DOES RECURSIVE KNOWLEDGE CONSTRUCTION HELP AGENTS VERIFY CONSENSUS CLAIMS IN SCIENCE?

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent scientific agents build knowledge recursively by retrieving evidence, constructing an intermediate state, and judging whether that state contains sufficient information. These agents are evaluated end-to-end, usually without ground truth, so the effect of each design choice is unknown. We examine two design axes: (1) what the recursion builds (the state of evidence) and (2) what controls it (the judge that decides when to stop). To measure them, we define consensus scientific claim verification and measure these effects on this task, where a system must establish a claim's consensus stance from the open literature. We build a new consensus claim dataset, ProClaim-eval, including 718 claims with SUPPORT and REFUTE gold labels from five expert-curated databases in molecular biology, nuclear physics, and astronomy. Our instrument, ProClaim, persists an evidence state outside the model's context, so each axis can be studied with the other held fixed. Against verification systems and general-purpose agent harnesses, ProClaim ranks among the top three, with a significant advantage over the harnesses. We find that the evidence state improves performance over in-context accumulation, and it also matters what is stored in it. In addition, a trained judge that reads the evidence state performs better than fixed-round stopping. A judge trained separately and never exposed to ProClaim-eval achieves the highest agreement with the gold labels on three of five subsets, where it matches the off-the-shelf Claude Sonnet 4.6 judge and outperforms Claude Haiku 4.5. Our results on ProClaim-eval support recursively maintaining a structured evidence state outside the model's context as an effective design for verifying claims in scientific agents.

1 INTRODUCTION

Scientific agents built on large language models (LLMs) can now retrieve, synthesise, and reason over the research literature at scale (Bachman et al., 2023; Wood et al., 2026). Many agentic systems (Ghareeb et al., 2025; Skarlinski et al., 2024; Mitchener et al., 2025; Gottweis et al., 2026) execute a common loop, which constructs a record of evidence and judges whether the record suffices for a given task, and if not, uses the record to direct the next retrievals. The loop is recursive because each step's record is the next step's input (Zhang et al., 2025). However, these agents contain many components and are evaluated only on final outputs, so effectiveness cannot be attributed to any single part. Whether *what the recursion builds* and *what controls it* actually help is therefore assumed rather than carefully examined. We call these the two design axes of *recursive knowledge construction*.

To measure each axis, we require a task that exercises the loop and has established consensus. Scientific claim verification is the natural candidate. Claims are the basic unit of literature synthesis and hypothesis generation, and building the consensus on a claim is essential to counter research misinformation (Deng et al., 2025; Diggelmann et al., 2020; van der Linden, 2022; West & Bergstrom, 2021) and to uncover gaps that may drive new discovery (Asai et al., 2026; Vladika & Matthes, 2023). Existing datasets (Diggelmann et al., 2020; Reisle et al.,

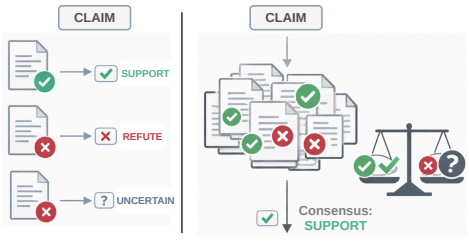

Figure 1: Document-guided vs. consensus scientific claim verification.

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

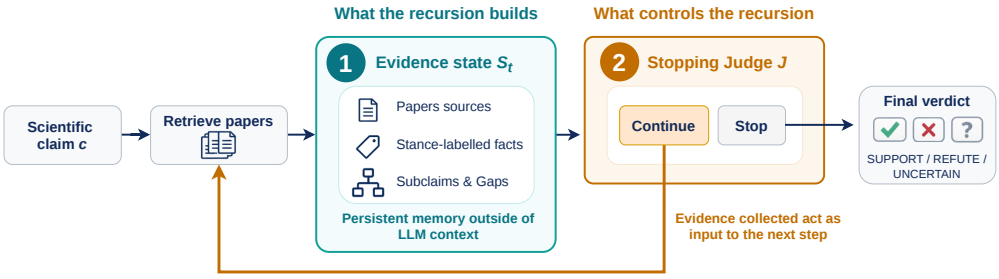

Figure 2: Recursive knowledge construction for claim verification. We examine two design choices in Section 4: (1) an evidence state that the next retrieval is conditioned on, (2) a judge that takes the state as input and decides either to continue or terminate the construction, after which the verdict is given from the final state.

2025; Saakyan et al., 2021; Sarrouti et al., 2021; Wadden et al., 2020; 2022a), however, have reduced verification to reading comprehension over a single retrieved passage (Vladika & Matthes, 2024). This document-guided setting (Figure 1) has inherent constraints, as verdicts grounded in a limited context cannot reflect the actual scientific stance of claims across the broader literature (Deng et al., 2025; Kumar et al., 2025; Skarlinski et al., 2024). We therefore define *consensus claim verification* as an open-domain (Vladika & Matthes, 2024), claim-level task with labels reflecting the epistemic state of the literature. Solving it requires retrieving, organising and reconciling sparse and conflicting evidence across many sources (Vladika et al., 2025; Wang et al., 2024b), which functionally demands a recursive agentic loop.

To make this setting evaluable, we release **ProClaim-eval**, a new dataset of 718 consensus claims built from five actively curated expert databases spanning molecular biology (Lo Surdo et al., 2025; Liu et al., 2026; Vazquez et al., 2022), nuclear structure (Tuli, 1996) and astronomy (Akeson et al., 2013; Christiansen et al., 2025). Prior claim verification datasets offer only passage-level annotations because curating consensus stances demands substantial expertise (Guo et al., 2022; Wadden et al., 2020; Wang et al., 2024b). These databases are actively maintained, so their entries track the current domain consensus rather than the stance of any one paper. Gold labels are SUPPORT or REFUTE, and a system may additionally answer UNCERTAIN, which we treat as an abstention. To vary the two design choices in recursive knowledge construction (Figure 2), we build **ProClaim**, a minimal open-source instrument that maintains an evolving evidence state and a replaceable judge. Findings from our instrument only generalise if it performs on par with systems in use, so we compare ProClaim with state-of-the-art verification systems (Xie et al., 2025; Asai et al., 2026) and general-purpose agent harnesses (CODEX and Claude Code).

Our work makes three main contributions:

- **We formulate consensus scientific claim verification and release ProClaim-eval** as a test domain for recursive knowledge construction, distinct from document-guided verification. ProClaim-eval contains 718 claims from five expert-curated databases across three domains, with binary gold labels. On ProClaim-eval, ProClaim ranks among the top three systems on four subsets, with a significant advantage over the Claude Code and Codex agent harnesses.
- **Recursively built evidence state beats in-context accumulation, and what it stores matters.** On ProClaim-eval, ProClaim outperforms a ReAct agent that accumulates the retrieved papers only in its context, with the same LLM backbone and retrieval tools. Using ProClaim-retrieved papers, we show that the extracted facts in the state give better performance than abstracts or full text as input.
- **A lightweight trained judge can transfer and replace off-the-shelf LLM judges.** With ProClaim as the instrument, we find that a trained judge that reads the state is better and uses fewer steps than a fixed round budget. We trained a lightweight MLP judge, fitted only on SciFact-Open and evaluated in transfer, as an alternative to off-the-shelf LLM judges, which matches the Claude Sonnet 4.6 judge and outperforms the Claude Haiku 4.5 judge on three subsets.

Table 1: **Claim diversity in the ProClaim-eval benchmark.** We show one example claim per subset, with the subset's claim count and SUPPORT / REFUTE split. The five subsets instantiate five distinct single-sentence templates over three scientific domains with a total of 718 claims.

	Subset	Claims	Example claim	Gold label
Molecular biology	**ConnectomeDB**	**305** 184 S / 121 R	A2M as ligand directly interacts extracellularly with HSPA5 as receptor	SUPPORT
	SIGNOR	**101** 38 S / 63 R	CSK directly activates SRC	REFUTE
	ExTRI	**125** 49 S / 76 R	CEBPA directly activates transcription of CSF3R	SUPPORT
Astronomy	**NASA Exoplanet Archive**	**102** 55 S / 47 R	TOI-2267 d is a planet orbiting the star TOI-2267 B	SUPPORT
Nuclear physics	**NUBASE**	**85** 77 S / 8 R	Niobium-86 has an isomeric state with a half-life of about 56.3 seconds	REFUTE
	Total	**718** 403 S / 315 R		

2 Document-Guided Versus "Consensus" Claim Verification

Establishing the consensus stance of a scientific claim is essential for uncovering research gaps (Asai et al., 2026; Vladika & Matthes, 2023) and curating reliable scientific knowledge (Deng et al., 2025; Diggelmann et al., 2020). When verifying consensus claims, the verdict emerges from reconciling heterogeneous and conflicting evidence across multiple retrieval passes. Therefore, no single passage can be assumed authoritative for this purpose. Existing benchmarks (Wadden et al., 2020; 2022a; Reisle et al., 2025) follow this document-guided paradigm, and are inherently limited. Here, we distinguish these two problem settings:

- **Document-guided verification.** A verdict $v = f(\mathcal{C}, d)$ is associated with a claim $\mathcal{C}$ *and a single passage* $d \in \mathcal{D} = \{d_1, d_2, \ldots, d_N\}$, where $\mathcal{D}$ is a fixed set of documents. The goal is to retrieve the passage given the claim and *classify the pair* $(\mathcal{C}, d)$.
- **Consensus claim verification.** A verdict $v = f(\mathcal{C}, \mathcal{D}_\mathcal{L})$ is associated with a claim $\mathcal{C}$ *against a set of passages* $\mathcal{D}_\mathcal{L} \subseteq \mathcal{L}$, where $\mathcal{L}$ is a large document collection representing the open scientific literature. Since the aim is to *predict the consensus verdict*, the system must determine both which documents to include in $\mathcal{D}_\mathcal{L}$ and how many passages are needed.

ProClaim-eval. Existing scientific claim verification datasets contain verdict annotations for each claim–passage pair. This limits evaluation when assembling findings across multiple papers to reach scientific consensus. To address this gap, we create ProClaim-eval (Appendix B), 718 single-sentence claims from five expert-curated databases spanning three domains (Table 1). None of the databases provides claim-level verdicts, so each claim's SUPPORT or REFUTE label is derived from the curators' own record of the underlying evidence. In molecular biology, ConnectomeDB (Liu et al., 2026) contributes 305 ligand-receptor interactions, labelled SUPPORT when added in its 2025 release with at least two supporting publications and REFUTE when rejected by its curators from the 2020 release. SIGNOR (Lo Surdo et al., 2025) contributes 101 expert re-curated claims about signed protein-protein interactions, whose REFUTE claims are interactions removed in later releases or confirmed interactions with a negated sign. ExTRI (Vazquez et al., 2022) (part of the CollecTRI database (Müller-Dott et al., 2023)) contributes 125 transcription factor-target regulations drawn from expert-read sentences, labelled SUPPORT when at least five papers agree on the direction of regulation and REFUTE when every curated sentence negates it. In astronomy, the NASA Exoplanet Archive (Akeson et al., 2013; Christiansen et al., 2025) contributes 102 claims that an object is a planet orbiting a given star, labelled SUPPORT for confirmed planets and REFUTE for published planets later refuted or found to be false positives. In nuclear physics, the NUBASE2020 evaluation (Kondev et al., 2021) contributes 85 claims that a nuclide has an isomeric state, labelled SUPPORT for established states and REFUTE for

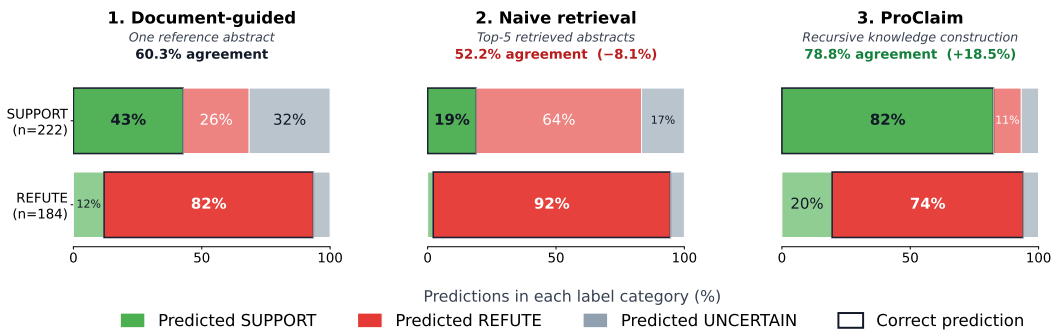

Figure 3: Verdict prediction-label agreement (outlined segments) across three verification settings on the SIGNOR and ConnectomeDB subsets of ProClaim-eval (406 claims), using Claude Sonnet 4.6. Both document-guided and naive retrieval verification do not produce consensus reliably and over-produce REFUTE predictions due to insufficient or noisy information, a limitation that ProClaim addresses. This comparison with ProClaim highlights the benefit of recursive knowledge construction, driven by adaptive retrieval and evidence organisation before deciding the final verdict.

states reported and later shown not to exist. In ExTRI, NUBASE, and NASA, since their most recent update took place before 2025, we further remove claims that an LLM answers correctly without retrieval, so that a verdict requires retrieving the literature rather than recalling it.

The need for consensus claim verification. ProClaim-eval assigns a label to each claim from categories {SUPPORT, REFUTE} (label definitions in Appendix B.1). Our analysis in Figure 3 shows that **document-guided verification is insufficient for consensus.** With one expert reference abstract as the "silver" standard context to guide LLM claim verification, the agreement between verdict predictions and labels is only 60.3%, which cannot reliably proxy scientific consensus. **Naive retrieval is insufficient too.** Five retrieved abstracts reduced agreement to 52.2% and misclassified 24% SUPPORT claims compared to using the reference abstract. Importantly, both document-guided verification and static retrieval overproduce REFUTE predictions due to failing to find explicit positive support under limited or noisy evidence, and absence of support after search is treated as REFUTE.

