PAPERSYNC: RECONSTRUCTING RESEARCH EVOLUTION FOR EVIDENCE-GATED INCREMENTAL MANUSCRIPT SYNCHRONIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Research evolution makes manuscript maintenance difficult: new experimental results may change empirical claims, code changes may require revised method descriptions, and new literature may reshape comparisons. However, many changes warrant no textual edit. Determining what to revise requires tracing changes across research artifacts and establishing what the available evidence supports. Thus, we propose **PaperSync**, a framework for tracing research evolution and selectively synchronizing manuscript claims. PaperSync constructs a versioned research graph to connect research changes with manuscript claims, traces potentially affected content, and uses evidence contracts to determine whether to edit, preserve, or defer revision. Supported edits are realized as localized patches with traceable provenance. We further introduce ManuSyncBench, a repository-grounded benchmark comprising 203 cases from 128 papers and 331 atomic synchronization targets across exact-consistency and semantic-incremental synchronization. Experimental results show that PaperSync improves exact-span recall over direct LLM editing across all four reported backbones and outperforms the general-purpose agentic baseline on this metric for three, reaching 4.26% versus 3.02% with GPT-5.6-sol.

1 INTRODUCTION

Scientific manuscripts are increasingly coupled with research artifacts that continue to evolve after writing. Code, configurations, and experimental results change as projects develop, while newly published work may alter the comparisons and positioning surrounding a contribution (Pineau et al., 2021; Werner et al., 2024; Vranješ et al., 2024). Existing systems address parts of this maintenance problem: paper–code consistency methods detect discrepancies between manuscripts and implementations (Baumgärtner & Gurevych, 2026; Liu et al., 2026), documentation systems propagate code changes into technical text (Birru et al., 2025), and reproducibility frameworks organize evolving research artifacts and experimental states (Aguilar et al., 2024; Costa et al., 2026). These approaches improve change detection and artifact inspection, but typically assume a fixed snapshot, a specified engineering change, or an externally supplied editing intent.

Research evolution, however, follows two related but distinct trajectories: evolution within a project and evolution of methods in its surrounding field. Our current corpus contains 203 project-evolution instances, each spanning 3–8 version steps reconstructed from historical changes in code, configurations, and experimental results. Field evolution is less amenable to closed-world annotation; we therefore approximate it through a standardized literature-retrieval procedure that identifies relevant post-publication methods and comparison targets for each project. These two forms of evolution can affect a manuscript differently: an implementation change may alter a method description without changing an empirical claim, while a newly published method may warrant revisiting a comparison even when the project itself remains unchanged.

A detected change therefore does not directly determine how a scientific manuscript should be revised. A newer implementation may reflect a different configuration rather than an error in the paper; a new experiment may affect only a subset of empirical claims; and newly available literature may change positioning without invalidating the project's own results. More generally, a research

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.

