering systems in NLP?
a8ba07610b6d77890e50144bfd4d4168	What are some common UI designs for sense-making, information organization, and AI writing tools?
ab3651c422a54f40aa4bac3fa630c13e	How do large language models like ChatGPT impact the diversity of published scientific ideas?
baee287ff68f0fd60dcbd0d8b9b741b8	Apart from preventing overfitting, are there any side effects (desirable or otherwise) of applying dropout in deep neural networks?
bb7198e650267504d67b14b6e4c67c7c	what are the most important open challenges in using neural networks in combination with PDE solvers for fluid simulation?
be5c0337461175e55f2a8fa9bcce5732	What are the leading approaches for anomoly detection in process mining?
ce433b751f38f7d0173095c2faa2f75a	What are advantages and disadvantages of top methods for picking the right number of topics in topic modeling?
e2d0eb391fbf674c070c74c387ee6248	During pre-training, why is the transformer embedding layer initialized randomly, rather than with pre-trained embeddings from existing models?
e894bc20daf0522da9c576ae27b257de	What are the main challenges in adapting transformer-based models for tabular data representation?
fb607bc177d2efb926cb3dff15668861	How can question generation be used to mitigate hallucination in LLMs?
fc927b39177cd2aad8a8cbcef75ab62c	What are the leading thoughts about how to teach computer science to college students in the age of generative AI?
0ed1770483ec64633a580366026dd16e	What are some of the challenges associated with creating a training dataset for question answering in scientific domains? Describe some recent methods that try to overcome these challenges.
34b00939c1190990c1c9be590cb07476	How has literature discussed AI as a design material?
3d8c315aed4cc104f2ad61f4deeda9c1	What are the leading techniques for person-following robot navigation (which must track the person and potentially re-identify if occluded)?
5480ba91951fc42e9beb989eea40360d	How does the addition of XAI techniques such as SHAP or LIME impact model interpretability in complex machine learning models ?
6453c86c72949978fb8db5405b48c923	is there any evidence that large language models can be effectively applied to robot planning tasks?
64ef9b9e4c220dd1a2f6115b2a9e242a	What are the most important open challenges in Federated Learning?
6e7e3524d565599b3064ae05375956f7	what are the best recent techniques for text watermarking, what kinds of transformations are they robust to, and what transformations or attacks are they not robust against
7670af632f0932d5ed20c34e1c9f01d2	What interfaces have researchers developed for helping people optimize LLMs for a task, and what are the biggest remaining user problems?
983e73defc06e6794a856330905dc787	Does active learning work well when fine-tuning large language models?
a5d23eb3a2555db0a82f6b64fed85baa	What are some tasks where fine tuning smaller models is beneficial over using LLMs?
aab38dd1282ff3387cc8bf9bdf13b3aa	What are the leading approaches to automatic scientific paper review generation and what are their strengths and weaknesses?
b069a1248503c4caa98dab0014d1a55a	In recommendation systems, how are new methods that optimize diversity typically evaluated?
e03c49cccc971cf3ae67556554b4666b	Are there papers that use different formats of Q&A with the user to clarify intent and compose more complicated prompts to LLM?
e09a30179e0b587d57edb17317ae3288	How is artificial intelligence being utilized to enhance the diagnosis and treatment of sleep apnea?
e61be146ef53f1a5483c668fc4f6390c	What principles of interactive machine teaching can be applied to interactively curating social media feeds?
eef3ee38231d9fa5ffabbec75d1c5b50	What are good benchmarks and evaluation strategies for comparing obstacle avoidance strategies in mobile robots?

F. Reproducibility Details

Run command. `python3 agent_workflow_eval/run_claude_eval.py --model sonnet --max-budget-usd 1.00` (run from the supplementary archive root). **Model.** Claude Sonnet (via Claude Code, non-interactive, `--print` mode). **Date of run.** 2026-05-08. **Working directory.** `/private/tmp/lacuna_workflow_eval` (script-managed). **Outputs.** Stream JSON and per-condition machine-readable summaries are written under `agent_workflow_eval/outputs/2026-05-08_pdf10_notes_baseline/`; `results.md` provides a human-readable summary. **Token accounting.** Output tokens are taken from the Claude Code summary stream; the $ figures are the harness-reported cost. **Map version.** The hosted map snapshot used for the run was the deployment on 2026-05-08; the scale counts in Table 1 were checked on 2026-04-29 and have not materially changed in the intervening days.