3 Recursive Knowledge Construction

Our results in Section 2 show that verifying scientific consensus cannot be reduced to reading from naively retrieved passages. In this section, we formally define the mechanism *recursive knowledge construction* that recent scientific agents use instead. This leads us to build the minimal instrument ProClaim that allows us to vary the two design axes of recursive knowledge construction to examine their effects on consensus verification.

3.1 Formal Definition

We formalise the process shared by recent scientific agents (Skarlinski et al., 2024; Mitchener et al., 2025) as a recursion on an explicit evidence state. Let c be a scientific claim and $\mathcal{L}$ a corpus of documents as in the consensus claim verification setting. At step t, a verifier maintains an *evidence state* s_t as an intermediate record, which is updated via:

$$s_{t+1} = U\big(s_t,\ R(c, s_t)\big), \qquad t = 0, 1, \ldots$$

where $R(c, s_t) \subseteq \mathcal{L}$ decides what to retrieve given what the state already holds, and U updates the state by recording the result. Verification halts at the first round T at which a judgement $J(c, s_T)$ holds, after which the agent produces its output verdict from the final record. Two properties make this process recursive. The evidence state contains the knowledge constructed by the agent, and the state in the previous round becomes the input to the next. The state also guides both where the agent looks next and when it should terminate.

Two design choices are key in the definition, and they are the objects of study in this paper.

- **Evidence state.** In contrast to claim verification systems such as FIRE (Xie et al., 2025), SAFE (Wei et al., 2024) and OpenScholar (Asai et al., 2026), which keep a record of the growing prompt context, in scientific agents, for example, PaperQA2 (Skarlinski et al., 2024) and Kosmos (Mitchener et al., 2025), the state is explicitly constructed as structure persisted outside the context.
- **Judgement.** The verification process needs a stopping rule. This could be a fixed number of rounds (Asai et al., 2026) or a function of the retrieved evidence (Xie et al., 2025; Asai et al., 2024) based on an LLM judgment of whether to keep searching.

We implement a minimal instrument agent, ProClaim, to perform recursive knowledge construction and examine the effects of those two design choices. The experiments therefore vary one axis with the other fixed in ProClaim.

3.2 The Evidence State

We design the evidence state as a persistent JSON outside the LLM context. The state records retrieved papers and claim-relevant facts linked to their source paper, with a stance in $\{\text{SUPPORT}, \text{REFUTE}, \text{NEUTRAL}\}$. We follow the CodeAct (Wang et al., 2024a) approach, so at each step, the agent writes a short Python program. This program is composed of operations to interact with the state from a small library, including fact extraction with stance labelling from abstracts or full text, subclaim decomposition, and the judge call.

This state design is minimal and preserves agentic behaviour rather than executing a fixed pipeline. Because the agent needs to access the state through explicit programmatic operations, its content is not compressed or paraphrased as steps accumulate, unlike evidence folded into the growing prompt that dilutes as the context grows. The agent's context is the execution log of the operations it has run and their printed results. Because the query at round $t + 1$ is issued after reading the state, retrieval is conditioned on the accumulated evidence as we have defined in Section 3.1.

3.3 The Judge

Section 2 shows a characteristic failure in retrieval tasks (Xie et al., 2025; Jeong et al., 2024; Asai et al., 2024), where claims require aggregating multiple documents, and an absence of support after a shallow retrieval leads to more REFUTE. Stopping too early yields a verdict from incomplete or one-sided evidence, but more retrieval on claims can introduce noise or conflict. Following the notion of *sufficient context* (Joren et al., 2025), we therefore want the judge to score a property of the evidence state itself. We design the judge as a scorer $J(c, s_t) \in [0, 1]$. The agent stops when $J(c, s_t) \geq \delta$ or reaches a maximum iteration $T_{\max}$ for practical constraints. We implement J in two ways that consume the same state and the same $\delta = 0.8$, with a cap $T_{\max} = 4$.

Trained judge. We train a lightweight 2-layer MLP (hidden dimensions 64 and 32) and expose it to the agent as a tool, so the stopping signal is external rather than a self-assessment. We prepare training data using SciFact-Open (Wadden et al., 2022a), which labels stance at the paper level for 279 claims. Papers sharing claims are grouped into the same evidence states and use a minority ratio to annotate sufficiency labels (Appendix D.3). Per-paper features are extracted from frozen, off-the-shelf encoders, then pooled across all papers in the evidence state into one global representation. We compute the cosine similarity from SBERT (Reimers & Gurevych, 2019) between claim and paper text. This distinguishes states with actually relevant evidence from states with off-topic papers. NLI stance probabilities (entailment, contradiction, neutral) come from a DeBERTa NLI model (He et al., 2021) applied to the claim against the most relevant text chunk from each paper. This acts as a soft, per-paper stance signal. Together with entity coverage and bibliometric metadata, these are pooled into a 10-dimensional global vector that feeds the MLP. We include implementation details in Appendix D.4.

Off-the-shelf judge. An LLM is prompted zero-shot with the claim and a rendering of s_t (paper titles, abstracts, the per-paper meta-data, and similarity scores from above) and asked to return a sufficiency score, with the definition above stated in the prompt (Appendix D.6).

Table 2: Agreement (Agr.) and abstention rate (Abst.) per ProClaim-eval subset. The shaded figure beside each agreement is ProClaim's agreement minus that row's, on the same subset; green where ProClaim is ahead, red where it is behind. The highest agreement per subset is underlined.

	SIGNOR		ConnectomeDB		ExTRI		NUBASE		NASA	
Method	Agr.↑	Abst.	Agr.↑	Abst.	Agr.↑	Abst.	Agr.↑	Abst.	Agr.↑	Abst.
Random	0.32	0.35	0.34	0.34	0.32	0.34	0.34	0.34	0.31	0.35
LLM Only										
Claude Sonnet 4.6	0.63 +0.17	0.26	0.53 +0.26	0.25	0.21 +0.51	0.66	0.05 +0.47	0.95	0.07 +0.59	0.89
Gemini 2.5 Flash	0.59 +0.21	0.26	0.57 +0.21	0.21	0.23 +0.49	0.56	0.31 +0.21	0.07	0.46 +0.20	0.03
Static Retrieval, S2 Top 5										
Claude Sonnet 4.6	0.66 +0.14	0.10	0.48 +0.30	0.12	0.60 +0.12	0.06	0.13 +0.39	0.32	0.36 +0.29	0.55
Gemini 2.5 Flash	0.64 +0.16	0.06	0.51 +0.27	0.04	0.62 +0.10	0.03	0.18 +0.34	0.02	0.38 +0.27	0.51
Adaptive Retrieval										
ReAct	0.78 +0.02	0.10	0.70 +0.08	0.08	0.74 -0.02	0.16	0.18 +0.34	0.71	0.55 +0.11	0.39
FIRE	0.75 +0.05	0.06	0.50 +0.29	0.14	0.73 -0.01	0.09	0.08 +0.44	0.12	0.70 -0.04	0.07
OpenScholar	0.36 +0.45	0.58	0.50 +0.28	0.32	0.22 +0.50	0.73	0.02 +0.49	0.98	0.03 +0.63	0.96
General-Purpose Agents										
Claude Code	0.83 -0.03	0.04	0.74 +0.05	0.05	0.82 -0.10	0.10	0.22 +0.29	0.65	0.55 +0.11	0.33
Codex	0.86 -0.06	0.04	0.74 +0.05	0.03	0.79 -0.07	0.10	0.19 +0.33	0.59	0.57 +0.09	0.28
ProClaim	0.80	0.12	0.78	0.05	0.72	0.14	0.52	0.36	0.66	0.14

4 EXPERIMENTS

Shared setting. Unless otherwise noted, all variations are evaluated across five subsets of the ProClaim-eval dataset. We report *agreement*, the fraction of all claims whose predicted verdict matches the gold label, as in Section 2. A system may answer UNCERTAIN to abstain, and we compute the *abstention rate* as the fraction of claims answered UNCERTAIN. Every agentic system, including ProClaim, uses Claude Sonnet 4.6 with thinking disabled and temperature 0. Paper search is PubMed and Semantic Scholar (S2) on the three biology subsets and S2 alone on non-biology subsets, except for LLM-only baselines where models answer from parameter knowledge. The fact extraction in ProClaim is done by Qwen 3.5 9B. For each finding, we compare two systems on the same claims with a paired exact McNemar test, pooled over subsets, and report Holm-adjusted p-values (Appendix C.2).

4.1 HOW DO VERIFIERS AND AGENTS PERFORM ON CONSENSUS CLAIM VERIFICATION?

Settings. *Random* draws each verdict uniformly from SUPPORT, REFUTE and UNCERTAIN, averaged over 10 draws, as a chance floor. *LLM-only* makes one call with no retrieved context. *Static retrieval* retrieves the top-5 S2 abstracts for the claim and asks the same LLM for a verdict in one shot. *ReAct* (Yao et al., 2023) runs a thought-action-observation loop over the paper-search tools. *FIRE* (Xie et al., 2025) is a claim verifier built around web search, and we update its prompts to adopt the paper-search backend as described above. FIRE stops when the LLM reports sufficient confidence. Both ReAct and FIRE are capped at 20 steps, with a budget matching ProClaim's average number of retrievals. *OpenScholar* (Asai et al., 2026) retrieves from the same paper-search sources and refines its answer with LLM feedback. We rephrased claims as questions to use OpenScholar's native LLM feedback pipeline. *Claude Code* and *Codex* are general-purpose agent harnesses given the same paper-search tools with web search disabled. To bound run time, Claude Code is capped at 60 turns and Codex at 45 minutes per claim. Every system receives the same label definitions (Appendix B.1).

Results. Table 2 reports agreement and abstention per subset. **First**, ProClaim has the highest agreement on ConnectomeDB (0.78) and NUBASE (0.52), trails Codex on SIGNOR (0.80 vs. 0.86), and trails Claude Code on ExTRI (0.72 vs. 0.82) and FIRE on NASA (0.66 vs. 0.70). **Second**, static retrieval scores below LLM-only on ConnectomeDB with both LLMs and on NUBASE and NASA

Table 3: Effect of the evidence state. The same LLM (Claude Sonnet 4.6) reads the different information retrieved in the evidence state of ProClaim, against no retrieval. Every row is scored on the same randomly sampled 40 claims per subset. We show the agreement (Agr.) and abstention rate (Abst.) per subset, and the highest agreement per subset is underlined.

	SIGNOR (40)		**ConnectomeDB** (40)		**ExTRI** (40)		**NUBASE** (40)		**NASA** (40)	
Evidence & Methods	Agr.↑	Abst.	Agr.↑	Abst.	Agr.↑	Abst.	Agr.↑	Abst.	Agr.↑	Abst.
No retrieval	0.63	0.30	0.70	0.23	0.28	0.65	0.10	0.90	0.08	0.88
Abstracts	0.78	0.13	0.83	0.10	0.75	0.25	0.28	0.45	0.63	0.23
Full text	0.80	0.08	0.83	0.10	0.83	0.10	0.38	0.28	0.65	0.18
Facts	0.80	0.10	0.93	0.08	<u>0.88</u>	0.05	<u>0.50</u>	0.15	0.73	0.08
Facts + abstracts	0.80	0.08	0.93	0.05	0.83	0.13	<u>0.50</u>	0.15	<u>0.75</u>	0.03
No evidence state, ReAct										
+ subclaims - state	0.78	0.23	0.75	0.15	0.83	0.15	0.20	0.70	0.43	0.50
+ subclaims + facts - state	0.73	0.20	0.78	0.20	<u>0.88</u>	0.13	0.23	0.65	0.55	0.38
ProClaim	<u>0.83</u>	0.05	<u>0.95</u>	0.03	0.85	0.10	<u>0.50</u>	0.30	0.73	0.05

with Gemini 2.5 Flash, showing that naively retrieving the top-5 papers or snippets does not add reliable benefit over parametric knowledge. **Third**, adaptive retrieval improves on static retrieval for the systems that commit to a verdict. ReAct and FIRE exceed static retrieval on almost every subset. OpenScholar abstains on most claims on every subset except ConnectomeDB (from 0.58 on SIGNOR to 0.98 on NUBASE) and scores below LLM-only Sonnet 4.6 on SIGNOR (0.36 vs. 0.63). **Fourth**, the SOTA harnesses are strongest on the biology subsets. Codex has the highest SIGNOR agreement (0.86) and Claude Code the highest ExTRI agreement (0.82), but on NUBASE both stay below the random baseline (0.22 and 0.19 vs. 0.34), abstaining on 0.65 and 0.59 of claims.

Finding 1: Recursive construction is consistently strong

Recursive knowledge construction shows strong performance on ProClaim-eval. Our instrument ProClaim ranks amongst the top 3 on four subsets, with a significant advantage against harnesses (pooled agreement 0.73 vs. 0.68 for Claude Code and 0.68 for Codex, $p = 0.022$).

4.2 What Does the Evidence State Contribute?

Without an evidence state, a verifier reads what it retrieves as raw text accumulating in its context, and it reaches the verdict from that text. The evidence state (Section 3.2) instead stores what was read as relevant facts, each linked to its source paper and labelled with a stance towards the claim. Stored this way, the evidence can be read in a distilled form rather than re-read as whole papers, and it persists across rounds rather than diluting as the context grows. We test whether this representation is a better input to the verdict than the papers it was extracted from, holding the retrieved papers fixed.

Settings. We vary what the recursion builds while holding the backbone and the retrieval tools fixed, in two ways. We first take the evidence ProClaim retrieved for each claim and gives it once to the same LLM with the same prompt, packaged as retrieved abstracts, full texts where available, the stance-labelled facts of ProClaim's evidence state, or facts plus abstracts. *No retrieval* is the LLM-only baseline. Every other settings gives the final state ProClaim built for the LLM in a single call, so these rows differ only in the content represented. *ProClaim* is the full loop of Table 2, which builds those states. As a control with no evidence state, *ReAct + subclaims − state* gives ReAct ProClaim's subclaims and the same retrieval tools for 20 steps, and accumulates the papers it reads in its context. *ReAct + subclaims + facts − state* adds ProClaim's fact extraction to this control, where every paper a search returns is passed, as full text where available, to the same extraction model (Qwen 3.5 9B) and prompt, and ReAct sees the extracted facts with each result. Every row is scored on randomly sampled 40 claims per subset.