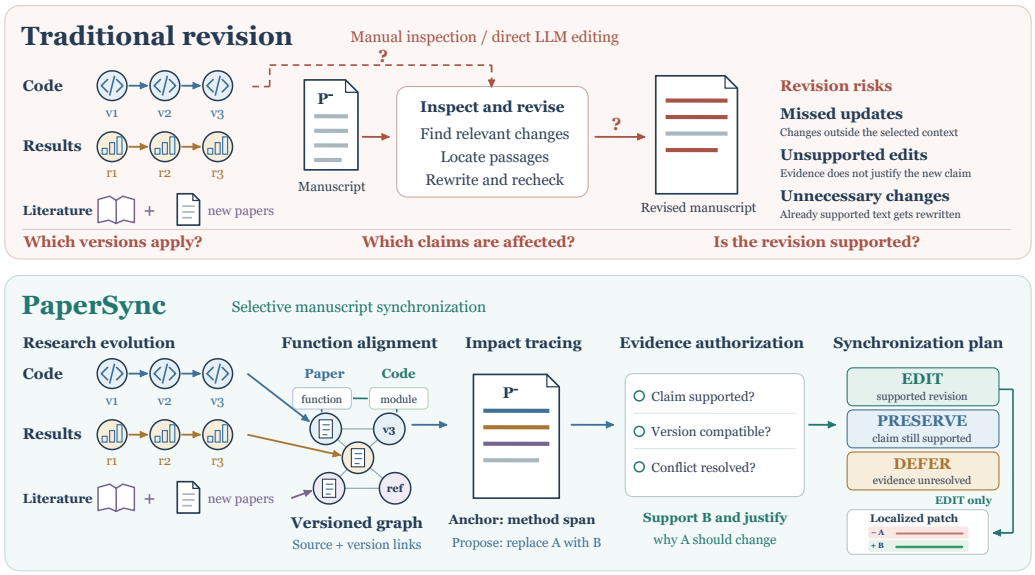

Figure 1: **Traditional manuscript revision versus PAPERSYNC.** Manual review and direct LLM editing must implicitly determine which changes matter, where they propagate, and whether a revision is supported. PAPERSYNC instead reconstructs research evolution, traces potentially affected claims, and separately authorizes revisions before producing localized patches.

change may require an edit, may leave existing text valid, or may remain unresolved because the available evidence is insufficient or conflicting. Manuscript synchronization is thus not simply a discrepancy-detection or rewriting problem: it requires deciding whether a change should propagate into the manuscript and what the available evidence permits the revision to say (Wadden et al., 2022; Zhang et al., 2023; Vladika & Matthes, 2024).

This setting introduces three challenges. First, research evolution must be reconstructed across heterogeneous artifacts and versions, since code, configurations, experiments, manuscript states, and subsequent literature may describe different research states. Second, changes must be localized to the scientific claims they may affect rather than propagated through temporal or structural proximity alone. Third, impact must be separated from revision authorization: identifying an affected claim provides a reason to inspect it, but does not establish that the claim is wrong or that a proposed replacement is supported. Conflating these stages can lead to missed updates, unsupported edits, or unnecessary rewriting of already valid content.

We address these challenges with PAPERSYNC, a framework for *selective, evidence-grounded manuscript synchronization* under research evolution (Figure 1). PAPERSYNC reconstructs project and field evolution in a versioned research graph, performs *impact tracing* to identify manuscript claims that merit review, and then applies *evidence authorization* to determine whether a candidate revision is supported under the relevant version and evidence scope. The resulting synchronization plan records explicit EDIT, PRESERVE, or DEFER decisions, and only authorized edits are realized as localized, reviewable patches.

To support systematic evaluation, we introduce *ManuSyncBench*, a repository-grounded benchmark covering exact-consistency and semantic-incremental manuscript synchronization. It evaluates whether systems can identify, localize, and correctly realize verifiable evidence-supported revisions while avoiding unintended changes to unrelated content.

Our contributions are threefold. First, we formulate manuscript maintenance under research evolution as a selective synchronization problem that separates change detection, impact localization, and revision authorization. Second, we introduce PAPERSYNC, which combines versioned research-evolution reconstruction, claim-centered impact tracing, evidence-constrained authorization, and

localized patching. Third, we introduce ManuSyncBench for evaluating verifiable manuscript synchronization under project evolution.

2 Related Work

From Static Alignment to Update Intent. Research-artifact frameworks support reproducibility. Experiments as Code (Aguilar et al., 2024) treats configuration, execution, and auditing as first-class entities, while SciRep (Costa et al., 2026) packages code, data, dependencies, and commands for reproducible computation. Neither assesses whether a paper remains consistent with its current artifacts. Paper–code consistency work instead detects discrepancies: SciCoQA (Baumgärtner & Gurevych, 2026) benchmarks real and synthetic paper–code discrepancies, BioCon (Xu et al., 2026) aligns method sentences with function fragments, and Dude (Liu et al., 2026) combines dual detection with granularity negotiation. For documentation systems, README maintenance (Gao et al., 2026) identifies PR-driven updates, CodeDocSync (Birru et al., 2025) explains code changes and updates retrieved documentation, and DocAgent (Yang et al., 2025) distributes understanding, retrieval, writing, and verification across agents. XtraGPT (Chen et al., 2025) revises academic text under human instructions. These settings assume a fixed snapshot, a single engineering change, or externally supplied editing intent. PaperSync instead infers revision intent from research evolution: changes to code, configurations, and experiments, together with new domain evidence, are assessed for their effects on scientific claims and localized to affected paper passages.

