Plato's Cave: A Human-Centered Research Verification System

Matheus Kunzler Maldaner[1*], Raul Valle[1*], Junsung Kim[1], Tonuka Sultan[1], Pranav Bhargava[1], Matthew Maloni[1], John Courtney[1], Hoang Nguyen[1], Aamogh Sawant[2], Kristian O'Connor[1], Stephen Wormald[1], Damon L. Woodard[1]

[1]University of Florida
[2]Georgia Institute of Technology
{mkunzlermaldaner, rvalle1, dwoodard}@ufl.edu

Abstract

The growing publication rate of research papers has created an urgent need for better ways to fact-check information, assess writing quality, and identify unverifiable claims. We present Plato's Cave as an open-source, human-centered research verification system that (i) creates a directed acyclic graph (DAG) from a document (ii) leverages web agents to assign credibility scores to nodes and edges from the DAG and (iii) gives a final score by interpreting and evaluating the paper's argumentative structure. We report the system implementation and results on a collected dataset of 104 research papers.

1 Introduction

Scientific publishing continues to scale faster than traditional review capacity. In 2022, there were 3.3 million research publications in science and engineering (National Science Board 2023) and in the field of artificial intelligence, this pressure has been increasing. Popular machine learning conferences such as NeurIPS, ICML and ICLR received more than 60,000 combined submissions in the last cycle alone (NeurIPS 2025 Program Chairs 2025; Paper Copilot 2026). This proliferation has strained traditional peer review mechanisms and allowed the spread of unreliable research. The total number of articles in the Retraction Watch Database increased 26% in 2025, now exceeding 63,000, and high-profile cases of fabricated data have undermined confidence in published findings (Retraction Watch 2024, 2025). This situation calls for the development of tools that can help researchers and reviewers rapidly audit the internal logic of papers as structured arguments where claims, methods, evidence, and conclusions are complex and interconnected.

Existing approaches towards research verification fall into three categories. (1) Citation-based metrics such as the h-index and impact factors of the journals measure popularity rather than correctness and may distort performance targets that should prioritize logic (Strathern 1997), (2) fact-checking systems such as SciFact (Wadden et al. 2020) and FEVER (Thorne et al. 2018) operate at the sentence level and fail to capture the argumentative structure that connects claims to evidence and (3) approaches to pure language models lack an in-depth knowledge of external sources and cannot reliably verify claims against primary sources (Liu and Shah 2023).

We address these limitations by posing a system in which scientific papers are modeled as knowledge graphs, nodes represent semantic units (hypotheses, claims, evidence, methods, results), and edges encode their relationships. Our system then traverses the graph and verifies individual claims using autonomous browser-use agents that search the web to assess the credibility of sources by extracting supporting or contradicting evidence. The verified node qualities propagate through the graph structure via a trust-gating mechanism: when a parent node has low trust, it reduces confidence in all dependent child nodes, ensuring that weak foundational claims appropriately diminish confidence in their downstream linked nodes. This propagation step is inspired by message-passing inference in probabilistic graphical models (i.e. belief propagation / sum-product), but here it propagates verification-derived trust rather than probabilistic marginals (Pearl 1988; Kschischang, Frey, and Loeliger 2001; Yedidia, Freeman, and Weiss 2003; Wainwright and Jordan 2008).

Our contributions include:

- A role-aware DAG representation for papers, together with a validation-and-repair step that enforces ontology and acyclicity constraints.
- A multi-level scoring framework that combines the quality of individual nodes, edge-level priors and semantic alignment, and global structure metrics (bridge coverage, best-path reliability, redundancy, fragility, coherence, and coverage).
- An efficient factorized sampling scheme that separately perturbs structure (K sampled DAGs) and aggregation parameters (M resampled node/propagation weights), enabling uncertainty estimates without retraining.
- A pilot system evaluation using 104 research papers showing end-to-end throughput, failure modes, and a moderate alignment with coarse predetermined labels.
- An open-source interactive system that exposes our pipeline to the user, enabling human-in-the-loop auditing rather than replacing reviewer judgment.

The remainder of this paper proceeds as follows: Section 2 reviews related work in claim verification, knowledge graphs, trust propagation, and argumentative structure in scientific

[*]These authors contributed equally, ordered by date of birth.

documents. Section 3 formalizes our approach towards the paper verification problem. Section 4 describes our implementation including the knowledge graph ontology, multi-agent verification pipeline, and tech stack used on our system design. Section 5 presents the pilot evaluation and the results of the cache-first calibration. Section 6 discusses strengths, limitations, and future work. Finally, Section 7 concludes.

2 Related Work

Research verification intersects several strands of prior work, spanning the verification of scientific claims, construction of knowledge graphs, trust propagation, and the emergence of multi-agent systems for automated information gathering. We structure this section around these themes to situate Plato's Cave within the broader landscape of present literature.

Scientific Claim Verification

Automated fact-checking has received significant attention in natural language processing. The FEVER dataset introduced the task of verifying claims against Wikipedia (Thorne et al. 2018), a study that led to numerous neural architectures for evidence retrieval and entailment classification. SciFact (Wadden et al. 2020) adapted this paradigm to scientific claims, pairing assertions with relevant abstracts and stance labels (i.e. supports, refutes, not_enough_info). However, these systems operate at the sentence level and treat verification as an isolated classification task rather than modeling the argumentative structure of complete papers.

Other studies have explored claim decomposition (Liu, Datta, and Lim 2015) and multi-hop reasoning (Yang et al. 2018), but these approaches still focus on question-answering rather than holistic paper assessment. Our work differs by extracting the complete dependency graph of a specific paper's arguments and propagating trust through this structure on top of verifying claims independently.