Results. First, any retrieved evidence moves agreement well above no retrieval on every subset. **Second**, with the retrieved papers held fixed, the stance-labelled facts are a better input to the verdict

Table 4: Effect of the sufficiency judge with a fixed evidence state design, on the same 40 randomly sampled claims per subset. Agreement (Agr.), abstention rate (Abst.) and mean judge calls at stop (Iter.) for each stopping rule, the highest agreement per subset is underlined.

	SIGNOR (40)			**ConnectomeDB** (40)			**ExTRI** (40)			**NUBASE** (40)			**NASA** (40)		
Judge	Agr.↑	Abst.	Iter.	Agr.↑	Abst.	Iter.	Agr.↑	Abst.	Iter.	Agr.↑	Abst.	Iter.	Agr.↑	Abst.	Iter.
Stopping rules that ignore the evidence state															
Fixed rounds	0.70	0.15	3.45	0.73	0.25	3.25	0.83	0.15	3.20	0.33	0.18	2.53	0.68	0.20	3.20
Always continue	0.75	0.13	2.68	0.75	0.15	3.15	0.88	0.08	3.40	0.55	0.15	2.65	0.50	0.33	2.60
Random score	0.70	0.20	3.13	0.90	0.08	3.33	0.88	0.08	3.15	0.53	0.23	2.10	0.48	0.28	2.73
Off-the-shelf LLM judges															
Qwen3.5-9B	0.63	0.23	2.73	0.90	0.08	2.68	0.80	0.13	2.73	0.50	0.33	1.93	0.58	0.23	2.05
Claude Haiku 4.5	0.70	0.15	2.20	0.88	0.10	2.48	0.83	0.15	2.63	0.58	0.20	2.23	0.65	0.13	2.18
Claude Sonnet 4.6	0.80	0.08	2.20	0.93	0.03	2.38	0.90	0.05	2.88	0.48	0.20	2.13	0.68	0.18	2.03
Trained judge															
Trained judge	0.83	0.05	3.20	0.95	0.03	2.83	0.85	0.10	2.93	0.50	0.30	2.30	0.73	0.05	2.20

than the papers they were extracted from on four of five subsets. Pooled over the five subsets, facts give higher agreement than both abstracts and full text (0.77 vs. 0.65 and 0.70, $p = 0.024$), and lower abstention on five subsets, most on NUBASE (0.15 vs. 0.45 with abstracts). **Third**, more raw text does not substitute for the facts on our dataset. Full text is equivalent to using abstracts on SIGNOR, ConnectomeDB and NASA. It is ahead of abstracts on ExTRI (0.83 vs. 0.75) and NUBASE (0.38 vs. 0.28) but stays below facts on both. **Fourth**, ProClaim has higher agreement than both controls without an evidence state on four subsets. Pooled over the five subsets, its agreement is 0.77 against 0.60 for ReAct + subclaims − state and 0.63 for ReAct + subclaims + facts − state ($p = 0.0012$). ReAct + subclaims − state is below both raw-paper rows on ConnectomeDB (0.75), NUBASE (0.20, abstaining on 0.70 of claims) and NASA (0.43). ProClaim stays ahead of controls on SIGNOR, ConnectomeDB, NUBASE and NASA.

Finding 2: The evidence state helps, but it matters what it stores

The facts in the evidence state are a better input than the abstracts or full text (pooled agreement 0.77 vs. 0.65 and 0.70, $p = 0.024$). This is consistent with the prior finding that LLMs use long contexts unevenly (Liu et al., 2024). An agent that keeps papers only in its context does no better than reading raw papers, and ProClaim outperforms it (pooled 0.77 vs. 0.60 and 0.63 for the two controls, $p = 0.0012$).

4.3 What Does a Trained Sufficiency Judge Add over an Off-the-Shelf One?

Settings. We hold the evidence state, the backbone, the retrieval tools and the stop threshold ($\delta = 0.8$) fixed and replace only the judge J (Section 3.3). Three stopping rules ignore the state. *Fixed rounds* prompts the agent to use up the four max rounds (we enforce in the prompt but early termination is still up to the agent). *Always continue* always returns CONTINUE, leaving the agent to decide when to stop. *Random score* randomly samples stop/continue signal. Three off-the-shelf judges are LLMs prompted zero-shot on the state (Appendix D.6): Qwen 3.5 9B, Claude Haiku 4.5 and Claude Sonnet 4.6. The trained judge is fitted once on SciFact-Open and never sees ProClaim-eval. *Iterations at stop* counts the number of times the judge is called, where the max is 4. Every judge is evaluated on the same 40 claims per subset, drawn at random.

Results. First, in Table 4, the trained judge, fitted only on SciFact-Open, has the highest agreement of every stopping rule on SIGNOR, ConnectomeDB and NASA, ahead of the strongest off-the-shelf judge, Sonnet 4.6 (pooled over these three subsets 0.83 vs. 0.80, $p = 0.50$), and the lowest abstention on SIGNOR and NASA (0.05 and 0.05). It trails Sonnet 4.6 on ExTRI (0.85 vs. 0.90) and Haiku 4.5 on NUBASE (0.50 vs. 0.58). **Second**, the judges that ignore the state are also unstable across subsets. Always continuing and the random score are among the best rules on ExTRI (0.88 and 0.88) but the worst on NASA (0.50 and 0.48). **Third**, among off-the-shelf LLM judges, more capable models are often better judges. Claude Sonnet 4.6 has the highest agreement on four of five subsets, Claude Haiku 4.5 leads on NUBASE (0.58), and Qwen 3.5 9B has the lowest agreement of any stopping rule

on SIGNOR (0.63). Every off-the-shelf judge stops after fewer judge calls than the trained judge on every subset; on SIGNOR, Qwen 3.5 9B, Haiku 4.5 and Sonnet 4.6 stop after 2.73, 2.20 and 2.20 calls, against 3.20 for the trained judge. In Appendix D.5, we studied this on the full SIGNOR subset, where the off-the-shelf judges score overconfidently and the trained judge is conservative.

Finding 3: A lightweight trained judge can transfer and replace off-the-shelf LLM judges

A trained judge that reads the evidence state beats a fixed round budget using fewer steps (pooled agreement 0.77 vs. 0.65, $p = 0.0021$). A two-layer MLP judge trained only on SciFact-Open, and never exposed to ProClaim-eval, gives the highest agreement of any stopping rule on SIGNOR, ConnectomeDB and NASA, where it matches Sonnet 4.6 (pooled 0.83 vs. 0.80, $p = 0.50$) and outperforms Haiku 4.5 (0.83 vs. 0.74, $p = 0.022$), hinting at miscalibrated LLM self-evaluation (Liu & van der Schaar, 2025).

5 Related Work

Scientific claim verification datasets. Prior datasets (Diggelmann et al., 2020; Saakyan et al., 2021; Sarrouti et al., 2021; Wadden et al., 2020; 2022a) offer verdict labels for claim-passage pairs, which are not suitable for evaluating the consensus claim verification setting. While CIViC-Fact (Reisle et al., 2025) generated claims from biomedical databases, the distinction is that it also focuses on annotating claim-passage pairs. A separate work (Vittor et al., 2026) validated LLMs for entries in large biomedical knowledge graphs, whereas we focus on recent entries that are difficult for LLMs and cover non-biology domains. Matter-of-Fact (Jansen et al., 2025) is a materials science benchmark that offers claim-level labels and aims at open-retrieval verification. However, it focuses on the feasibility of claims. Recent work has highlighted the mismatch between abstract-only evidence and real verification demands. Claims may require compiling evidence from multiple sources (Thorne et al., 2018; Wadden et al., 2022a), and claims do not reliably appear in paper abstracts (Reisle et al., 2025; Wadden et al., 2022b). These limitations motivate open-domain, claim-level evaluation for consensus claim verification.

Recursive knowledge construction. Recent scientific agents (Skarlinski et al., 2024; Mitchener et al., 2025; Ghareeb et al., 2025) and search-augmented verifiers (Wei et al., 2024; Xie et al., 2025; Asai et al., 2026) run a loop in which a retrieval step's output becomes the next step's input (Zhang et al., 2025). The two design choices the loop rests on, what it builds and what controls it, are compared only implicitly. Verifiers and deep-research systems (Wei et al., 2024; Xie et al., 2025; Asai et al., 2026; Huang et al., 2026) keep the state in the growing prompt, where retrieved content is used unevenly as the context lengthens (Liu et al., 2024). Scientific agents instead persist structure outside the context, as PaperQA2 does with ranked context state and Kosmos with a world model, but none tests whether that structure provides gains, as we show in Section 4.2. The criterion for deciding when to stop the recursion is either a fixed budget (Wei et al., 2024; Asai et al., 2026) or the LLM's own self-assessment (Asai et al., 2024; Xie et al., 2025). As shown by our results in Section 4.3, a trained judge reading the state provides a clear advantage, and the LLM assessment may sometimes be miscalibrated (Wen et al., 2025; Zhou et al., 2024; Liu & van der Schaar, 2025).

6 Conclusion and Future Work

We formalise consensus scientific claim verification, a setting no prior benchmark addresses. For evaluating this verification setting, we introduce a new benchmark, ProClaim-eval, with 718 claims based on five expert-curated databases. Using ProClaim to examine one design axis at a time, we find that the evidence state helps mainly through what it stores. A trained judge that reads the evidence state outperforms fixed-round stopping on every subset, and a lightweight trained judge can transfer and replace off-the-shelf LLM judges. Together, these results support recursively updating a structured evidence state outside the model's context as an effective design for claim verification in scientific agents. For future work, we want to apply this design to curate consensus in scientific databases such as OptimusKG (Vittor et al., 2026) or biological meta-databases such as OmniPath (Turei et al., 2021), which could improve the robustness of downstream scientific analyses.

REPRODUCIBILITY STATEMENT

We explain how we set up the experiments in Section 4, and provide further details in the Appendix on experimentation and implementation. We also attach the source code for the implementation and experiments as supplementary materials.

AI DISCLOSURE

In this work, we used generative AI tools to design or provide feedback on research methodology or experiments, to implement methods, to clean and reformat the dataset, and to support qualitative and thematic data analysis and aid in interpreting results. We have not used generative AI tools to generate synthetic data sets, propose or refine hypotheses, and assist with translation. AI uses such as helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, and assisting in the writing of proofs do not apply to this work. Additionally, we used generative AI tools to create or modify scientific figures or images, create or edit software code, create artifacts, draft parts of a research paper, brainstorm, sourcing/searching for information, edit a research paper to improve readability, and identify relevant literature. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

R. L. Akeson, X. Chen, D. Ciardi, M. Crane, J. Good, M. Harbut, E. Jackson, S. R. Kane, A. C. Laity, S. Leifer, M. Lynn, D. L. McElroy, M. Papin, P. Plavchan, S. V. Ramírez, R. Rey, K. von Braun, M. Wittman, M. Abajian, B. Ali, C. Beichman, A. Beekley, G. B. Berriman, S. Berukoff, G. Bryden, B. Chan, S. Groom, C. Lau, A. N. Payne, M. Regelson, M. Saucedo, M. Schmitz, J. Stauffer, P. Wyatt, and A. Zhang. The NASA Exoplanet Archive: Data and Tools for Exoplanet Research. *Publications of the Astronomical Society of the Pacific*, 125(930):989, 2013. ISSN 1538-3873. doi: 10.1086/672273. URL https://iopscience.iop.org/article/10.1086/672273.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *The International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hSyW5go0v8.

Akari Asai, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Mike D'Arcy, David Wadden, Matt Latzke, Jenna Sparks, Jena D. Hwang, Varsha Kishore, Minyang Tian, Pan Ji, Shengyan Liu, Hao Tong, Bohao Wu, Yanyu Xiong, Luke Zettlemoyer, Graham Neubig, Daniel S. Weld, Doug Downey, Wen-tau Yih, Pang Wei Koh, and Hannaneh Hajishirzi. Synthesizing scientific literature with retrieval-augmented language models. *Nature*, pp. 1–7, 2026. ISSN 1476-4687. doi: 10.1038/s41586-025-10072-4. URL https://www.nature.com/articles/s41586-025-10072-4.

John A. Bachman, Benjamin M. Gyori, and Peter K. Sorger. Automated assembly of molecular mechanisms at scale from text mining and curated databases. *Molecular Systems Biology*, 19 (5):MSB202211325, 2023. ISSN 1744-4292. doi: 10.15252/msb.202211325. URL https://doi.org/10.15252/msb.202211325.

Jessie L. Christiansen, Douglas L. McElroy, Marcy Harbut, David R. Ciardi, Megan Crane, John Good, Kevin K. Hardegree-Ullman, Aurora Y. Kesseli, Michael B. Lund, Meca Lynn, Ananda Muthiar, Ricky Nilsson, Toba Oluyide, Michael Papin, Amalia Rivera, Melanie Swain, Nicholas D. Susemiehl, Raymond Tam, Julian van Eyken, and Charles Beichman. The NASA Exoplanet Archive and Exoplanet Follow-up Observing Program: Data, Tools, and Usage, 2025. URL http://arxiv.org/abs/2506.03299.

Xingyu Deng, Xi Wang, and Mark Stevenson. The Next Phase of Scientific Fact-Checking: Advanced Evidence Retrieval from Complex Structured Academic Papers. In *Proceedings of the 2025 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval (ICTIR)*, pp. 436–448, 2025. doi: 10.1145/3731120.3744614. URL http://arxiv.org/abs/2506.20844. arXiv:2506.20844 [cs].

Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. CLIMATE-FEVER: A Dataset for Verification of Real-World Climate Claims. In *The Tackling Climate Change with Machine Learning Workshop at NeurIPS*, 2020. URL https://arxiv.org/abs/2012.00614v2.