Cross-Scale Impact Localization. Existing graph-based work models dependencies at distinct scopes. EvoGraph (Costa & Baran, 2025) links code, documentation, build pipelines, and tickets for software-artifact co-evolution. ARA (Riehl et al., 2026) extracts a sources→methods→experiments→outputs workflow from a paper, while KnoVo (Rubaiat et al., 2025) represents related-work comparisons through explicit dimensions and a novelty graph. Revision-propagation work (Kikuta, 2026) locates follow-on edits in conversation-generated artifacts after a local revision. These methods model engineering dependencies, paper-internal research chains, literature relations, or conversation history, but not their joint implications for paper claims. PaperSync performs claim-centered impact analysis over project-internal and domain-external evolution, returning inspectable candidate paper locations rather than arbitrary graph-reachable nodes.

Evidence-Constrained Updating. Scientific claim-verification work makes claim–evidence relations explicit. SciVer (Wang et al., 2025) provides expert-annotated evidence for multimodal scientific claims; SciClaimEval (Ho et al., 2026) constructs supporting and refuting cases around figures and tables; and SciLens (Wang et al., 2026) decomposes complex claims into empirical atoms for entailment testing. CLAIMCHECK (Ou et al., 2025) verifies claim–critique correspondence in paper review, while BrainPilot (Li et al., 2026) audits traceable multi-agent research processes after generation. PaperSync instead uses evidence as a pre-write admission criterion: a candidate revision is admissible only when its supporting evidence is traceable to current project artifacts or domain literature, supports the target claim, and remains within the identified impact scope. Minimal editing is therefore a testable scope constraint: local research changes should preserve unrelated passages and remove obsolete statements.

3 Task Formulation

Manuscript synchronization under research evolution. We formulate manuscript synchronization as a selective revision problem under evolving research evidence. A research change does not necessarily imply that the manuscript should be edited: it may affect an existing claim, leave the claim valid under its original scope, or raise a question that cannot yet be resolved from the available evidence. The task is therefore to identify manuscript content potentially affected by research evolution and determine what revision, if any, is warranted. Given an editable manuscript P^-, an evidence cutoff τ, and an editing scope u, let

$$\mathcal{A}_{\leq\tau} = \mathcal{A}^{\mathrm{proj}}_{\leq\tau} \cup \mathcal{A}^{\mathrm{field}}_{\leq\tau},$$

where project sources $\mathcal{A}^{\mathrm{proj}}_{\leq\tau}$ include code, configurations, experiment records, version history, and prior manuscript versions (Wonsil et al., 2023; Randrianaina et al., 2024), and field sources $\mathcal{A}^{\mathrm{field}}_{\leq\tau}$