Knowledge Graph Construction

Automated knowledge graph construction from text has progressed through rule-based and supervised relation (Cheng et al. 2024; Liu 2020), and more recently, LLM-based approaches (Zhu et al. 2024). Scientific knowledge graphs like Open Academic Graph (Zhang et al. 2019), which seeks to unify the Microsoft Academic Graph (Sinha et al. 2015) and AMiner (Tang et al. 2008), as well as Semantic Scholar's API (Kinney et al. 2023), focus on citation networks and metadata (such as authors, institutions, field of study) rather than the internal argumentative structure of papers.

Argumentation mining (Lawrence and Reed 2019) seeks to identify claim-premise relationships in text, but existing approaches focus on pre-defined structures, such as internet forums or student essays. Our work therefore extends these ideas to scientific papers, introducing a domain-specific ontology of 10 semantic roles and enforcing structural constraints like acyclicity and role-specific parent-child relationships.

Prior work on argumentative structure in scientific writing is also directly relevant here as existing studies analyze such structures in scientific articles (Kirschner, Eckle-Kohler, and Gurevych 2015), introduce argument-annotated corpora for scientific publications (Lauscher, Glavaš, and Ponzetto 2018), and survey argument mining for processing academic documents (Al Khatib et al. 2021). These lines of work motivate our role-aware document representation, with Plato's Cave differing by coupling document-structure extraction to external verification, trust propagation, and a reviewer-facing score rather than treating it as a standalone parsing task.

Trust Propagation Methods

In probabilistic graphical models, belief propagation (BP) is a canonical message-passing algorithm for propagating local evidence through a graph to compute marginal beliefs, with the sum-product algorithm providing a unifying view on factor graphs (Pearl 1988; Kschischang, Frey, and Loeliger 2001). Loopy and generalized variants connect BP fixed points to variational objectives (e.g., Bethe-style free energies), providing a principled lens on approximate propagation in graphs with complex dependencies (Yedidia, Freeman, and Weiss 2003; Wainwright and Jordan 2008).

Trust propagation has been widely studied in social network analysis (Guha et al. 2004) and citation networks (Brin and Page 1998; Kleinberg 1999). These approaches model trust as flowing through edges weighted by structural properties (citation count, user ratings). However, they lack grounding in content-based verification of the actual claims being made, which we incorporate to improve automated claim validation.

Recent work on detecting misinformation (Mridha et al. 2021) has incorporated source credibility and cross-source consistency, but these methods aggregate signals at the document level rather than modeling fine-grained dependency chains within the argumentation of a single paper. Our trust-gating mechanism differs by explicitly modeling how parent node quality gates the confidence of child nodes, preventing weak evidence from inflating downstream claims through a differentiable threshold function.

Multi-Agent Systems

Autonomous LLM-based agents with web-browsing capabilities have emerged as a powerful paradigm for information gathering (Nakano et al. 2021; Browser-Use 2024). Recent systems combine large language models with browser automation to answer questions, compare products, and fact-check claims (Mozannar et al. 2025a). Our work applies this approach to scientific verification by coordinating multiple agents to extract verification metrics across six dimensions (credibility, relevance, evidence strength, method rigor, reproducibility, and citation support).

Positioning this Work in the Broader Literature

We uniquely combine knowledge graph construction with content-based multi-agent verification and trust propagation in a learnable framework, aiming to expand fact checking from sentence-level judgments to complete argumentative structures. Unlike citation-network methods, our signals are grounded in verified content rather than popularity alone. This paper should therefore be read as a study on the proposed system and calibration schemes rather than as a definitive benchmark comparison against external baselines.

3 Problem Formulation

We formalize the scoring problem as follows. Given a candidate knowledge graph extracted from a paper (nodes with roles and text, and directed dependency edges) together with per-node verification metrics, our goal is to compute calibrated node qualities, edge confidences, and an overall graph score. Formally, we represent the paper as a knowledge graph $\mathcal{G} = (V, E)$ where:

- Each node $v \in V$ represents a semantic unit with role $\rho_v \in \mathcal{R}$ and quality score $q_v \in [0, 1]$.
- Each edge $(u, v) \in E$ represents a logical dependency with confidence $C_{u \to v} \in [0, 1]$.
- The graph $\mathcal{G}$ is a directed acyclic graph (DAG) respecting role-specific constraints.

The semantic role set is defined as $\mathcal{R} := \{$ Assumption, Claim, Conclusion, Context, Counterevidence, Evidence, Hypothesis, Limitation, Method, Result$\}$ following the scorer's canonical role ontology. From $\mathcal{G}$, we compute an overall graph score $S_{\text{graph}} \in [0, 1]$ that aggregates six interpretable components $\mathcal{T} := \{$ bridge coverage, best-path reliability, redundancy, fragility, coherence, and coverage $\}$. Our objective is for S_{graph} to correlate with coarse human triage judgments while remaining decomposable into auditable sub-scores.

The coefficients $\{\gamma_t\}_{t \in \mathcal{T}}$ are calibration weights. Positive values reward desirable structural properties, while negative values encode penalties (e.g. fragility). The raw weighted sum is normalized back to $[0, 1]$ after aggregation.

$$S_{\text{graph}} = \phi_{\text{clip}[0,1]} \left(\sum_{t \in \mathcal{T}} \gamma_t \, S_t \right). \tag{1}$$

The key challenges are addressed with the following **four-step** implementation. First, via **(1) Structure Extraction**, we parse unstructured text into a structured DAG with correct semantic roles. Then, through **(2) Claim verification**, we assess the validity of individual nodes using external knowledge. This is followed by **(3) Trust propagation** where node qualities flow through edges via a BP-inspired message-passing scheme such that weak evidence does not inflate downstream claims and finally **(4) Interpretability** ensures scores are explainable and aligned with human reasoning (Pearl 1988; Kschischang, Frey, and Loeliger 2001; Yedidia, Freeman, and Weiss 2003).