Ali Essam Ghareeb, Benjamin Chang, Ludovico Mitchener, Angela Yiu, Caralyn J. Szostkiewicz, Jon M. Laurent, Muhammed T. Razzak, Andrew D. White, Michaela M. Hinks, and Samuel G. Rodriques. Robin: A multi-agent system for automating scientific discovery, 2025. URL https://arxiv.org/abs/2505.13400v1.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, Artiom Myaskovsky, Grzegorz Glowaty, Felix Weissenberger, Alessio Orlandi, Dan Popovici, Anil Palepu, Keran Rong, Ryutaro Tanno, Khaled Saab, Fan Zhang, Jacob Blum, Andrew Carroll, Kavita Kulkarni, Nenad Tomašev, Dina Zverinski, Ivor Rendulic, Elahe Vedadi, Florian Hasler, Luka Rimanic, Marina Boia, Ivan Budiselic, Ben Feinstein, Mathias Bellaiche, Tom Sheffer, Jan Freyberg, Jeremy Ratcliff, Ottavia Bertolli, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R. D. Costa, José R. Penadés, Gary Peltz, Yossi Matias, James Manyika, Demis Hassabis, Yunhan Xu, Pushmeet Kohli, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Accelerating scientific discovery with Co-Scientist. *Nature*, pp. 1–3, 2026. ISSN 1476-4687. doi: 10.1038/s41586-026-10644-y. URL https://www.nature.com/articles/s41586-026-10644-y.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. A Survey on Automated Fact-Checking. *Transactions of the Association for Computational Linguistics*, 10:178–206, 2022. doi: 10.1162/tacl_a_00454. URL https://aclanthology.org/2022.tacl-1.11/.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

Rui Hou, Elena Denisenko, Huan Ting Ong, Jordan A. Ramilowski, and Alistair R. R. Forrest. Predicting cell-to-cell communication networks using NATMI. *Nature Communications*, 11 (1):5011, 2020. ISSN 2041-1723. doi: 10.1038/s41467-020-18873-z. URL https://www.nature.com/articles/s41467-020-18873-z.

Yukun Huang, Leonardo F. R. Ribeiro, Momchil Hardalov, Bhuwan Dhingra, Markus Dreyer, and Venkatesh Saligrama. DeepFact: Co-Evolving Benchmarks and Agents for Deep Research Factuality, 2026. URL http://arxiv.org/abs/2603.05912. arXiv:2603.05912 [cs] version: 1.

Peter Jansen, Samiah Hassan, and Ruoyao Wang. Matter-of-Fact: A Benchmark for Verifying the Feasibility of Literature-Supported Claims in Materials Science. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 4090–4102, Suzhou, China, 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.203. URL https://aclanthology.org/2025.emnlp-main.203/.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Mexico City, Mexico, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.389. URL https://aclanthology.org/2024.naacl-long.389/.

Hailey Joren, Jianyi Zhang, Chun-Sung Ferng, Da-Cheng Juan, Ankur Taly, and Cyrus Rashtchian. Sufficient Context: A New Lens on Retrieval Augmented Generation Systems. In *The International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Jjr2Odj8DJ.

F. G. Kondev, M. Wang, W. J. Huang, S. Naimi, and G. Audi. The NUBASE2020 evaluation of nuclear physics properties. *Chinese Physics C*, 45(3):030001, 2021. doi: 10.1088/1674-1137/abddae.

Sujit Kumar, Anshul Sharma, Siddharth Hemant Khincha, Gargi Shroff, Sanasam Ranbir Singh, and Rahul Mishra. SciClaimHunt: A Large Dataset for Evidence-based Scientific Claim Verification, 2025. URL http://arxiv.org/abs/2502.10003. arXiv:2502.10003 [cs].

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. ISSN 2307-387X. doi: 10.1162/tacl_a_00638. URL https://doi.org/10.1162/tacl_a_00638.

Peiwen Liu, Sakura Eri B Maezono, Weitao Lin, Yen Yeow, Ya-Yu Liu, Yasmin Hisham, Rui Hou, Jordan A Ramilowski, and Alistair R R Forrest. connectomeDB2025: a rigorously curated, multi-species resource of experimentally supported ligand–receptor interactions. *Nucleic Acids Research*, 54(D1):D546–D554, 2026. ISSN 1362-4962. doi: 10.1093/nar/gkaf1108. URL https://doi.org/10.1093/nar/gkaf1108.

Tennison Liu and Mihaela van der Schaar. Position: Truly Self-Improving Agents Require Intrinsic Metacognitive Learning. 2025. URL https://openreview.net/forum?id=4KhDd0Ozqe.

Paola Lo Surdo, Marta Iannuccelli, Krystal Karis, Elena Meo, Paniz Omidi, Marisa Tosoni, Serena Graziosi, Simona Panni, Maria Teresa Fiorillo, Luana Licata, Francesca Sacco, Benjamin M. Gyori, and Livia Perfetto. SIGNOR 4.0: the 2025 update with focus on phosphorylation data. *Nucleic Acids Research*, nov 2025. doi: 10.1093/nar/gkaf1237. gkaf1237, Epub ahead of print.

Ludovico Mitchener, Angela Yiu, Benjamin Chang, Mathieu Bourdenx, Tyler Nadolski, Arvis Sulovari, Eric C. Landsness, Daniel L. Barabasi, Siddharth Narayanan, Nicky Evans, Shriya Reddy, Martha Foiani, Aizad Kamal, Leah P. Shriver, Fang Cao, Asmamaw T. Wassie, Jon M. Laurent, Edwin Melville-Green, Mayk Caldas, Albert Bou, Kaleigh F. Roberts, Sladjana Zagorac, Timothy C. Orr, Miranda E. Orr, Kevin J. Zwezdaryk, Ali E. Ghareeb, Laurie McCoy, Bruna Gomes, Euan A. Ashley, Karen E. Duff, Tonio Buonassisi, Tom Rainforth, Randall J. Bateman, Michael Skarlinski, Samuel G. Rodriques, Michaela M. Hinks, and Andrew D. White. Kosmos: An AI Scientist for Autonomous Discovery, 2025. URL http://arxiv.org/abs/2511.02824. arXiv:2511.02824 [cs].

Sophia Müller-Dott, Eirini Tsirvouli, Miguel Vazquez, Ricardo O Ramirez Flores, Pau Badia-i Mompel, Robin Fallegger, Dénes Türei, Astrid Lægreid, and Julio Saez-Rodriguez. Expanding the coverage of regulons from high-confidence prior knowledge for accurate estimation of transcription factor activities. *Nucleic Acids Research*, 51(20):10934–10949, 2023. ISSN 0305-1048. doi: 10.1093/nar/gkad841. URL https://doi.org/10.1093/nar/gkad841.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. Scispacy: fast and robust models for biomedical natural language processing. In *Proceedings of the 18th BioNLP workshop and shared task*, pp. 319–327, 2019.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 3982–3992, 2019.

Caralyn Reisle, Cameron J. Grisdale, Kilannin Krysiak, Arpad M. Danos, Mariam Khanfar, Erin Pleasance, Jason Saliba, Melika Hanos, Nilan V. Patel, Asmita Jain, Joshua F. McMichael, Ajay C. Venigalla, Malachi Griffith, Obi L. Griffith, and Steven J. M. Jones. Evaluating Language Models for Biomedical Fact-Checking: A Benchmark Dataset for Cancer Variant Interpretation Verification, 2025. URL https://www.biorxiv.org/content/10.1101/2025.09.10.675443v1. ISSN: 2692-8205.

Arkadiy Saakyan, Tuhin Chakrabarty, and Smaranda Muresan. COVID-Fact: Fact Extraction and Verification of Real-World Claims on COVID-19 Pandemic. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 2116–2129, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.165. URL https://aclanthology.org/2021.acl-long.165/.

Mourad Sarrouti, Asma Ben Abacha, Yassine Mrabet, and Dina Demner-Fushman. Evidence-based Fact-Checking of Health-related Claims. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 3499–3512, Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.297. URL https://aclanthology.org/2021.findings-emnlp.297/.

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

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a Large-scale Dataset for Fact Extraction and VERification. In Marilyn Walker, Heng Ji, and Amanda Stent (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 809–819, New Orleans, Louisiana, 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1074. URL https://aclanthology.org/N18-1074/.

J. K. Tuli. Evaluated nuclear structure data file. *Nuclear Instruments and Methods in Physics Research Section A*, 369:506–510, 1996. doi: 10.1016/S0168-9002(96)80040-4.

Denes Turei, Alberto Valdeolivas, Lejla Gul, Nicolas Palacio-Escat, Michal Klein, Olga Ivanova, Marton Olbei, Attila Gabor, Fabian Theis, Dezso Modos, Tamas Korcsmaros, and Julio Saez-Rodriguez. Integrated intra- and intercellular signaling knowledge for multicellular omics analysis. *Molecular Systems Biology*, 2021. doi: 10.15252/msb.20209923.

Sander van der Linden. Misinformation: susceptibility, spread, and interventions to immunize the public. *Nature Medicine*, 28(3):460–467, 2022. ISSN 1546-170X. doi: 10.1038/s41591-022-01713-6. URL https://www.nature.com/articles/s41591-022-01713-6.

Miguel Vazquez, Martin Krallinger, Florian Leitner, Martin Kuiper, Alfonso Valencia, and Astrid Laegreid. ExTRI: Extraction of transcription regulation interactions from literature. *Biochimica et Biophysica Acta (BBA) - Gene Regulatory Mechanisms*, 1865(1):194778, 2022. ISSN 1874-9399. doi: 10.1016/j.bbagrm.2021.194778. URL https://www.sciencedirect.com/science/article/pii/S1874939921000961.

Lucas Vittor, Ayush Noori, Iñaki Arango, Joaquín Polonuer, Sam Rodriques, Andrew White, David A. Clifton, and Marinka Zitnik. Unifying biomedical knowledge in a modern multimodal graph, 2026. URL http://arxiv.org/abs/2604.27269.

Juraj Vladika and Florian Matthes. Scientific Fact-Checking: A Survey of Resources and Approaches. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 6215–6230, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.387. URL https://aclanthology.org/2023.findings-acl.387/.

Juraj Vladika and Florian Matthes. Comparing Knowledge Sources for Open-Domain Scientific Claim Verification. In Yvette Graham and Matthew Purver (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2103–2114, St. Julian's, Malta, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.128. URL https://aclanthology.org/2024.eacl-long.128/.

Juraj Vladika, Ivana Hacajova, and Florian Matthes. Step-by-Step Fact Verification System for Medical Claims with Explainable Reasoning. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 805–816, Albuquerque, New Mexico, 2025. Association for Computational Linguistics. ISBN 979-8-89176-190-2. doi: 10.18653/v1/2025.naacl-short.68. URL https://aclanthology.org/2025.naacl-short.68/.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or Fiction: Verifying Scientific Claims. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7534–7550, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.609. URL https://aclanthology.org/2020.emnlp-main.609/.

David Wadden, Kyle Lo, Bailey Kuehl, Arman Cohan, Iz Beltagy, Lucy Lu Wang, and Hannaneh Hajishirzi. SciFact-Open: Towards open-domain scientific claim verification. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 4719–4734, Abu Dhabi, United Arab Emirates, 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.347. URL https://aclanthology.org/2022.findings-emnlp.347/.

David Wadden, Kyle Lo, Lucy Lu Wang, Arman Cohan, Iz Beltagy, and Hannaneh Hajishirzi. MultiVerS: Improving scientific claim verification with weak supervision and full-document context. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (eds.), *Findings of the Association for Computational Linguistics: NAACL 2022*, pp. 61–76, Seattle, United States, 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.6. URL https://aclanthology.org/2022.findings-naacl.6/.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. Executable code actions elicit better LLM agents. In *Proceedings of the International Conference on Machine Learning*, volume 235 of *ICML'24*, pp. 50208–50232, Vienna, Austria, 2024a. JMLR.org.

Yuxia Wang, Revanth Gangi Reddy, Zain Muhammad Mujahid, Arnav Arora, Aleksandr Rubashevskii, Jiahui Geng, Osama Mohammed Afzal, Liangming Pan, Nadav Borenstein, Aditya Pillai, Isabelle Augenstein, Iryna Gurevych, and Preslav Nakov. Factcheck-Bench: Fine-Grained Evaluation Benchmark for Automatic Fact-checkers. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 14199–14230, Miami, Florida, USA, 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.830. URL https://aclanthology.org/2024.findings-emnlp.830/.

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Zixia Hu, Jie Huang, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, and Quoc V. Le. Long-form factuality in large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=4M9f8VMt2C.

Bingbing Wen, Jihan Yao, Shangbin Feng, Chenjun Xu, Yulia Tsvetkov, Bill Howe, and Lucy Lu Wang. Know Your Limits: A Survey of Abstention in Large Language Models. *Transactions of the Association for Computational Linguistics*, 13:529–556, 2025. doi: 10.1162/tacl_a_00754. URL https://aclanthology.org/2025.tacl-1.26/.

Jevin D. West and Carl T. Bergstrom. Misinformation in and about science. *Proceedings of the National Academy of Sciences*, 118(15):e1912444117, 2021. doi: 10.1073/pnas.1912444117. URL https://www.pnas.org/doi/10.1073/pnas.1912444117.

Valerie Wood, Matt Jeffryes, Andrew F Green, Matthias Blum, Sandra Orchard, Simona Panni, Federica Quaglia, Raul Rodriguez-Esteban, James Seager, Silvio C E Tosatto, Ulrike Wittig, and Melissa Harrison. Empowering biological knowledgebases: advances in human-in-the-loop AI-driven literature curation. *Bioinformatics Advances*, 6(1):vbag028, 2026. ISSN 2635-0041. doi: 10.1093/bioadv/vbag028. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC12904773/.

Zhuohan Xie, Rui Xing, Yuxia Wang, Jiahui Geng, Hasan Iqbal, Dhruv Sahnan, Iryna Gurevych, and Preslav Nakov. FIRE: Fact-checking with Iterative Retrieval and Verification. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 2901–2914, Albuquerque, New Mexico, 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.158. URL https://aclanthology.org/2025.findings-naacl.158/.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *Proceedings of the International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Alex L. Zhang, Tim Kraska, and Omar Khattab. Recursive Language Models, 2025. URL http://arxiv.org/abs/2512.24601.

Lexin Zhou, Wout Schellaert, Fernando Martínez-Plumed, Yael Moros-Daval, Cèsar Ferri, and José Hernández-Orallo. Larger and more instructable language models become less reliable. *Nature*, 634(8032):61–68, 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-07930-y. URL https://www.nature.com/articles/s41586-024-07930-y.