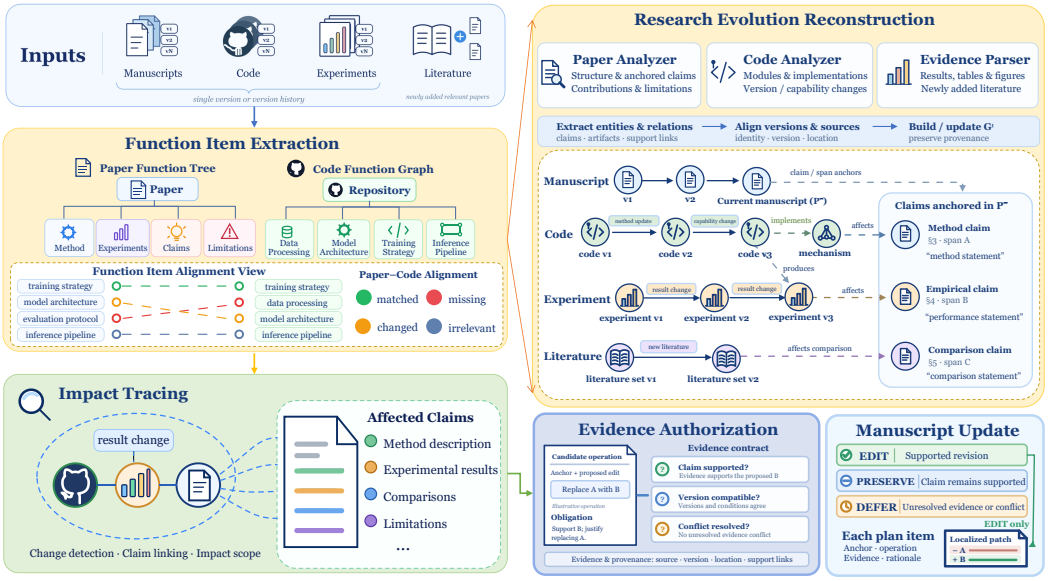

Figure 2: **Overview of PaperSync.** PaperSync reconstructs project evolution from versioned research artifacts and aligns functional units across the manuscript and implementation. Impact tracing identifies claims that merit review, while evidence authorization determines whether a proposed revision is supported under the applicable version and evidence scope. The resulting plan records EDIT, PRESERVE, or DEFER decisions before authorized edits are realized as localized patches.

include dated publications and findings relevant to the manuscript's methods and comparisons (Vladika & Matthes, 2024). The cutoff τ determines which evidence is available, while u specifies which manuscript content may be revised. Sources retain their provenance, location, applicable version, and experimental or methodological conditions, since artifacts available at similar times need not describe the same research state. Manuscript synchronization can then be written as

$$\Pi = F(P^{-}, \mathcal{A}_{\leq\tau}, u),$$

where Π is a revision plan rather than an unconditional set of manuscript edits.

Revision plan and evidence constraints. The output is a set of decisions over inspected manuscript spans or permissible insertion points (Birru et al., 2025):

$$\Pi = \{\pi_j\}_{j=1}^{m}, \qquad \pi_j = (a_j, z_j, p_j, E_j, r_j), \qquad z_j \in \{\mathsf{edit}, \mathsf{preserve}, \mathsf{defer}\}. \tag{1}$$

Here a_j is the manuscript anchor, z_j is the synchronization decision, p_j is a proposed patch when $z_j = \mathsf{edit}$, $E_j \subseteq \mathcal{A}_{\leq\tau}$ records the evidence examined, and r_j records the justification. An edit inserts, replaces, or deletes content and is authorized only when the evidence supports both the proposed scientific assertions and the revision operation under the relevant version and conditions. For example, code may establish that a component was implemented, whereas an empirical effect requires evidence from an appropriate experiment and configuration (Wonsil et al., 2023; Randrianaina et al., 2024); similarly, literature may support an attributed comparison without by itself establishing a novelty claim. A preserve decision records that the inspected content remains valid within scope u, whereas defer records that the available evidence is insufficient or conflicting and therefore supports neither an edit nor affirmative preservation (Wadden et al., 2022; Zhang et al., 2023). Successful synchronization should realize warranted revisions while leaving unrelated manuscript content unchanged. ManuSyncBench evaluates the verifiable-revision subset of this formulation, in which required updates and their supporting project evidence can be established reliably.

4 PAPERSYNC

PaperSync addresses two questions that an observed research change cannot answer on its own: which manuscript claims might it affect, and what revisions, if any, does the available evidence justify? We treat the first as *impact tracing* and the second as *evidence authorization*. This distinction is central: a change may provide a reason to inspect a claim without establishing that the claim is false or that a replacement statement is supported (Wadden et al., 2022; Zhang et al., 2023). As shown in Figure 2, PaperSync reconstructs research evolution, links changes to manuscript claims, and authorizes only evidence-supported revisions.

4.1 RECONSTRUCTING RESEARCH EVOLUTION

Reconstructing research evolution requires resolving differences in artifact granularity, preserving version-specific evidence, and handling incomplete histories. To address these challenges, PaperSync constructs a *versioned research evolution graph* that aligns research artifacts by function and records their temporal and evidential relationships. The construction proceeds in three steps.

First, PaperSync extracts manuscript claims and their text anchors, implementation units and configurations, and evidence from experiments and literature. Each record retains its source location, date, version, and applicable conditions.

Second, PaperSync aligns manuscript and implementation units by research function and integrates them into a graph with temporal, functional, and evidential relations. Experimental results remain linked to their evaluated configurations, preventing evidence from being generalized beyond its scope (Wonsil et al., 2023; Randrianaina et al., 2024).

Third, PaperSync reconstructs project changes from linked artifact versions and incorporates newly available literature relevant to comparisons, positioning, or limitations (Vladika & Matthes, 2024). Where history is unavailable, it records only the observed state. Inferred semantic relations are distinguished from observed provenance: they guide review but do not themselves authorize revisions (Wadden et al., 2022; Zhang et al., 2023).