4 Implementation

This section describes the reference implementation of Plato's Cave, as seen in Figure 1. This is intentionally *factorized* into many sections, including structure extraction, node verification, and graph scoring. These separate modules are connected by simple JSON artifacts, enabling independent iteration and ablation.

Knowledge Graph Construction

We use a fixed ontology of 10 roles: Hypothesis, Claim, Evidence, Method, Result, Context, Assumption, Counterevidence, Limitation and Conclusion. Each node is a short excerpt of the paper (typically 1-3 sentences) labeled with one of these roles. Edges represent *dependency flow* from higher-level toward downstream synthesis; in the default priors, Hypothesis nodes are treated as roots and Conclusion nodes as sinks, with intermediate support roles (i.e. Evidence/Method-/Result/Claim) acting as hypothesis to conclusion bridges.

A DAG extractor prompts an LLM with the paper text, the ontology, and a strict JSON schema (nodes with `id`, `role`, `text`, `parents`, `children`). The output is validated for: (i) ontology compliance, (i.e. the role of each node must be one of the 10 allowed values), (ii) referential integrity (i.e. parent/child IDs must exist), and (iii) acyclicity. If validation fails, the system attempts a bounded "repair" pass by returning the validation errors to the LLM and requesting a corrected JSON. In Figure 2, the DAG extraction uses `gpt-5-mini` and enforces a maximum of 16 nodes per sampled DAG.

Node Verification

Node verification is performed by an ensemble of LLM verifiers, where this agentic system is made available in a corresponding code release to aid replication. In this released scorer, verifier agents designate the presence or absence of six metrics per node (i.e. credibility, relevance, evidence strength, method rigor, reproducibility, citation support), each in $[0, 1]$. Each verification agent is powered by browser-use, essentially an LLM with access to web search, navigation, and content extraction tools. The system also relies on Exa, an AI-powered search engine, as a fallback, in the event the web-surfer is unreachable. Agents receive a node (including the role and associated text snippet) and execute a search-verify-assess workflow:

1. **Search**: Formulate search queries based on the node text and role (e.g., for Evidence nodes, search for the cited source; for Method nodes, search for validation studies).
2. **Navigate**: Click search results, scroll to relevant sections, extract text, take screenshots when relevant.
3. **Assess**: Evaluate source credibility (domain authority, publication venue), relevance (node claim alignment), and supporting evidence (data, citations).

We use the browser-use library (Browser-Use 2024), which optimizes LLM-browser interaction with action batching and caching, achieving 3-5x speedup over naive implementations. Agents run in Docker containers with remote browsers accessed via Chrome DevTools Protocol, enabling real-time visual monitoring through noVNC and interaction in the browser being used by the agent. For each node v, verification agents extract six metrics $\mathbf{m}_v = (m_{v,1}, \ldots, m_{v,6}) \in [0, 1]^6$: ($m_1$) **Credibility**, the trustworthiness of sources where peer-reviewed journals score higher than blogs); (m_2) **Relevance**, the alignment between node claim and retrieved evidence; (m_3) **Evidence strength**: the quality and quantity of supporting data; (m_4) **Method Rigor**, the soundness of experimental design for method nodes; (m_5) **Reproducibility**, the availability of code, data, or detailed procedures and (m_6) **Citation support**, the number and quality of citations backing the claim. The agent extracts these metrics by prompting the LLM with retrieved content and a structured output schema. Metrics are normalized and stored alongside the node.