TABLE OF CONTENTS

A Limitations. **17**

B ProClaim-eval Details. **17**

B.1 Verdict Label Definitions. 17

B.2 Data Generation Process: SIGNOR 17

B.3 Data Generation Process: ConnectomeDB 19

B.4 Data Generation Process: ExTRI 19

B.5 Data Generation Process: NUBASE 20

B.6 Data Generation Process: NASA Exoplanet Archive 20

B.7 Data Release 20

C Experimental details **20**

C.1 Experimental Results in Section 4.1 20

C.2 Significance Tests for the Findings 21

C.3 LLM Thinking Mode and Temperature 21

C.4 Computational Resources and Cost 22

C.5 Prompts for Compared Methods in Section 4.1 23

D ProClaim Details **23**

D.1 Implementation 23

D.2 ProClaim Tools and Operations 24

D.3 Dataset for training the sufficiency judge 24

D.4 Sufficiency judge Implementation 25

D.5 Judge Ablation: How Does the Choice of Judge Affect ProClaim's Performance? . 25

D.6 LLM Judge Prompt 26

D.7 ProClaim Prompts 27

E Prompts for Baselines and Compared Methods **32**

A LIMITATIONS.

Results are based on a single frontier model, Claude Sonnet 4.6, at the time when we started our investigation. Owing to the resource constraints, we only run extensive analyses on a subset of ProClaim-eval in Section 4.2 and 4.3. ProClaim-eval covers factual claim sentences from three domains instead of free-form claims. While we select the best suitable databases, the gold labels can inherit the coverage and update lag of the curated databases they come from. Our comparisons focus on one design axis at a time within ProClaim, but ProClaim still differs in prompt and uses several times more compute, and the judge comparison uses 40 claims per subset, so small differences should be read as indicative rather than conclusive. Finally, the trained judge relies on features and training data from biomedicine (SciFact-Open) mainly, and its transfer to other fields is shown only on the NASA and NUBASE subsets.

B PROCLAIM-EVAL DETAILS.

B.1 VERDICT LABEL DEFINITIONS.

Scientific literature, especially in biological and biomedical domains, does not constitute ground truth. Individual papers report findings under specific experimental conditions, and the same phenomenon may be reported differently across studies depending on methodology, statistical power, or the state of knowledge at the time of publication. As a result, verification should not be reduced to a binary decision. Systems restricted to SUPPORT/REFUTE labels risk systematically misclassifying claims when evidence is inconclusive, internally conflicting, or incomplete.

We adopt a three-way verdict taxonomy motivated by the epistemology of scientific hypothesis testing. In this framework, a claim carries the burden of proof, where the initial position of a claim is non-acceptance, and the evidence must affirmatively establish the claim's validity for a SUPPORT verdict to be warranted. This is analogous to the logic of null hypothesis significance testing, where the null is maintained until sufficient evidence is gathered to reject it. In particular, we group active contradiction and substantive absence under the REFUTE verdict, which concerns evidential warrant for accepting the claim, not a metaphysical assertion that the claim is false. The UNCERTAIN verdict is distinct from SUPPORT and REFUTE in that relevant evidence exists and engages with the claim, but fails to resolve it.

Verdict definition

We used the same verdict definition (below) in our experimentation for baselines and ProClaim. Minor modifications are made to the LLM-only setting when no retrieval is performed; there would be no external evidence.

- **SUPPORT:** The retrieved evidence contains statements that directly corroborate the claim. The evidence, taken at face value, is sufficient to conclude that the claim is true or highly likely true.
- **REFUTE:** Either (a) the retrieved evidence contains statements that directly contradict the claim, or (b) given the scope of the retrieved corpus, a thorough search yields no evidence that substantiates the claim. In both cases, the evidence base does not support accepting the claim as true.
- **UNCERTAIN (abstention):** The retrieved evidence is relevant to the claim but is ambiguous, incomplete, or internally conflicting such that neither a clear supportive nor a clear refutatory conclusion can be drawn. This includes cases where evidence partially supports the claim but with meaningful caveats, or where sources of comparable credibility disagree.

B.2 DATA GENERATION PROCESS: SIGNOR

We constructed one subset of ProClaim-eval from the SIGNOR database (Lo Surdo et al., 2025), a manually curated resource of experimentally supported causal interactions between biological entities. Each claim is expressed as a natural-language statement of the form "*{Protein A} directly*

activates/inhibits {Protein B}" This subset comprises two complementary edge sets: a set of ground-truth incorrect edges that should be removed, and a set of ground-truth correct edges that serve as positive controls.

REFUTE examples. We established the set of ground-truth incorrect edges by systematically comparing SIGNOR releases spanning 2018 to 2025. Candidate removals were identified by tracking yearly changes in both SIGNOR IDs and entity names to filter out administrative identifier reassignments. To ensure these removals reflect deliberate curation decisions rather than artifacts of entity deprecation or database restructuring, we retained only those edges whose constituent entities persisted in the October 2025 release.

SUPPORT examples. To construct a matched set of positive edges, we filtered the October 2025 SIGNOR release (26,127 interactions). We prioritized 3,887 interactions involving proteins from the incorrect edge set, assuming that these entities had undergone more rigorous expert scrutiny. These candidates were ranked by their reference counts in the OmniPath database (Turei et al., 2021) and passed through a deduplication pipeline that resolved bidirectional redundancies and selected the most-referenced edges for each source and target node. To ensure the benchmark's challenge, we excluded 'easy' cases, where the edges are correctly classified by a small language model (gpt-oss-20b) across five independent trials. Following final human expert adjudication, 65 edges were confirmed as the expert-curated base set: 35 SUPPORT and 30 REFUTE.

Claim augmentation via effect flipping. To increase the number of REFUTE examples beyond those derived from historically removed edges, we augmented the dataset by flipping the effect sign of 36 base edges (changing "activates" to "inhibits" or vice versa). For 33 of the 35 SUPPORT edges, the original edge is confirmed correct, so the sign-flipped claim is labelled REFUTE. For 3 REFUTE edges, the adjudication found the curated effect sign to be wrong, so the sign-flipped claim is labelled SUPPORT. This subset therefore comprises 38 SUPPORT and 63 REFUTE, for a total of 101 claims.

Verdict curation protocol. We recruited three experts with molecular biology or computational biology background for curation. To ensure the curation process has the same level of standard, we provide human curators with a structured guideline for validating signed directed protein–protein interactions (PPIs) sourced from existing databases.

1. **Claim parsing**: extract source protein, target protein, sign (activation/inhibition), and mechanism.
2. **Protein characterisation**: identify functional class, known aliases, and gene family membership using authoritative databases (e.g., UniProt, NCBI Gene).
3. **Confusable name check**: flag lexically similar protein names (e.g., CDK2 vs. CK2) that are a major source of annotation error in text-mined databases.
4. **Biological plausibility assessment**: evaluate whether the source protein's functional class is consistent with the claimed mechanism and cellular context.
5. **Literature evidence search**: conduct systematic searches using gene symbols, aliases, and identified confusable names, examining retrieved papers for both supporting evidence and potential mis-annotation sources (e.g., shared regulators, pathway co-membership).
6. **Evidence quality evaluation**: assess experimental methodology, directness, specificity, and sign/direction consistency.
7. **Verdict assignment**: classify each interaction with justifications as true, false.

We decided the final verdict based on majority voting over independent expert annotations, where each annotator assigns a stance label to an edge. The majority label is taken as the provisional decision, and cases without a clear majority or with substantive disagreement are escalated to adjudication. During adjudication, experts review the underlying evidence and resolve disagreements to produce a single consensus label.

B.3 Data Generation Process: ConnectomeDB

Claim creation. We included ligand-receptor interactions involved in cell-cell communication based on the ConnectomeDB database. The most recent two releases were in 2020 (Hou et al., 2020) and 2025 (latest) (Liu et al., 2026). We constructed this subset by converting curated interaction records from ConnectomeDB into a uniform natural-language claim format "*{Ligand} as ligand directly interacts extracellularly with {Receptor} as receptor*" and for each claim, we extracted the primary ligand and receptor gene symbols. Then, we attached one verdict per claim as described below.

SUPPORT examples. SUPPORT examples are created from the newly added interaction set in the 2025 ConnectomeDB release with direct literature support from at least two publications.

REFUTE examples. REFUTE examples are derived from data in the 2020 release that were removed during the 2025 curation process. We retain only curator-rejected data with specific mechanistic grounds for rejection and exclude entries without primary evidence due to the ambiguity of which evidence was considered. The remaining rejected pairs are labelled deterministically from the curator's comments. Cases such as intracellular interactions, in-cis interactions, wrong directionality, retracted papers, or non-protein interactions are mapped to REFUTE, while cases labelled "old paper without gene symbol" are dropped. Importantly, all examples use the same affirmative claim template.

Dataset. The final dataset contains 305 claims: 184 SUPPORT, 121 REFUTE. It combines 184 positive ligand-receptor entries from the ConnectomeDB 2025 release with 121 expert-curation rejected entries from the 2020 release. Each instance includes the claim, label, expert annotator comment, ligand, and receptor symbols and identifiers. For positive examples, provenance is taken from the Perplexity AI summary of the 2025 release curation, which embeds the supporting PubMed identifiers. For rejected examples, provenance includes the curator comment and the PMID of the originally cited publication. The positive subset is supported by literature spanning multiple species, with 145 human, 36 mouse, and 3 rat interactions, while the rejected subset is human-curated and removed in the ConnectomeDB 2025 release.

B.4 Data Generation Process: ExTRI

Claim creation. We built this subset from the curated sentence corpus behind ExTRI (Vazquez et al., 2022), 47,373 sentences on transcription factor (TF) to target-gene regulation, each labelled independently by at least two domain experts with disagreements resolved to consensus. We did not use the text-mined ExTRI resource, whose edges are machine output. We kept sentences marked valid that passed quality control, excluded the complex identifiers NFKB and AP1, and grouped the remaining 24,256 sentences into 11,354 TF–target pairs. Each claim uses the template "*{TF} directly activates/represses transcription of {target}*", with the direction taken from the curated regulation sign, so the wording does not encode the label.

SUPPORT examples. A pair is SUPPORT when all of its curated sentences agree on one regulation sign and none negates the regulation. We further required at least five independent supporting papers, so that a SUPPORT claim reflects accumulated evidence rather than a single report.

REFUTE examples. A pair is REFUTE when every curated sentence for it states that the regulation does not occur (a curated negation), with nothing affirming it. The claim states the curated direction affirmatively, and the label follows deterministically from the negation. Curated negations are almost always single-paper, so REFUTE claims are exempt from the five-paper requirement.

Filtering. To keep claims independent, we selected them so that no paper is shared between two claims (1,504 candidates). We removed 'easy' claims that a small language model (gpt-oss-20b) answers correctly in all of five trials, removed a claim whose curated sentences are all non-mammalian, and kept at most two claims per TF for each label.

Dataset. The final subset contains 125 claims: 49 SUPPORT and 76 REFUTE. Each instance includes the claim, label, the curated sign or negation, the supporting PubMed identifiers, and the TF and target symbols.

B.5 Data Generation Process: NUBASE

Claim creation. We built this subset from NUBASE2020 (Kondev et al., 2021), the evaluated table of nuclear ground- and isomeric-state properties (5,868 records). A claim asserts that a nuclear isomeric state exists, named by its reported half-life: "*{Element}-{A} has an isomeric state with a half-life of about {value} {unit}*". Two REFUTE states carry only an upper limit and are phrased "...with a half-life shorter than {value} {unit}". Nuclides are written with the element name (e.g. Cobalt-52), and units are spelled out.

SUPPORT examples. SUPPORT examples are measured isomeric states discovered in 2015 or later. Of 84 candidates, we removed 7 that gpt-oss-20b answers correctly in all of five trials.

REFUTE examples. REFUTE examples are isomeric states that NUBASE2020 flags as reported but non-existent (states claimed in the literature and later disproved). The label follows from this flag. States flagged as of uncertain existence were excluded.

Dataset. The final subset contains 85 claims: 77 SUPPORT and 8 REFUTE. Each instance includes the claim, label, the NUBASE2020 status, the nuclide and the state, with the NUBASE2020 publication as evidence.

B.6 Data Generation Process: NASA Exoplanet Archive

Claim creation. We built this subset from the dispositions of the NASA Exoplanet Archive (Akeson et al., 2013; Christiansen et al., 2025), where every candidate planet carries a curated status. All claims use the template "*{object} is a planet orbiting the star {star}*", and the label follows deterministically from the archive's disposition.

SUPPORT examples. SUPPORT examples are planets confirmed in 2024 or later, not flagged as controversial, and backed by at least two distinct publications (excluding the archive's link to its own follow-up service). As for the other subsets, we removed claims that gpt-oss-20b answers correctly in all of five trials.

REFUTE examples. REFUTE examples come from K2 mission objects in two groups. All 8 planets that were published and later refuted, with their refuting papers attached, and 39 of the 242 K2 false positives. We excluded pending candidates, false positives without stated grounds, and false-positive flags from automated vetting pipelines, whose adjudication is not in the citable literature.

Dataset. The final subset contains 102 claims: 55 SUPPORT and 47 REFUTE. Each instance includes the claim, label, the archive disposition, the object and its host star, and the supporting references.

B.7 Data Release

To preserve the dataset's value as an evaluation benchmark, we release the full ProClaim-eval in a serialized binary format rather than as plain text. This prevents the dataset from being ingested during web-crawl-based LLM pretraining, which would risk data contamination and undermine its utility for future evaluation. Researchers can load the dataset programmatically using our provided loading scripts. We believe this release strategy balances open access for reproducibility with long-term benchmark integrity.

C Experimental details

C.1 Experimental Results in Section 4.1

Table 5 reports macro-F1 and balanced accuracy for the same runs as Table 2.

Table 5: Macro-F1 (F1) and balanced accuracy (BAcc) per ProClaim-eval subset, for the same run behind each cell of Table 2. *Random* is the mean over its 10 draws. The shaded figure beside each macro-F1 is ProClaim's macro-F1 minus that row's, on the same subset; green where ProClaim is ahead, red where it is behind. The highest value per column is underlined.