4.2 IMPACT TRACING AS A REVIEW HYPOTHESIS

Given the reconstructed graph, PaperSync traces how recorded changes may bear on manuscript claims through claim-relevant relations. A changed implementation component may lead to a method description, a new experimental result to a performance claim under the corresponding configuration, and a newly introduced method to a comparison or positioning claim. Temporal succession or structural containment alone is insufficient: an impact path must explain why the changed object is relevant to the manuscript claim.

The output is a set of review targets, each containing a manuscript anchor or permissible insertion point, the triggering research change, and the relations connecting them. Impact tracing is intentionally recall-oriented: it surfaces claims that may require assessment but does not determine whether an edit should be made. An implementation change may justify reviewing a method description without affecting an empirical claim, while new literature may warrant revisiting a comparison without changing the project's own results (Birru et al., 2025). An impact relation is therefore a review hypothesis rather than an editing instruction.

4.3 EVIDENCE CONTRACTS FOR REVISION AUTHORIZATION

For each review target, PaperSync evaluates a concrete manuscript operation: insertion, replacement, deletion, or preservation. The candidate operation is paired with an *evidence contract* specifying the assertions it would add, modify, or withdraw; the sources examined; their applicable versions and conditions; and any material counterevidence. Authorization therefore concerns the proposed scientific content and its scope, rather than the mere presence of a retrieved source (Wadden et al., 2022; Zhang et al., 2023).

Different operations impose different evidential obligations. Insertions require support for added assertions; replacements additionally require evidence that the original statement should be superseded; deletions require evidence for withdrawing existing content; and preservation requires evi-

dence that the inspected statement remains applicable. Source roles further constrain authorization: code may establish implementation details, configuration-linked experiments may support empirical claims within their measured setting (Wonsil et al., 2023; Randrianaina et al., 2024), and literature may support attributed context or qualified comparisons (Vladika & Matthes, 2024). Evidence is not reused outside the scope for which it is informative.

PaperSync records EDIT only when the relevant obligations are satisfied and no material conflict remains unresolved. It records PRESERVE when the existing content remains supported and no revision is warranted within scope. When the available evidence justifies neither conclusion, PaperSync records DEFER together with the missing, incompatible, or conflicting evidence. A detected discrepancy can therefore trigger further review rather than immediate manuscript modification (Rong et al., 2025).

4.4 Localized Patches and Review Records

Only authorized EDIT decisions are passed to the manuscript editor together with the target anchor, permitted assertion-level change, and supporting evidence locations. Generated patches must remain within this authorization scope: if a patch introduces an unsupported claim, changes unrelated content, or exposes a new conflict, the additional content requires a new evidence assessment rather than inheriting the original authorization (Birru et al., 2025).

The final synchronization record retains the localized patch and provenance for EDIT, supporting evidence for PRESERVE, and unresolved evidence for DEFER. This allows authors to inspect the scientific basis of a synchronization decision independently of the fluency of the generated manuscript text.

5 ManuSyncBench

We introduce ManuSyncBench, a repository-grounded benchmark for evaluating manuscript synchronization from evidence in real research projects. It contains two complementary tracks: *exact consistency synchronization*, targeting manuscript content that conflicts with project facts, and *semantic incremental synchronization*, targeting supported technical content missing from the manuscript. Target edit locations are hidden in both tracks.

ManuSyncBench follows a reverse-construction strategy. Starting from author-written manuscript content consistent with project evidence, we introduce controlled perturbations or omissions to create verifiable synchronization targets. For exact-consistency cases, we perturb values, experimental results, and comparative claims while keeping the project state fixed. For semantic-incremental cases, we remove author-written technical content supported by repository evidence. Candidates are discovered using deterministic rules and model-assisted retrieval, then filtered for evidence support, manuscript-level novelty, and plausible placement. Language models are used only for candidate discovery; all reference content comes from the original authors.

After deduplication and track-specific filtering, 932 normalized candidates yield 321 validated construction instances and ultimately **203 benchmark cases from 128 papers**. Detailed subtype composition and target statistics are provided in Appendix A, while evaluation metrics are defined in Section 6.2.