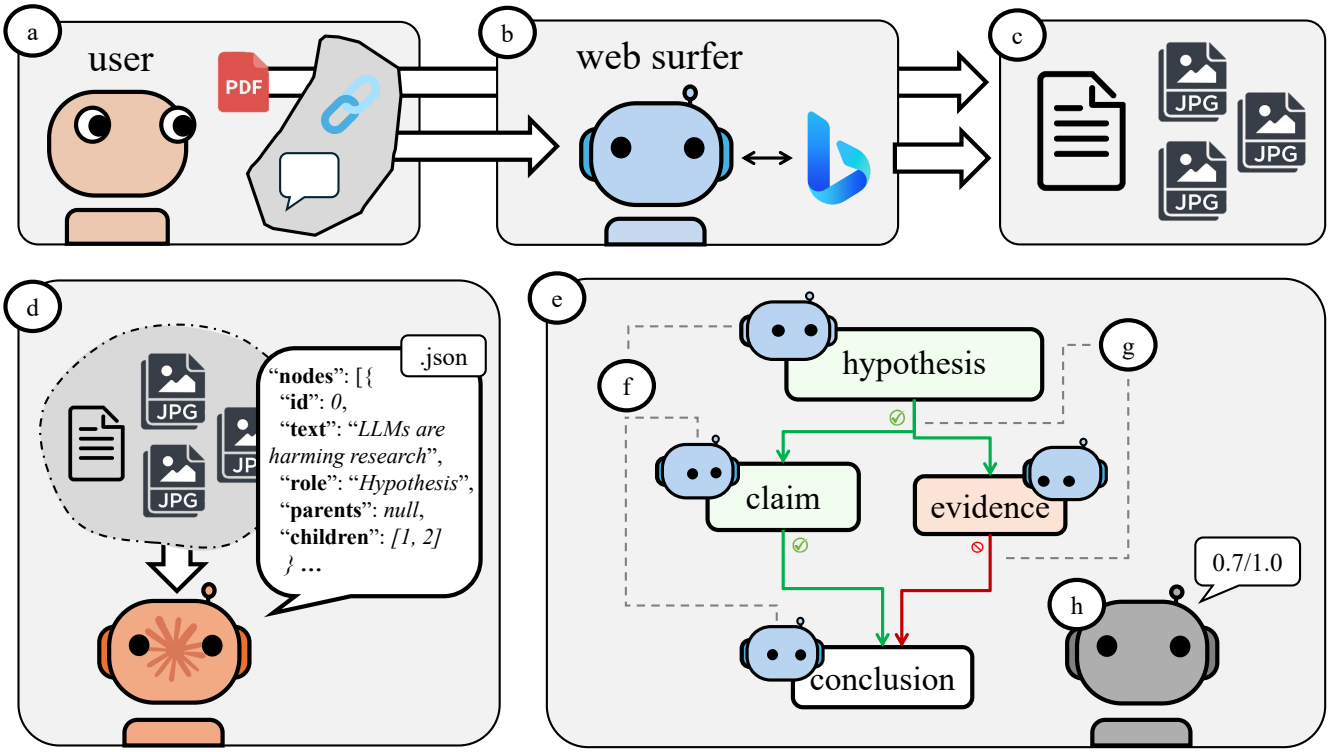

Figure 1: System Overview. **(a)** The user provides a PDF, URL, or natural-language query. **(b)** For URLs or queries, a web-surfer agent browses for the specific paper. **(c)** All text and images from the document are extracted and stored. **(d)** The content is passed into an LLM and converted into a role-labeled DAG serialized as JSON (nodes and directed dependencies). **(e)** The frontend parses the DAG JSON and renders an interactive graph. **(f)** Web-surfer agents verify each node sequentially using external sources to produce normalized verification metrics. **(g)** Node qualities influence downstream nodes via trust-gated propagation along dependency edges. **(h)** The scorer aggregates node and edge signals into an overall paper-level score.

Trust-Propagating Graph Scoring

This subsection summarizes the *real-time* scoring backend used to evaluate a candidate paper-DAG after node-level metrics are available. The goal is to assign (i) a **quality** to each node, (ii) a **confidence** to each dependency edge, and (iii) a **paper-level score** that reflects whether at least one hypothesis is supported by a coherent chain of evidence, method, and results.

For **inputs and update protocol** the scorer consumes a validated DAG $\mathcal{G} = (V, E)$ where each node $v \in V$ has: (i) role $\rho_v \in \mathcal{R}$, (ii) node text t_v, and (iii) metrics $\mathbf{m}_v \in [0,1]^6$. Node qualities and edge confidences are recomputed after metrics are available for a node and its parents, ensuring stability of intermediate results.

For the **node quality (role-aware metric fusion)** each node's metrics are optionally reweighted by a *global* per-metric weight vector, then fused into a single node quality score $q_v \in [0,1]$ using *role-specific* weights. Weights are normalized internally (by ℓ_1 magnitude) and the resulting score is clipped to $[0,1]$.

For **edge confidence**, each directed edge $(u \to v)$ receives two related scores:

- **Raw edge confidence** $C^{\text{raw}}_{u\to v}$: a local plausibility score computed as a weighted mixture of: (i) a role-transition prior for (ρ_u, ρ_v), (ii) parent and child node qualities (q_u, q_v), (iii) a lightweight lexical alignment score between node texts (Jaccard overlap), and (iv) a role-pair *synergy* term that mixes parent and child metrics in a role-specific way.
- **Final edge confidence** $C_{u\to v}$: a trust-gated version of $C^{\text{raw}}_{u\to v}$ that down-weights dependencies coming from untrustworthy upstream nodes.

Following that, **trust propagation** assigns each node a propagated trust value $t_v \in [0,1]$ that combines its own quality q_v with the "weakest-link" character of upstream support. Each parent contributes an amount proportional to its trust (raised to an exponent α) multiplied by the *raw* edge confidence into v. Parent contributions are aggregated using one of four modes: `min`, `mean`, `softmin`, or `dampmin`. Finally, exposed edge confidence is gated by parent trust with a floor parameter η. Viewed through the lens of graphical-model inference, this is a deterministic message-passing rule analogous in spirit to (loopy) belief propagation / sum-product, but operating on bounded trust scores and incorporating an explicit trust gate to prevent unreliable parents from boosting descendants. (Kschischang, Frey, and Loeliger 2001; Yedidia, Freeman, and Weiss 2003; Wainwright and Jordan 2008)

The **graph paper-level score** aggregates six interpretable

Figure 2: Plato's Cave interface showing a finalized run with the visualized DAG and Integrity Score

components computed on the *bridge* subgraph that connects at least one **Hypothesis** to at least one **Conclusion**: bridge coverage, best-path reliability, redundancy (max-flow), fragility (min-cut, penalized), coherence, and coverage of key roles. We treat best-path reliability as the primary *argument strength* signal by selecting a single hypothesis to conclusion chain maximizing product confidence and reporting its length-normalized (geometric-mean) confidence.

The **learnable parameter surface** implementation exposes an interpretable parameter surface for calibration and ablations: global metric weights, role-specific node-quality weights, edge-combination weights, role transition priors, role-pair synergy specifications, propagation hyperparameters, and graph-score weights. We do not claim these defaults are uniquely correct a priori; instead, they define an auditable basis set whose contribution can be isolated through cache-first ablations over fixed DAG and node-score artifacts.

The current release does not implement gradient-based learning from human labels, as it focuses on transparent decomposition, feature export, and reproducible calibration.

Cache-First Calibration and Ablation

To calibrate the exposed parameter surface we use offline studies that reuse cached factorized runs rather than issuing new LLM calls. This separates the question of which signals the pipeline extracts from the question of how those signals should be aggregated.

1. **Dense exploration**: We first search a broad parameter region over metric weights, edge-combination weights, propagation hyperparameters, and graph-level aggregation weights.
2. **Local refinement**: We then narrow the search around the best-performing dense configurations to improve ranking discrimination while preserving interpretability.
3. **Sparse fine-tuning**: Finally, we apply sparse local perturbations around the best refined configuration to test whether small coordinate-wise changes still improve the tuned objective.