	SIGNOR		ConnectomeDB		ExTRI		NUBASE		NASA	
Method	F1↑	BAcc↑	F1↑	BAcc↑	F1↑	BAcc↑	F1↑	BAcc↑	F1↑	BAcc↑
Random	0.38	0.32	0.40	0.33	0.38	0.32	0.32	0.34	0.37	0.31
LLM Only										
Claude Sonnet 4.6	0.68 +0.17	0.59	0.60 +0.19	0.56	0.31 +0.47	0.25	0.05 +0.49	0.03	0.11 +0.59	0.06
Gemini 2.5 Flash	0.66 +0.19	0.57	0.64 +0.16	0.58	0.31 +0.47	0.28	0.28 +0.25	0.39	0.32 +0.38	0.43
Static Retrieval, S2 Top 5										
Claude Sonnet 4.6	0.56 +0.29	0.57	0.48 +0.31	0.55	0.45 +0.33	0.51	0.16 +0.38	0.35	0.46 +0.23	0.35
Gemini 2.5 Flash	0.49 +0.36	0.54	0.51 +0.29	0.58	0.44 +0.34	0.52	0.18 +0.36	0.55	0.48 +0.22	0.37
Adaptive Retrieval										
ReAct	0.79 +0.06	0.73	0.74 +0.06	0.73	0.80 -0.02	0.73	0.16 +0.37	0.10	0.66 +0.04	0.54
FIRE	0.72 +0.13	0.69	0.51 +0.29	0.56	0.69 +0.09	0.66	0.09 +0.45	0.33	0.72 -0.02	0.71
OpenScholar	0.50 +0.35	0.35	0.61 +0.19	0.53	0.37 +0.41	0.25	0.03 +0.51	0.01	0.05 +0.65	0.03
General-Purpose Agents										
Claude Code	0.82 +0.03	0.79	0.76 +0.04	0.75	0.86 -0.08	0.82	0.24 +0.29	0.18	0.63 +0.06	0.54
Codex	0.86 -0.01	0.82	0.75 +0.04	0.76	0.83 -0.05	0.78	0.20 +0.34	0.16	0.64 +0.05	0.56
ProClaim	0.85	0.81	0.79	0.77	0.78	0.75	0.54	0.51	0.70	0.65

C.2 Significance Tests for the Findings

Each finding statement is tested with a paired comparison of two systems, A and B, on the same claims. The five statements are: (1) the trained judge beats the Haiku 4.5 judge and matches the Sonnet 4.6 judge on SIGNOR, ConnectomeDB and NASA, and (2) the trained judge beats fixed rounds (Finding 3); (3) facts beat abstracts and full text and (4) ProClaim beats in-context accumulation (Finding 2); and (5) ProClaim beats the SOTA harnesses Claude Code and Codex (Finding 1). The unit is a claim scored by both systems. A verdict is correct (1) if it matches the gold label and incorrect (0) otherwise. UNCERTAIN, missing and errored answers count as 0. Pooled over the statement's subsets, b counts the claims A gets right and B gets wrong, and c the claims B gets right and A gets wrong. We apply an exact two-sided binomial McNemar test to b against c, and a Holm correction across the five statements. A statement that compares A with two systems holds only if A beats both, so its p-value is the larger of its two tests (an intersection–union test). Every comparison uses the runs shown in the corresponding table. Statements (2)–(5) pool all five subsets. Statement (1) pools the 40-claim samples of SIGNOR, ConnectomeDB and NASA. Its comparison with Sonnet 4.6 is reported with its two-sided p-value and is not part of the Holm correction. Table 6 reports, for each comparison, the number of claims n, the discordant counts b and c, the difference Δ between A's and B's pooled agreement, the exact p-value, and the statement's Holm-adjusted p-value.

C.3 LLM Thinking Mode and Temperature

All methods, including ProClaim and each baseline, were evaluated with thinking/reasoning mode disabled and a fixed LLM temperature of 0. Disabling the thinking mode ensures that the models operate under comparable inference budgets and that any performance differences could inform the architectural and methodological choices rather than variations in internal chain-of-thought reasoning afforded by extended thinking tokens. Setting the temperature to 0 maximizes the reproducibility of model outputs and comparisons.

Table 6: Paired exact McNemar tests of the finding statements, pooled over subsets. n: claims; b: claims A gets right and B gets wrong; c: claims B gets right and A gets wrong; Δ: A's pooled agreement minus B's; p: exact two-sided McNemar p-value; p_{Holm}: the statement's p-value (the larger of its tests), Holm-adjusted across the five statements.

Statement: A vs. B	**Subsets**	n	b	c	Δ	p	p_{Holm}
(1) Trained judge vs. Haiku 4.5 judge	3×40	120	13	2	+0.092	0.0074	0.022
Trained judge vs. Sonnet 4.6 judge		120	12	8	+0.033	0.50*	
(2) Trained judge vs. Fixed rounds	5×40	200	35	11	+0.120	<0.001	0.0021
(3) Facts vs. Abstracts	5×40	200	30	7	+0.115	<0.001	0.024
Facts vs. Full text		200	24	10	+0.070	0.024	
(4) ProClaim vs. ReAct + subclaims − state	5×40	200	47	12	+0.175	<0.001	0.0012
ProClaim vs. ReAct + subclaims + facts − state		200	42	14	+0.140	<0.001	
(5) ProClaim vs. Claude Code	5, full	718	99	65	+0.047	0.0098	0.022
ProClaim vs. Codex		718	104	68	+0.050	0.0074	

Table 7: Mean input (In) and output (Out) tokens per claim, in thousands, for the run behind each cell of Table 2. Input tokens include cached input. ProClaim counts its main agent only.

	SIGNOR		**ConnectomeDB**		**ExTRI**		**NUBASE**		**NASA**	
Method	In	Out	In	Out	In	Out	In	Out	In	Out
LLM Only										
Claude Sonnet 4.6	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.1
Gemini 2.5 Flash	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.1
Static Retrieval, S2 Top 5										
Claude Sonnet 4.6	2.1	0.1	2.4	0.2	2.3	0.2	1.2	0.1	1.1	0.1
Gemini 2.5 Flash	1.8	0.1	2.0	0.1	1.9	0.1	1.1	0.1	1.0	0.0
Adaptive Retrieval										
ReAct	81.8	1.5	91.4	1.7	94.8	1.5	261.2	2.3	213.3	1.6
FIRE	3.6	1.1	3.3	1.1	4.9	1.2	5.8	1.0	7.1	0.9
OpenScholar	15.2	2.4	17.0	3.1	15.4	2.5	10.7	1.5	12.8	1.5
General-Purpose Agents										
Claude Code	347.5	5.6	134.0	3.9	336.9	4.6	642.9	6.0	272.0	3.6
Codex	322.2	4.5	270.2	3.8	102.4	1.9	529.6	4.1	246.4	2.1
ProClaim	626.8	13.5	612.1	14.0	721.0	14.7	1157.9	18.3	902.1	14.8

C.4 Computational Resources and Cost

All experiments were run on an internal HPC cluster using 8 NVIDIA A100 (80 GB) GPUs, 8 CPU cores, and 64 GB RAM. We served a local Qwen 3.5 9B instance for fact extraction and called the Anthropic API (Claude Sonnet 4.6) as the main agent. For claims from ConnectomeDB (305 cases), total GPU time was ≈40 A100 GPU-hours. For claims from SIGNOR (101 cases), the total GPU time was ≈12 A100 GPU-hours. For claims from ExTRI (125 cases), NUBASE (85 cases) and NASA (102 cases), the total GPU time was ≈13, ≈20 and ≈17 A100 GPU-hours, respectively. Total GPU time for the main evaluations (Section 4.1) is ≈102 A100 GPU-hours.

Our architecture follows a modular design where computationally intensive but routine subtasks are handled by a locally hosted model at zero marginal API cost, while the main agent leverages a frontier model for high-level planning. Since the subagent accounts for the majority of LLM calls per claim, hosting it locally on institutional hardware substantially reduces the per-claim cost. The main agent could alternatively use a locally hosted model to reduce API costs.

The primary temporal bottleneck is the per-paper NLP feature extraction required by the sufficiency judge (Section D.4). For each newly retrieved paper, the pipeline computes named entity coverage (via `en_core_sci_sm` (Neumann et al., 2019), ≈15 MB), semantic similarity (via SBERT `all-MiniLM-L6-v2` (Reimers & Gurevych, 2019), ≈80 MB), and natural language inference scores (via `cross-encoder/nli-deberta-v3-large` (He et al., 2021), ≈1.5 GB), which together account for approximately 40% of total wall time per claim. The main agent (Claude Sonnet 4.6) contributes roughly 25% of wall time, with an average latency of approximately 30 seconds per agent step. Fact extraction via the local Qwen 3.5 9B subagent accounts for ≈10% of wall time.

Table 7 reports the mean input and output tokens per claim for every run in Table 2. Since ProClaim's main agent re-sends its full execution log (accumulated tool calls and outputs), the input context grows substantially over a claim's trajectory. With prompt caching, most of this repeated prefix is billed at the cached-input rate rather than the full input rate.

C.5 Prompts for Compared Methods in Section 4.1

We have included updated prompts for all compared methods in Appendix E. When adapting SAFE, FIRE and OpenScholar, we made minimal revisions to the prompts via Claude Sonnet 4.6 for ProClaim-eval's label definitions.

D ProClaim Details

D.1 Implementation

We implement ProClaim in a persistent notebook-style execution environment in which the agent alternates between narrative reasoning and executable evidence updates. The concrete runtime uses domain-specific Python tools for literature search, fact extraction, and stance labelling, while preserving the abstract loop defined in Section 3.

State representation. The evidence state is implemented as a structured data model with JSON persistence. In addition to retrieved papers, it stores extracted facts with source identifiers and text spans, detected conflicts between papers, optional subclaim decompositions, and the history of sufficiency scores. Because the state persists after each update, the agent can accumulate evidence across many steps without depending on the context window to remember previously processed papers.

Execution layer. The primitive operations are realised as Python functions executed in separate kernel sessions. In our current implementation, this includes literature search over PubMed and Semantic Scholar, batched fact extraction and stance assignment, and evidence filtering utilities that reduce neutral or irrelevant papers before sufficiency assessment. Search operations return paper metadata and abstracts. Extraction operations create fact-stance pairs from abstracts or full text, if available. The execution log transcript accumulated so far (markdown reasoning, executed code cells, and their outputs) is sent as user message to the LLM at each step, so the agent can condition its decisions on the full history of evidence operations and their results.

Programmatic control. Running inside an executable kernel allows the agent to compose evidence operations with loops, conditionals, and targeted subroutines. This is important when verification requires addressing unresolved subclaims or launching gap-driven retrieval conditioned on the current state. We implemented full paper text access since abstracts may not always be sufficient to resolve a claim (Reisle et al., 2025; Wadden et al., 2022b). Our implementation is restricted to accessing full texts through licensed APIs (Unpaywall and Elsevier) and does not redistribute available full-text content.

Audit trail. The notebook also functions as a reproducible trace of the verification process. Executed code cells record evidence-gathering actions and printed outputs, while interleaved markdown cells capture the agent's intermediate reasoning. The saved notebook, therefore, exposes both the final evidence state and the sequence of decisions that produced it.

Caching. The execution log's length directly determines the size of the input context the model must process. The log is append-only, where each iteration's reasoning, executed code cells, and outputs are appended without modifying or removing earlier entries, so the prompt sent to the agent at step t is always an exact prefix of the prompt at step $t + 1$. This stability lets us mark the system prompt and the unchanged prefix of the execution log with cache breakpoints. For Claude Sonnet 4.6, this allows repeated turns to reuse cached prefixes at a $0.1\times$ read price (with a $1.25\times$ write surcharge on first use). This significantly reduces per-claim costs without affecting the agent's behaviour or its verdicts.

Hyperparameters. The hyperparameter δ gives a threshold of evidence sufficiency for ProClaim to emit a verdict. For all of our experiments, δ is set to 0.8. The maximum number of iterations $T_{\max}$ is set to 4 for all experiments. The hyperparameter τ controls the conflict ratio threshold for sufficiency judge training labels (Section D.4), and it is set to 0.50 for the main experiment.

D.2 ProClaim Tools and Operations

At the outer loop, ProClaim is given only two generic tools: `bash` and `read_file`. We refer to all functionality exposed to ProClaim as the evidence API, and the Python function schema is injected into the system prompt. This design encourages ProClaim to solve the task through lightweight code generation over a stable, structured interface rather than through a large collection of specialized tools. The `read_file` tool complements this setup by allowing ProClaim to inspect persisted artifacts such as `evidence_state.json`, `verdict.json`, and large intermediate outputs when standard output is insufficient.

Below, we describe the main functions available to ProClaim. Each subprocess begins with `setup_workspace()`, which initializes the verification session, loads the evidence state, and constructs handles for the LLM subagents. This makes every subprocess idempotent, so ProClaim can resume from disk on every step without depending on hidden conversational state. To retrieve papers, ProClaim uses retrieval functions such as `search_pubmed_llm()` and `search_semantic_scholar()`, which return identifiers for potentially relevant papers. The function `extract_and_add_facts()` then retrieves available full text and converts it into stance-labelled evidence records stored in the evidence state. Each fact remains linked to its source paper and is stored in a structured format that can be reused across iterations.

Later iterations can target missing evidence directly through `search_for_gap()`, where ProClaim writes the query. The direct setting also exposes broader corpus expansion functions such as `search_semantic_scholar_recommendations()`, which let ProClaim move beyond PubMed keyword search via Semantic Scholar Recommendations seeded with papers already judged relevant. Seeds are derived automatically from papers that produced at least one SUPPORT fact. Papers with only REFUTE facts are used as negative seeds to steer results away from irrelevant directions. Once candidate papers are identified, `get_paper_text()` or `get_full_text_article()` retrieves the available paper text for downstream extraction.

Before deciding whether it has enough evidence, ProClaim calls `populate_paper_features()`. This function computes the natural language and metadata-derived feature representation consumed by the sufficiency judge. The sufficiency decision itself is made by `check_sufficiency()` to return a confidence score. In addition, this automatically invokes LLM-based gap findings from the current evidence state. When the accumulated evidence is sufficient, ProClaim finalizes the run with `emit_verdict()`, which writes a structured verdict object containing the final label, reasoning, and supporting evidence references. If the evidence remains insufficient, ProClaim continues the loop by reading the latest sufficiency output or sufficiency history, issuing new gap-targeted searches, and repeating the extraction and feature-population steps.

D.3 Dataset for training the sufficiency judge

We intend to label a state SUFFICIENT when the evidence is very clear and unanimous, while any conflicting support or refute findings or added noise makes it INSUFFICIENT. That means SUFFICIENT here does not mean there is strong evidence, but that the current evidence state is a clean, internally

consistent stopping state. So the model is trained to withhold a positive label unless the whole pool looks settled.

We train the MLP on the SciFact-Open dataset, which contains 279 claims with labels assigned at the claim-paper pair level. Of these, 206 claims have at least one evidence paper labelled as either SUPPORT or REFUTE, and 73 claims have no supporting evidence. To quantify the degree of conflict between supporting and refuting evidence, we define the minority ratio as $r_{\text{minority}} = \min(N_{\text{support}}, N_{\text{refute}})/(N_{\text{support}} + N_{\text{refute}})$ when at least one paper provides a stance (i.e., $N_{\text{support}} + N_{\text{refute}} > 0$). If $r_{\text{minority}} \geq \tau$, the state is labelled insufficient ($y = 0$) and contains conflicting support/contradict evidence or irrelevant noise papers sampled for claims without evidence annotations. For the 73 NEI claims, we perform data augmentation by randomly appending 1 to 3 irrelevant papers and labelling the resulting states as insufficient to train the model's ability to recognise irrelevant evidence. Otherwise, the state is categorized as sufficient ($y = 1$). In Section 3.3, we use $\tau = 0.5$ as a simple criterion to decide conflicting states as they have equal amount of supporting and refuting evidence.

For each claim-paper pair, we extracted metadata features from PubMed/OpenAlex, including publication year, journal impact proxy, normalized citation count, and maximum author h-index, together with NLP features computed from the paper full text when available, otherwise from the abstract. The NLP features are claim–evidence semantic similarity, biomedical entity coverage, and NLI entailment/contradiction/neutral scores, computed over overlapping text chunks for long documents.

D.4 SUFFICIENCY JUDGE IMPLEMENTATION

For each paper $P_i \in \mathcal{E}$, we extracted a feature vector $\mathbf{v}_i = \psi(\mathcal{C}, P_i) = [\mathbf{v}_i^{\text{meta}}, \mathbf{v}_i^{\text{nlp}}] \in \mathbb{R}^k$ combining publication quality metadata with NLP-based signals. These local vectors are aggregated into a global representation $\mathbf{v}_{\text{global}} = g(\mathbf{v}_1, \ldots, \mathbf{v}_N) \in \mathbb{R}^d$ via pooling and cross-weighting statistics that summarize corpus-level consensus and controversy. These paper-level signals are aggregated into a 10-dimensional state representation.

We train a two-hidden-layer MLP with hidden dimensions 64 and 32 using class-weighted binary cross-entropy to predict the sufficiency label,

$$\hat{y} = f_\phi(\mathcal{C}, \mathcal{E}) = \sigma(\text{MLP}(\mathbf{v}_{\text{global}}))\,,$$

where $\mathbf{v}_{\text{global}}$ denotes the aggregated feature representation of the current evidence state for claim $\mathcal{C}$. We use a stratified $70/15/15$ train/validation/test split. At inference time, the model outputs a sufficiency score for the full evidence state, which serves as the stopping criterion in the recursive retrieval loop.

The sufficiency MLP classifier was trained to reward clean, consensus-like states. Its input is a 10-dimensional vector pooled over the papers in the state: (1) mean claim–paper similarity, (2) mean entity coverage, (3) contradiction ratio (the fraction of papers whose most likely NLI class is contradiction), (4) controversy index (the entropy of the papers' NLI stance votes), (5) recency-weighted entailment ($\sum_i e^{-0.1\,a_i}\, p_i^{\text{ent}}$, with a_i the paper's age in years), (6) mean normalised citation count, (7) mean log impact factor, (8) year span, (9) number of papers, and (10) number of full texts. To ensure reliability of the sufficiency signal, a sufficient label is forced back to insufficient when the state holds fewer than 3 papers, except in the last round. This fallback applies to every judge that can return sufficient, including the off-the-shelf LLM judges and the random score in Section 4.3. This design aims at making the whole system keep searching unless the evidence state is both strong and structurally clean. From our experiments, we fourrnd that the sufficiency labels are not monotonic. Adding papers can lower consensus features or increase controversy, so a run can become sufficient and then fall back to insufficient.

D.5 JUDGE ABLATION: HOW DOES THE CHOICE OF JUDGE AFFECT PROCLAIM'S PERFORMANCE?

We compare the trained judge $J(c, s_t)$ with two off-the-shelf LLM judges, Qwen 3.5 9B and Claude Haiku 4.5, prompted zero-shot on the same evidence state (Appendix D.6), on the full SIGNOR subset. All other settings follow Section 4.3. Agreement and abstention are computed as in the main

Table 8: Judge ablation on SIGNOR (101 claims), with the evidence state, backbone, retrieval tools and stop threshold held fixed. Agreement (Agr., abstention counted as a disagreement), abstention rate (Abst.), macro F1 over the two gold labels, mean judge calls at stop (Iter.), and precision (P), recall (R) and F1 per gold label.

					SUPPORT			REFUTE		
	Agr.↑	Abst.	Macro F1↑	Iter.	P	R	F1	P	R	F1
Trained judge	0.80	0.12	0.85	3.21	0.82	0.84	0.83	0.98	0.78	0.87
Qwen3.5 9B	0.71	0.18	0.78	2.63	0.78	0.82	0.79	0.95	0.65	0.77
Claude Haiku 4.5	0.76	0.11	0.81	2.48	0.74	0.92	0.82	0.98	0.67	0.79

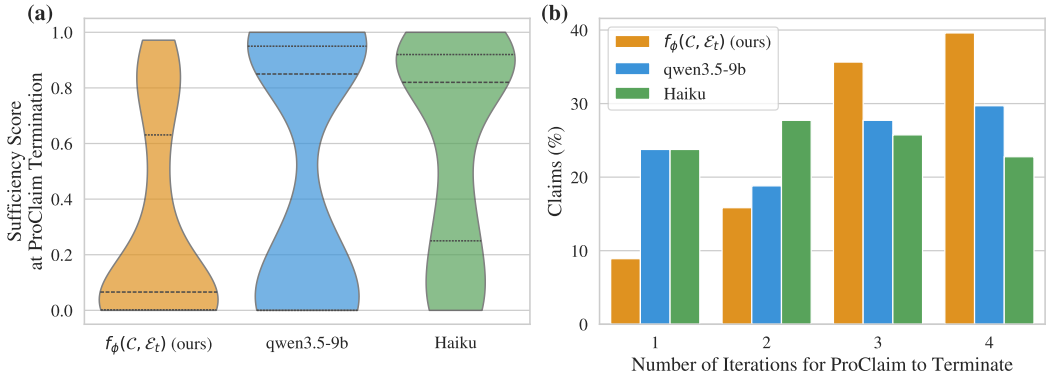

Figure 4: Effect of the judge on sufficiency scoring and ProClaim termination. **(a)** The trained judge is conservative, assigning low scores for most claims and driving the agent to collect more evidence across multiple rounds. Qwen 3.5 9B exhibits a bimodal distribution, with a dominant mode near 1.0 and a secondary mode near 0.0. Haiku assigns uniformly high scores with almost no mass near zero. **(b)** The trained judge's conservative scoring terminates ProClaim at three and four iterations, while Qwen 3.5 9B's bimodal pattern produces peaks at both one and four iterations, and Haiku distributes more evenly.

text, and precision (P), recall (R) and F1 are computed for the two gold labels. Table 8 shows that the trained judge has the highest agreement (0.80 vs. 0.71 for Qwen 3.5 9B and 0.76 for Haiku 4.5) and macro F1 (0.85 vs. 0.78 and 0.81), with the highest REFUTE recall (0.78 vs. 0.65 and 0.67), while Haiku 4.5 has the highest SUPPORT recall (0.92). The trained judge also keeps the recursion running longest (3.21 judge calls at stop vs. 2.63 and 2.48). Both LLM judges assign higher sufficiency scores than the trained judge on average (Figure 4a). Haiku 4.5's scores cluster near 1.0, the systematic overconfidence discussed in prior work (Wen et al., 2025; Zhou et al., 2024), and Qwen 3.5 9B's are bimodal, near one or near zero for most claims. The trained judge is fitted on SciFact-Open, which covers general biomedical topics, and is used on SIGNOR without domain-specific fine-tuning.

D.6 LLM JUDGE PROMPT

To evaluate the impact of replacing our MLP sufficiency judge with an LLM, we designed an ablation study using an LLM-based sufficiency judge. The LLM is prompted to act as a scientific evaluator. It assesses whether the provided reference context contains sufficient evidence to support or refute the claim. The complete prompt template used for this LLM-based evaluator is presented below.

Prompt Template: LLM sufficiency judge

You are a scientific evaluator. Given a CLAIM and a REFERENCE (paper titles, abstracts, and NLP features), assess whether the REFERENCE contains sufficient evidence to support or refute the CLAIM.

Assign a sufficiency score between 0.0 and 1.0:

- **1.0**: The REFERENCE contains decisive evidence — either clearly supporting OR clearly refuting the CLAIM. Both directions are equally "sufficient". Papers that unanimously refute the claim deserve the same high score as papers that unanimously support it.
- **0.0**: The REFERENCE is entirely irrelevant (entities, topic, and semantics are unrelated to the claim), or no papers were found.
- Values in between reflect partial relevance, inconclusive evidence, or genuinely conflicting evidence. Conflicting evidence means the papers are relevant and credible, but split in direction — after weighing their content, methodology, and credibility, no clear verdict can be reached. Do NOT judge conflict by the ratio of papers alone; a single high-quality paper can outweigh several weaker ones.

FEATURE DEFINITIONS
Each paper in the REFERENCE includes pre-computed Metadata and NLP fields:

Metadata:

- `publication_year`: Year the paper was published.
- `log_impact_factor`: log(1 + journal impact factor). Higher = more prestigious venue.
- `normalized_citation_count`: Citations / paper age. Measures community attention.
- `author_h_index_max`: Highest H-index among all authors. Proxy for author credibility.

NLP (computed relative to the CLAIM):

- `claim_entity_coverage`: Fraction of claim biomedical entities also in the paper (0–1).
- `semantic_similarity`: SBERT cosine similarity between claim and abstract (0–1).
- `nli_entailment`: DeBERTa NLI probability that the `nli_best_chunk_text` ENTAILS the claim (0–1).
- `nli_contradiction`: Probability the `nli_best_chunk_text` CONTRADICTS (refutes) the claim (0–1).
- `nli_neutral`: Probability the `nli_best_chunk_text` is NEUTRAL (0–1). Sum ≈ 1.
- `nli_best_chunk_text`: The most relevant text passage used to compute NLI scores.

CLAIM
{claim}

REFERENCE
{reference}

First, output a brief **### EXPLANATION** (plain text reasoning).
Then output **### EVALUATION** as a JSON object with exactly these keys:

```
{
"sufficiency_score":  <float 0.0-1.0>,
"reasoning":  "<one-sentence summary>"
}
```

Output ONLY the EXPLANATION block followed by the EVALUATION JSON block. No other text.

D.7 ProClaim Prompts

ProClaim System Prompt

```
You are an evidence-programming agent that verifies scientific
claims and produces verdicts [{verdict_names}].
{verdict_definitions}

You work by writing Python code executed via the `bash` tool.  The
`command`
argument you pass to `bash` must be the literal shell command
```

````
`python3 -c '...'` with your Python code inside the quotes -- NOT
the bare
Python code by itself (bare Python code passed directly as the
shell command
will fail with `command not found` / syntax errors, since bash does
not know
Python syntax).  All evidence API functions are available as
Python imports.

## First step -- set up

Call `bash` with the setup code to bootstrap the workspace. The
`command`
argument must look exactly like this (a single shell string,
`python3 -c`
followed by your code in quotes):

```bash
python3 -c '
from proclaim.verification.evidence_api import setup_workspace
state, llm, workspace = setup_workspace(
    claim="{claim}",
    workspace_path="{workspace}",
)
print("Ready")
'
```

After this, every subsequent bash call must re-load state from disk
(there is no persistent kernel).  Use setup_workspace which is
idempotent
(loads existing state if present). Again, wrap it in `python3 -c
'...'`:

```bash
python3 -c '
from proclaim.verification.evidence_api import setup_workspace
state, llm, workspace = setup_workspace(claim="{claim}",
workspace_path="{workspace}")
# ... your evidence API calls here ...
'
```

{function_docs}

{schemas}

## Workflow

1. Call bash with the setup code above.
2. Decompose the claim into 1-5 atomic subclaims, each a single
independently
   verifiable assertion. Use 1 (the original claim) if the claim
   is already simple
   enough to search directly. Include alternative names or aliases
   for key entities.
   Set subclaims before searching: `state.subclaims = [...]` then
   `state._auto_save()`.
{subclaim_examples}
3. Search (iteration 0):
   a. call search_pubmed_llm(state.claim, state, llm) -- runs two
   LLM-generated PubMed queries (claim-only and
   subclaim-enriched), deduplicated
````

```
   b. call search_semantic_scholar_dual(state.claim, state, llm)
   -- runs two S2 queries (claim-only and subclaim-enriched),
   covers bioRxiv preprints and non-MEDLINE journals
4. Extract facts: call extract_and_add_facts(llm, pmids, state,
max_workers=8)
  to process all newly retrieved papers in parallel.  Do NOT loop
  over paper IDs.
5. Call populate_paper_features(state) -- REQUIRED before
check_sufficiency().
6. Call filter_papers_by_stance(state) -- removes papers with only
neutral facts.
7. Check sufficiency: result = check_sufficiency(state, llm);
print(result)
8. Call get_sufficiency_history(state) to monitor the confidence
trend.
9. If insufficient: use gap-targeted retrieval:
   a. search_for_gap(gap_description, state)
   b. search_semantic_scholar_recommendations(state) -- S2 graph
   expansion
   c. formulate_gap_queries(llm, state) -- LLM-generated gap
   queries
10. Repeat until confidence >= {sufficiency_threshold} or
{max_iterations} iterations.{sufficiency_mode_note}
11. Call emit_verdict(...) to produce the final verdict.

## Rules

- Use bash for ALL evidence API calls -- write Python code.
- Print results to stdout so you can see them.
- ALL state mutations are auto-saved to evidence_state.json.
- `state` does NOT persist across bash calls -- reload it each time
  (or call setup_workspace again).
- When emit_verdict is called, the verification is complete.
- Do NOT attempt to debug, patch, or work around evidence API
functions that return 0 facts. A 0-fact result means the paper
lacks relevant evidence or accessible full text -- not a tool bug.
Move on to other papers or emit a verdict.

## CRITICAL: Grounded Evidence Only

- NEVER fabricate facts.  Every fact must come from a retrieved
paper.
- Use extract_and_add_facts(llm, pmids, state) -- pass the full
paper ID list.
- If extraction returns 0 for multiple paper IDs, use
refine_search_for_failed_papers:
  failed_ids = [paper_id for paper_id, count in results.items() if
  count == 0]
  new_ids = refine_search_for_failed_papers(failed_ids, state,
  llm, max_new_papers=5)
  results2 = extract_and_add_facts(llm, new_ids, state)
- NEVER call add_facts_from_dicts with manually written text.
- If no papers contain relevant evidence, say so in the verdict.
- add_extraction_context_note is for SYNONYM/ALIAS MAPPINGS ONLY.
Never write
  search goals, task descriptions, or paper-specific findings into
  it.

## Feature Computation for MLP Classifier

You MUST call populate_paper_features(state) after extracting
facts and
```

```
before check_sufficiency().  Without it, the MLP classifier
receives
all-zero features and predictions will be inaccurate.

Required sequence in EVERY iteration:
1. search (PubMed + Semantic Scholar)
2. extract_and_add_facts(llm, pmids, state)
3. populate_paper_features(state)           ← REQUIRED
4. filter_papers_by_stance(state)           ← REQUIRED (removes
neutral/irrelevant papers)
5. check_sufficiency(state, llm)

## Important

- Workspace path: {workspace}
- Max iterations: {max_iterations}
- Sufficiency threshold: {sufficiency_threshold}

## Common Python Mistakes to Avoid

- When building the `reasoning` string for emit_verdict, compute
all values
  BEFORE constructing the string.  Use f-strings (prefix with `f`)
  or
  `.format()`, never bare curly braces in a regular string.
  BAD:  reasoning = "Found {{len(facts)}} facts"   -- literal
  braces, not evaluated
  GOOD: n = len(facts); reasoning = f"Found {{n}} facts"  --
  f-string evaluates n
```

ProClaim Fact Extraction

```
You are a scientific fact extraction specialist.

Given a paper and a claim with subclaims, extract every atomic
fact that addresses the claim, accounting for varying terminology,
synonyms, or aliases used in the text.

For each fact provide a JSON object with:
- "text": factual statement (one sentence, self-contained)
- "stance": one of {stance_options}
- "source_pmid": "{source_pmid}"
- "confidence": 0.0-1.0, how clearly the paper states this

Example: [{{"text": "Gs alpha directly stimulates adenylyl cyclase
activity.", "stance": "SUPPORT", "source_pmid": "{source_pmid}",
"confidence": 0.9}}]

Stance definitions:
{stance_block}

Rules:
- Base facts strictly on the provided text. Recognize equivalent
terms, but do not hallucinate logical leaps not present in the
paper.
- If the paper has no relevance to the claim or subclaims, return
[].
- Each fact must be independently verifiable from the source paper.
Now extract facts for the following:

Claim: {claim}
```

```
Subclaims:
{subclaims_str}
{context_block}
Paper (paper ID: {source_pmid}):
{paper_text}

Output ONLY a JSON array of fact objects. No other text.
```

ProClaim PubMed Query Generation

```
You are a biomedical search expert. Generate a comprehensive
PubMed query for the following scientific claim:

Claim: {claim}

Requirements:
- Include key entities (genes, mutations, diseases, drugs,
proteins)
- Add synonyms and common variants using OR operators where
appropriate
- Use appropriate field tags when beneficial (e.g.,
[Title/Abstract], [MeSH])
- Keep query length reasonable (5-20 words)
- Use proper PubMed syntax with Boolean operators (AND, OR, NOT)
- Use parentheses to group related terms

Examples:
- Diagnosis claim: "JAK2 V617F is not associated with lymphoid
leukemia"
  → (JAK2 AND V617F) AND (lymphoid leukemia OR B-ALL OR T-ALL OR
  CLL)

- Drug resistance claim: "NT5C2 K359Q mutation does not confer
resistance to Nelarabine"
  → (NT5C2 AND K359Q) AND (Nelarabine OR Arabinosylguanine) AND
  (resistance OR sensitivity)

- PPI claim: "MAPK1 activates H3-3A through phosphorylation"
  → (MAPK1 OR ERK2) AND (H3-3A OR HIST1H3A) AND (phosphorylation
  OR activation)

Output ONLY the query string without any explanation or markdown
formatting.
```

ProClaim Semantic Scholar Query Generation

```
You are a biomedical search expert. Generate a Semantic Scholar
query for the following scientific claim:

Claim: {claim}{subclaims_section}

Requirements:
- Use plain keyword phrases (Semantic Scholar does not support
PubMed field tags)
- Include alternative names, gene symbols, aliases, and common
synonyms for key entities
- Keep query concise (3-15 words)
- Prefer terms that appear in titles and abstracts of relevant
papers
```

```
Examples:
- PPI claim: "MAPK1 activates H3-3A through phosphorylation"
  → MAPK1 ERK2 H3.3 phosphorylation activation

- Drug resistance: "NT5C2 K359Q mutation does not confer
resistance to Nelarabine"
  → NT5C2 K359Q nelarabine arabinosylguanine resistance

Output ONLY the query string without any explanation or markdown
formatting.
```

ProClaim Gap-Targeted Query Generation

```
Generate a focused PubMed query to find evidence for this gap:

Claim: {claim}
Gap: {gap_description}

Generate a query that specifically targets this missing evidence.
Keep it concise (3-10 words).
Output ONLY the query string without explanation.
```

E PROMPTS FOR BASELINES AND COMPARED METHODS

Static retrieval searches S2 with the claim text. On NUBASE it searches the nuclide followed by "isomer half-life" (e.g. "Cobalt-52 isomer half-life"), because the full claim sentence returns no paper for 83 of the 85 claims.

Retrieval Final Verdict System Prompt

```
You are a scientific claim verification expert.

Given a claim and retrieved evidence passages, assign one of the
following verdicts:

- SUPPORT -- The retrieved evidence contains statements that
directly corroborate the claim. The evidence, taken at face value,
is sufficient to conclude that the claim is true or highly likely
true.
- REFUTE -- Either (a) the retrieved evidence contains statements
that directly contradict the claim, or (b) given the scope of the
retrieved corpus, a thorough search yields no evidence that
substantiates the claim. In both cases, the evidence base does not
support accepting the claim as true.
- UNCERTAIN -- The retrieved evidence is relevant to the claim but
is ambiguous, incomplete, or internally conflicting such that
neither a clear supportive nor a clear refutatory conclusion can
be drawn. This includes cases where evidence partially supports
the claim but with meaningful caveats, or where sources of
comparable credibility disagree.

Respond with valid JSON only -- no markdown fences, no extra keys:
{
    "label": "SUPPORT" | "REFUTE" | "UNCERTAIN",
    "reasoning": "One or two sentences citing specific evidence",
    "evidence": ["PMID1", "PMID2"]
}
```

Retrieval Final Verdict User Template

```
Claim: {claim}

Retrieved Evidence:
{evidence}

Based on the above evidence, classify the claim. Output JSON.
```

LLM-Only Final Verdict System Prompt

```
You are a scientific claim verification expert.

Given a claim, assess whether it is supported by established
scientific knowledge:

- SUPPORT -- Your knowledge of the scientific literature supports
the claim as true or highly likely true based on established
experimental evidence.
- REFUTE -- Your knowledge of the scientific literature
contradicts the claim, or the claim is not substantiated by any
known experimental evidence.
- UNCERTAIN -- The scientific literature is ambiguous, incomplete,
or conflicting on this claim, or your knowledge is insufficient to
make a reliable determination.

Respond with valid JSON only -- no markdown fences, no extra keys:
{
    "label": "SUPPORT" | "REFUTE" | "UNCERTAIN",
    "reasoning": "One or two sentences explaining your verdict"
}
```

LLM-Only Final Verdict User Template

```
Claim: {claim}

Based on your scientific knowledge, classify the claim. Output
JSON.
```

ReAct System Prompt (Semantic Scholar)

```
You are a scientific claim verification agent using the ReAct
framework.

Your task: determine whether a scientific claim is "SUPPORT" or
"REFUTE" or "UNCERTAIN" based on evidence you retrieve from
academic literature.

Verdict definitions:
   - "SUPPORT" -- The retrieved evidence contains statements that
   directly corroborate the claim. The evidence, taken at face
   value, is sufficient to conclude that the claim is true or
   highly likely true.
   - "REFUTE" -- Either (a) the retrieved evidence contains
   statements that directly contradict the claim, or (b) given the
   scope of the retrieved corpus, a thorough search yields no
   evidence that substantiates the claim. In both cases, the
   evidence base does not support accepting the claim as true.
```

```
   - "UNCERTAIN" -- The retrieved evidence is relevant to the
   claim but is ambiguous, incomplete, or internally conflicting
   such that neither a clear supportive nor a clear refutatory
   conclusion can be drawn. This includes cases where evidence
   partially supports the claim but with meaningful caveats, or
   where sources of comparable credibility disagree.

Available tools:
- search_papers(query): Search Semantic Scholar for academic
papers. Returns titles, abstracts, and metadata. Use specific
queries targeting the entities and relationships in the claim.

Strategy:
1. Think step-by-step about what evidence you need.
2. Issue targeted search queries to find that evidence.
3. After each search result, assess whether you have enough
evidence.
4. When confident, respond with your final verdict in the EXACT
format:
   VERDICT: <label>
   REASONING: <one or two sentences citing key evidence>
```

ReAct User Prompt

```
Verify this scientific claim:

{claim}
```

FIRE System Prompt

```
You are a fact-checking agent responsible for verifying the
accuracy of claims.
```

FIRE Final Answer Or Next Search Prompt

```
Instructions:
1. You are provided with a STATEMENT and relevant KNOWLEDGE points.
2. Based on the KNOWLEDGE, assess the factual accuracy of the
STATEMENT.
3. Before presenting your conclusion, think through the process
step-by-step.
   Include a summary of the key points from the KNOWLEDGE as part
   of your reasoning.
4. If the KNOWLEDGE allows you to confidently make a decision,
output the final
   answer as a JSON object in the following format:
   {
      "final_answer": "SUPPORT" or "REFUTE" or "UNCERTAIN"
   }
    - "SUPPORT" -- The retrieved evidence contains statements that
    directly corroborate the claim. The evidence, taken at face
    value, is sufficient to conclude that the claim is true or
    highly likely true.
    - "REFUTE" -- Either (a) the retrieved evidence contains
    statements that directly contradict the claim, or (b) given
    the scope of the retrieved corpus, a thorough search yields no
    evidence that substantiates the claim. In both cases, the
    evidence base does not support accepting the claim as true.
```

```
    - "UNCERTAIN" -- The retrieved evidence is relevant to the
    claim but is ambiguous, incomplete, or internally conflicting
    such that neither a clear supportive nor a clear refutatory
    conclusion can be drawn. This includes cases where evidence
    partially supports the claim but with meaningful caveats, or
    where sources of comparable credibility disagree.

5. If the KNOWLEDGE is insufficient to make a judgment, issue ONE
PubMed and Semantic Scholar search
   query that could provide additional evidence. Output the search
   query in JSON
   format, as follows:
   {
     "search_query": "Your PubMed and Semantic Scholar search
     query here"
   }
6. The query should aim to obtain new information not already
present in the
   KNOWLEDGE, specifically helpful for verifying the STATEMENT's
   accuracy.

KNOWLEDGE:
[KNOWLEDGE]

STATEMENT:
[STATEMENT]
```

OpenScholar Claim Framing

```
Is the following scientific claim supported or refuted by the
literature: {claim}? Answer uncertain otherwise.
```

OpenScholar Claim Verdict Question

```
Given a scientific question about whether a claim is supported or
refuted and a set of retrieved reference passages, output a
verdict as valid JSON with exactly these keys:
{
  "label": "SUPPORT" | "REFUTE" | "UNCERTAIN",
  "reasoning": "<one or two sentences citing specific reference
  numbers>",
  "evidence": ["[0]", "[1]"]
}
Verdict definitions:
- SUPPORT: The retrieved passages contain statements that directly
corroborate the claim. The evidence, taken at face value, is
sufficient to conclude that the claim is true or highly likely
true.
- REFUTE: Either (a) the retrieved passages contain statements
that directly contradict the claim, or (b) given the scope of the
retrieved corpus, a thorough reading yields no evidence that
substantiates the claim. In both cases the evidence base does not
support accepting the claim as true.
- UNCERTAIN: The retrieved passages are relevant to the claim but
are ambiguous, incomplete, or internally conflicting such that
neither a clear supportive nor a clear refutatory conclusion can
be drawn. This includes cases where evidence partially supports
the claim but with meaningful caveats, or where sources of
comparable credibility disagree.
Rules:
```

```
- Base your verdict ONLY on the provided references. Do not use
outside knowledge.
- If a passage explicitly states the opposite of the claim, that
is strong evidence for REFUTE.
- If no passage mentions the entities or relationship in the
claim, prefer REFUTE over UNCERTAIN.
- Output valid JSON only. No markdown fences, no extra keys, no
prose outside the JSON object.
Question: Is the following scientific claim supported or refuted
by the literature: {claim}? Answer uncertain otherwise.
```