In parallel, the release includes cache-first ablations over node metrics, node roles, edge features, propagation settings, and graph-head components. These experiments are intended to test sensitivity and reproducibility of the aggregation rule, not to claim that the present ontology or weight choices are uniquely optimal.

Architecture and Implementation

We implement Plato's Cave as an open-source full-stack web application with modular components[1].

The frontend is built with Gatsby.js and React, providing an interactive graph visualization using ReactFlow with Dagre layout for hierarchical DAG rendering. Users upload PDFs, provide URLs or prompt the system with natural language, monitor real-time verification progress via WebSocket connections, and explore the resulting knowledge graph with hover tooltips displaying the six verification metrics per node and confidence breakdowns per edge. The interface highlights the best hypothesis-to-conclusion path and allows filtering by node role or quality threshold. Figure 3 depicts the system architecture.

The release provides a self-contained scoring in Python:

- `kg_realtime_scoring.py`: validation of KG JSON into a DAG, real-time updates to node qualities, trust

[1]Available at https://github.com/matheusmaldaner/PlatosCave

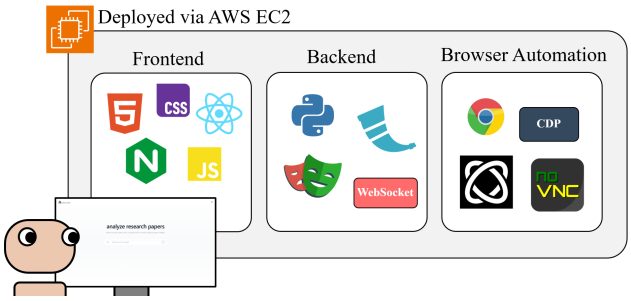

Figure 3: System Architecture for Plato's Cave

values, edge confidences, and graph-level scoring; plus export utilities (node/edge feature matrices, random-walk corpus, and a paper fingerprint vector).

- `service_adapter.py`: a thin session wrapper (`KGSession`) that (i) computes a BFS node order from Hypothesis roots, (ii) accepts metric updates for the current node, and (iii) returns updated edges and scores for UI/agent loops.

Browser Automation

We deploy browser-use agents in Docker containers with remote browsers accessed via Chrome DevTools Protocol. This architecture provides isolation (each agent runs in its own container), visual monitoring (noVNC allows observing agent actions in real-time), and scalability (horizontal scaling by launching more containers). The browser-use library optimizes LLM-browser interaction through action batching, DOM caching, and smart scrolling, improving efficiency over naive selenium-based approaches.

Analysis Modes

Scientific and technical documents differ substantially in structure, evidentiary norms, and verification requirements. A clinical trial, an arXiv preprint, and an SEC 10-k expose different failure modes and demand different extraction priorities. To address this heterogeneity without fragmenting the core pipeline, we introduce *analysis modes*: lightweight, domain-specific configurations that modify claim extraction and graph construction while preserving a shared verification and scoring backend. The system additionally supports three mutually exclusive modes, selectable via a CLI flag (`--mode`) or frontend user interface:

- **Academic**: preprints, theses, conference papers; emphasizes argument structure and theoretical grounding.
- **Journal**: peer-reviewed articles and clinical studies; emphasizes statistical rigor, replication, and disclosure.
- **Finance**: earnings reports, SEC filings, investor materials; emphasizes financial metrics and risk disclosures.

Modes act exclusively at DAG extraction time by injecting domain-specific instructions into the LLM prompt. All modes share the same base ontology of 10 universal roles (i.e. Hypothesis, Claim, Evidence, Method, Result, Conclusion, Assumption, Counterevidence, Limitation, Context).

The extractor prompt specifies the expanded role set and mode-specific prioritization rules (e.g., extracting exact p-values and confidence intervals in Journal mode, or separating historical data from forward-looking statements in Finance mode). DAG validation accepts the union of base and mode-specific roles, ensuring schema consistency across modes.

Importantly, modes do *not* alter downstream verification or scoring logic. Once a valid DAG is produced, node verification agents and trust-propagating graph scoring operate identically across modes. This design choice isolates domain assumptions to extraction only, enabling controlled ablations and preventing silent changes to scoring semantics.

Design Rationale

Several architectural decisions merit explanation. We chose a multi-agent architecture over a single sequential agent because verification tasks are independent and parallelization reduces total latency from minutes to seconds. We separated the graph scorer into its own service to enable offline experimentation with scoring parameters without re-running verification. We use WebSockets for real-time updates rather than polling to provide responsive user feedback during the lengthy verification process. We enforce DAG constraints at generation time rather than post-hoc to prevent invalid graphs from reaching verification, saving compute resources.

The modular design allows future extensions such as replacing the LLM backend, swapping verification agents (e.g., defining custom functions for tool calling), and integrating learned parameters (by exposing the scorer's gradient interface) with Figure 3 depicting our implementation.

5 Evaluation

We evaluate Plato's Cave as an *end-to-end verification pipeline* and as a *scoring system*. The attached results correspond to a pilot run (`runs/experiments_debug4`) intended to validate throughput, artifact quality, and failure modes prior to a larger human-annotated evaluation.

Experimental Setup

The run loaded 104 paper records from a spreadsheet collection spanning three topical sheets (Economics, ML/Computing, and Psychology). Each processed paper also has a spreadsheet triage label in $\{\texttt{Bad}, \texttt{Neutral}, \texttt{Good}\}$ assigned during collection curation. We use these labels only as weak supervision for calibration: they summarize an overall document-level judgment of the paper's central claims and practical credibility, rather than serving as gold labels for every extracted node or edge. Accordingly, we treat this evaluation as a noisy ranking signal rather than a definitive quality benchmark. For each processed paper we sample $K = 8$ candidate DAGs and, for each *valid* DAG, run $M = 8$ scoring trials by resampling node-quality/propagation weights. This produced 810 valid DAG samples and 6480 total trials. DAG extraction uses `gpt-5-mini`; node verification uses `gpt-5-nano`. The run uses bounded concurrency at three levels (papers, nodes, and global LLM calls) to avoid rate-limit collapse while maintaining parallelism.

Regarding **Throughput**, the full collection run completed in 13.99 hours wall-clock time. Among successfully processed papers, the median per-paper runtime was 65.4 minutes (90th percentile **70.7** minutes).

DAG Validity and Repair

Across 832 sampled DAGs, 22 failed validation (2.6%). The most common failure modes were (i) *unknown role labels* (e.g., "Recommendation", "Example") and (ii) *dangling references* where a parent/child ID did not exist. These failures are actionable: tightening the extraction prompt to forbid out-of-ontology role names and to require explicit, consistent IDs reduces wasted trials and improves effective throughput.

Ranking Performance and Calibration

We evaluate each paper's mean score (averaged over all valid DAG samples and node-weight trials) against the spreadsheet's coarse `Bad` / `Neutral` / `Good` label. Because these labels are weak supervision, we treat this as a calibration problem rather than a definitive benchmark and our experiments emphasize cache-first staged search and internal ablations over fixed cached artifacts. We additionally do not provide any external baseline comparison.

The calibration study uses three consecutive search stages over the cached `runs/experiments_debug4` artifacts: dense, refine, and sparse. For binary Good-vs-Bad discrimination, we report AUROC (area under the receiver operating characteristic curve), where 0.5 indicates chance-level separation and 1.0 indicates perfect separation. On the 104 papers whose labels are exactly `Good`, `Neutral`, or `Bad`, the best configurations achieve:

Search Stage	Best AUROC	Best Spearman
Dense search	0.7025	0.313
Refine search	0.7594	0.395
Sparse stage-3	0.7658	0.401

Performance improves monotonically across the tuned objective, the final sparse stage yields a narrow gain over the refine stage, showing signs of overfitting on this benchmark. For that reason, we treat the refine-stage configuration as the main focus and the sparse stage as evidence that the scoring surface requires further optimization.

Interpretation

Relative to the earlier untuned scorer, offline calibration produces moderate Good-vs-Bad separation and moderate ordinal agreement with the coarse triage labels. At the same time, the weakness of the supervision signal and the small stage-3 gain argue against stronger claims: the current evidence supports Plato's Cave as a tunable, auditable ranking scaffold, not yet as a definitive paper-quality estimator.

From the performance log, the median per-DAG extraction time was **43.1** s and the median per "score-nodes-once" trial time was **53.6** s (measured on the subset of papers captured in the perf log). These latencies imply that DAG extraction dominates per-paper cost under the $K \times M$ sampling regime.

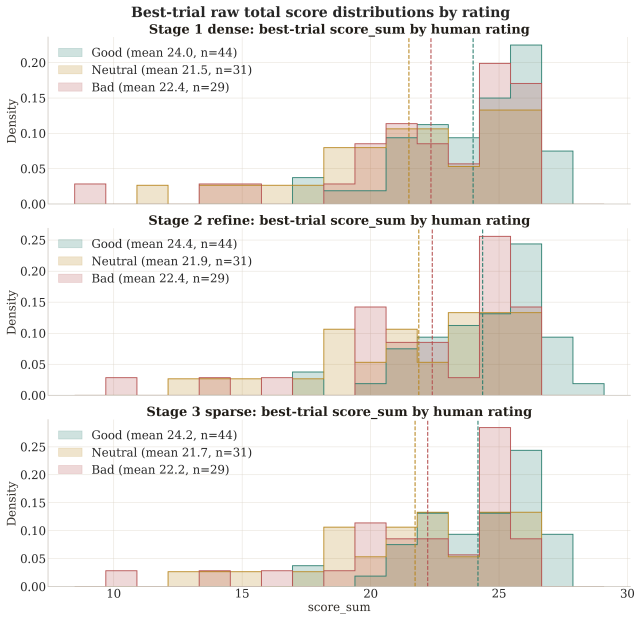

Figure 4: Paper-level mean score grouped by spreadsheet rating under the selected calibrated setting. We use a discrete, non-smoothed summary to avoid overstating distributional structure under weak supervision.

6 Discussion and Future Work

While the current system demonstrates that trust-propagating graph scoring over LLM-extracted DAGs can produce moderate alignment with human triage labels, there are many directions to be explored. This section outlines the strengths, limitations and future areas of work.

Strengths

Our trust-propagating knowledge graph approach offers several advantages over existing methods. The system provides interpretability by exposing the complete reasoning chain from evidence to conclusions, allowing users to inspect which nodes are weak and why. The trust-gating mechanism addresses a fundamental limitation of citation networks and knowledge graphs: it prevents weak evidence from inflating strong claims by modeling parent quality as a gate function. The multi-agent verification architecture grounds assessments in external knowledge rather than relying solely on parametric knowledge in language models. The differentiable scoring framework enables learning from human feedback, allowing continuous improvement with more annotations.

The attached release includes staged tuning over fixed artifacts, which make the scorer reproducible and inspectable. However, these studies should be interpreted as calibration evidence for the aggregation rule, not as proof that the current ontology, equations, or default weights are uniquely correct.

Limitations

Several limitations warrant discussion. The system requires substantial computational resources, particularly for multi-agent verification which involves multiple LLM API calls

and browser automation sessions per paper. In the full collection run, the median end-to-end per-paper runtime was 65.4 **minutes** and costs approximately $\$0.15$ in API fees. This limits scalability to real-time monitoring of preprint servers.

Our evaluation dataset contains **104** papers, which is modest compared to large-scale fact-checking benchmarks. We focus on depth (fine-grained annotations, multiple metrics) over breadth, but larger-scale evaluation would strengthen our claims. Future work should explore multi-annotator aggregation methods.

The semantic role ontology with 10 roles captures many aspects of scientific argumentation but may require domain-specific extensions. For example, biology papers might need roles for "Organism Model" or "Experimental Control," while computer science papers might need "Proof Sketch" nodes. We currently use a fixed ontology across domains, which may limit performance on specialized fields.

The trust-gate and aggregation parameters are currently selected by offline cache-first search rather than learned from human annotations. This is an improvement over purely hand-set defaults, but it still does not establish that the present weights will generalize beyond this benchmark or beyond the current weak supervision signal.

The system is human-centered because the reviewer and verification agent share the same live browser session. Reviewers can see which pages are being visited in real time and intervene when needed (for example, solving CAPTCHAs or entering university affiliation credentials to access paywalled sources), then allow the agent to continue verification with that access.

Future Work

We identify several promising research directions. Learning the trust gate parameters and role-specific weights from human annotations could improve performance and domain adaptation. Active learning could identify which nodes to verify next, reducing verification costs while maintaining accuracy. Cross-domain generalization experiments testing whether a model trained on computer science papers transfers to biology would assess the robustness of our semantic ontology. Integration with peer review systems could provide real-time feedback to reviewers and authors during the submission process.

Longitudinal monitoring of preprint servers like arXiv could flag potentially problematic papers early, before formal peer review; recent work on agents that can wait, monitor, and act over extended periods (Mozannar et al. 2025b) suggests that such sustained surveillance is becoming practical. Extending the system to handle multi-paper claims (e.g., a claim supported by evidence from multiple papers) would enable analysis of research programs rather than individual papers. Formalizing the types of explanations our system produces (trust scores, graph structure, verification chains) using general XAI syntax (Wormald et al. 2025) could standardize the interface and guide the design of richer, neurosymbolic explanation strategies. Explanations in natural language would further make the system more accessible to non-expert users.

Finally, exploring alternative verification methods beyond web search (e.g., running code to reproduce results, querying structured databases like PubMed) could improve accuracy for claims amenable to automated checking.

7 Conclusion

We presented Plato's Cave, a human-centered system for scientific paper verification that combines role-aware knowledge graphs, external node verification, and trust-gated score propagation. This produces interpretable DAGs, node/edge metrics, and auditable score traces, for reviewer inspection.

On a heterogeneous collection of 104 papers the factorized pipeline was operationally stable: only **2.6%** of DAG samples failed strict validation, and **97.4%** of attempted scoring trials completed successfully. Cache-first calibration then achieved Good-vs-Bad AUROC **0.759** at the conservative refine stage (**0.766** in the final sparse stage) with Spearman **0.395/0.401** on the 104 papers with exact `Good`/`Neutral`/`Bad` labels.

Overall, the current evidence supports Plato's Cave as a reproducible, auditable decision-support scaffold rather than a definitive paper-quality estimator. The main next steps are stronger supervision, external baselines, and explanation interfaces tied to specific low-trust nodes and edges.

References

Al Khatib, K.; Ghosal, T.; Hou, Y.; de Waard, A.; and Freitag, D. 2021. Argument Mining for Scholarly Document Processing: Taking Stock and Looking Ahead. In *Proceedings of the Second Workshop on Scholarly Document Processing*, 56–65. Online: Association for Computational Linguistics.

Brin, S.; and Page, L. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks*, 30: 107–117.

Browser-Use. 2024. https://browser-use.com/. Accessed: 2026-01-03.

Cheng, M.; Shah, S. M.; Nanni, A.; and Gao, H. O. 2024. Automated knowledge graphs for complex systems (AutoGraCS): Applications to management of bridge networks. *Resilient Cities and Structures*, 3(4): 95–106.

Guha, R.; Kumar, R.; Raghavan, P.; and Tomkins, A. 2004. Propagation of trust and distrust. In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, 403–412. New York, NY, USA: Association for Computing Machinery. ISBN 158113844X.

Kinney, R. M.; Anastasiades, C.; Authur, R.; Beltagy, I.; Bragg, J.; Buraczynski, A.; Cachola, I.; Candra, S.; Chandrasekhar, Y.; Cohan, A.; Crawford, M.; Downey, D.; Dunkelberger, J.; Etzioni, O.; Evans, R.; Feldman, S.; Gorney, J.; Graham, D. W.; Hu, F.; Huff, R.; King, D.; Kohlmeier, S.; Kuehl, B.; Langan, M.; Lin, D.; Liu, H.; Lo, K.; Lochner, J.; MacMillan, K.; Murray, T. C.; Newell, C.; Rao, S.; Rohatgi, S.; Sayre, P.; Shen, S. Z.; Singh, A.; Soldaini, L.; Subramanian, S.; Tanaka, A.; Wade, A. D.; Wagner, L. M.; Wang, L. L.; Wilhelm, C.; Wu, C.; Yang, J.; Zamarron, A.; van Zuylen, M.; and Weld, D. S. 2023. The Semantic Scholar Open Data Platform. *ArXiv*, abs/2301.10140.

Kirschner, C.; Eckle-Kohler, J.; and Gurevych, I. 2015. Linking the Thoughts: Analysis of Argumentation Structures in Scientific Publications. In *Proceedings of the 2nd Workshop*

on Argumentation Mining, 1–11. Denver, CO: Association for Computational Linguistics.

Kleinberg, J. M. 1999. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5): 604–632.

Kschischang, F. R.; Frey, B. J.; and Loeliger, H. 2001. Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*, 47(2): 498–519.

Lauscher, A.; Glavaš, G.; and Ponzetto, S. P. 2018. An Argument-Annotated Corpus of Scientific Publications. In *Proceedings of the 5th Workshop on Argument Mining*, 40–46. Brussels, Belgium: Association for Computational Linguistics.

Lawrence, J.; and Reed, C. 2019. Argument Mining: A Survey. *Computational Linguistics*, 45(4): 765–818.

Liu, K. 2020. A survey on neural relation extraction. *Science China Technological Sciences*, 63(10): 1971–1989.

Liu, R.; and Shah, N. B. 2023. ReviewerGPT? An Exploratory Study on Using Large Language Models for Paper Reviewing. *arXiv preprint arXiv:2306.00622*.

Liu, X.; Datta, A.; and Lim, E.-P. 2015. *Judging the Veracity of Claims and Reliability of Sources*, 40–71. CRC Press.

Mozannar, H.; Bansal, G.; Tan, C.; Fourney, A.; Dibia, V.; Chen, J.; Gerrits, J.; Payne, T.; Maldaner, M. K.; Grunde-McLaughlin, M.; Zhu, E.; Bassman, G.; Alber, J.; Chang, P.; Loynd, R.; Niedtner, F.; Kamar, E.; Murad, M.; Hosn, R.; and Amershi, S. 2025a. Magentic-UI: Towards Human-in-the-loop Agentic Systems. arXiv:2507.22358.

Mozannar, H.; Maldaner, M. K.; Murad, M.; Chen, J.; Bansal, G.; Hosn, R.; and Fourney, A. 2025b. Tell Me When: Building Agents That Can Wait, Monitor, and Act. https://www.microsoft.com/en-us/research/blog/tell-me-when-building-agents-that-can-wait-monitor-and-act/. Microsoft Research Blog. Accessed: 2026-03-01.

Mridha, M. F.; Keya, A. J.; Hamid, M. A.; Monowar, M. M.; and Rahman, M. S. 2021. A Comprehensive Review on Fake News Detection With Deep Learning. *IEEE Access*, 9: 156151–156170.

Nakano, R.; Hilton, J.; Balaji, S.; Wu, J.; Ouyang, L.; Kim, C.; Hesse, C.; Jain, S.; Kosaraju, V.; Saunders, W.; Jiang, X.; Cobbe, K.; Eloundou, T.; Krueger, G.; Button, K.; Knight, M.; Chess, B.; and Schulman, J. 2021. WebGPT: Browser-assisted question-answering with human feedback. *CoRR*, abs/2112.09332.

National Science Board. 2023. Publication Output by Region, Country, or Economy and by Scientific Field. Nsb-2023-33, National Science Foundation, Alexandria, VA.

NeurIPS 2025 Program Chairs. 2025. Reflections on the 2025 Review Process from the Program Committee Chairs. https://blog.neurips.cc/2025/09/30/reflections-on-the-2025-review-process-from-the-program-committee-chairs/. Accessed: 2026-03-01.

Paper Copilot. 2026. Conference Statistics: ICML, ICLR, NeurIPS. https://papercopilot.com/statistics/. Accessed: 2026-03-01.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA, USA: Morgan Kaufmann Publishers. ISBN 1558604790.

Retraction Watch. 2024. Retraction Watch Database. https://retractionwatch.com/2024/. Accessed: 2026-01-01.

Retraction Watch. 2025. Cheers, 2025: Retraction Watch Turned 15, Center for Scientific Integrity. https://retractionwatch.com/2025/12/30/cheers-2025-retraction-watch-turned-15-center-for-scientific-integrity/. Accessed: 2026-01-01.

Sinha, A.; Shen, Z.; Song, Y.; Ma, H.; Eide, D.; and Wang, K. 2015. An Overview of Microsoft Academic Service (MAS) and Applications. In *WWW - World Wide Web Consortium (W3C)*.

Strathern, M. 1997. 'Improving Ratings': Audit in the British University System. *European Review*, 5(3): 305–321.

Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; and Su, Z. 2008. ArnetMiner: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, 990–998. New York, NY, USA: Association for Computing Machinery. ISBN 9781605581934.

Thorne, J.; Vlachos, A.; Christodoulopoulos, C.; and Mittal, A. 2018. FEVER: a large-scale dataset for Fact Extraction and VERification. arXiv:1803.05355.

Wadden, D.; Lin, S.; Lo, K.; Wang, L. L.; van Zuylen, M.; Cohan, A.; and Hajishirzi, H. 2020. Fact or Fiction: Verifying Scientific Claims. arXiv:2004.14974.

Wainwright, M. J.; and Jordan, M. I. 2008. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning*, 1(1-2): 1–305.

Wormald, S.; Maldaner, M. K.; O'Connor, K. D.; Dizon-Paradis, O. P.; and Woodard, D. L. 2025. Abstracting general syntax for XAI after decomposing explanation subcomponents. *Artificial Intelligence Review*.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yedidia, J. S.; Freeman, W. T.; and Weiss, Y. 2003. Understanding Belief Propagation and its Generalizations. In Lakemeyer, G.; and Nebel, B., eds., *Exploring Artificial Intelligence in the New Millennium*, 239–269. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Zhang, F.; Liu, X.; Tang, J.; Dong, Y.; Yao, P.; Zhang, J.; Gu, X.; Wang, Y.; Shao, B.; Li, R.; and Wang, K. 2019. OAG: Toward Linking Large-scale Heterogeneous Entity Graphs. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, 2585–2595. New York, NY, USA: Association for Computing Machinery. ISBN 9781450362016.

Zhu, Y.; Wang, X.; Chen, J.; Qiao, S.; Ou, Y.; Yao, Y.; Deng, S.; Chen, H.; and Zhang, N. 2024. LLMs for Knowledge Graph Construction and Reasoning: Recent Capabilities and Future Opportunities. arXiv:2305.13168.