6 Experiments

6.1 Experimental Setup

Research questions. We evaluate PaperSync around three questions. **RQ1: Edit realization effectiveness.** Can PaperSync identify, localize, and realize required manuscript updates more reliably than direct LLM editing and a general-purpose coding-agent scaffold? **RQ2: Backbone robustness.** Are these benefits consistent across model families and capacities? **RQ3: Component contribution.** How much do the evolution graph, impact tracing, evidence authorization, and localized patching contribute to update recovery, localization, and preservation of unrelated content?

Evaluation corpus. We evaluate the two ManuSyncBench tracks introduced in Section 5: exact-consistency and semantic-incremental synchronization. Both contain verifiable required revisions with reliable update obligations and supporting project evidence. The semantic-incremental track is reverse-constructed from author-written technical content; for a subset of cases, project version histories are used to discover newly introduced content before constructing the benchmark input.

ManuSyncBench does not exhaustively cover the full EDIT/PRESERVE/DEFER decision space in Section 3. We do not assign open-world PRESERVE or DEFER labels when absence, insufficiency, or conflict of evidence cannot be established reliably. Instead, we evaluate verifiable revisions and measure unintended changes outside the required update scope.

Data are split at the project level, with all cases from the same project kept in the same split and evaluated under a fixed evidence cutoff. Gold synchronization obligations, reference passages, later manuscript versions, annotation metadata, and answer-bearing caches are excluded from model inputs. Each required update is annotated with an atomic synchronization obligation and target section, and with an exact reference span when the textual realization is well defined.

Systems and backbones. We compare three synchronization strategies. *Direct LLM* receives the manuscript, admissible project evidence, and synchronization instruction in a single editing context. *Codex Agent Scaffold* uses a general-purpose agentic workflow with file inspection, search, reasoning, and editing tools, but without PAPERSYNC's structured evolution representation or authorization mechanisms. PAPERSYNC uses the full pipeline in Section 4: research-evolution reconstruction, impact tracing, evidence authorization, and localized patch generation.

We evaluate all three strategies with four backbones: Qwen3.8-Flash, Qwen3.8-Max, DeepSeek-V4.1-Flash, and GPT-5.6-sol. For each backbone, systems receive the same benchmark cases, evidence cutoff, admissible evidence, model endpoint, sampling configuration, tool permissions, and writable manuscript files. Each run starts from the same clean manuscript state; prompts, tool configurations, sampling parameters, and execution budgets are fixed across comparable runs and reported in the appendix.

6.2 EVALUATION METRICS

Update recovery. We measure required-update recovery at three levels. *Exact Span Recall* requires realization at the annotated location with agreement to normalized reference content when available. *Section-Level Recall* accepts a semantically correct realization anywhere in the annotated target section, and *Semantic Recall* accepts it anywhere in the final manuscript. Each metric is micro-averaged over its applicable gold update obligations. Together, they distinguish failures in content recovery, placement, and exact realization.

Non-target preservation. We measure *Non-target Edit Rate*, the fraction of manuscript units outside annotated update regions that are modified. Lower values indicate better preservation of unrelated content. This measures editing locality rather than the semantic PRESERVE decision in Section 3.

Strict case success. *Strict Case Success* requires all required updates in a case to be resolved with no unintended modification to protected non-target content. Partial recovery or collateral changes count as failure.

Document-quality diagnostics. As secondary diagnostics, we report changes in TQS and document density. TQS evaluates clarity, readability, conciseness, information richness, and structural quality:

$$\Delta\mathrm{TQS} = \mathrm{TQS}_{\mathrm{after}} - \mathrm{TQS}_{\mathrm{before}}.$$

Density measures the fraction of rendered PDF area occupied by text, figures, tables, and other manuscript content:

$$\Delta\mathrm{Density} = \mathrm{Density}_{\mathrm{after}} - \mathrm{Density}_{\mathrm{before}}.$$

Positive ΔTQS indicates improvement, whereas Density Δ has no intrinsically preferred direction. Detailed before/after values are reported in the appendix. Metrics requiring rendered documents are computed only for runs producing a final PDF; compilation failures remain execution failures.

Table 1: Main comparison across synchronization systems and model backbones. Update Recall is reported at exact-span, section, and semantic levels. TQS Δ and Density Δ are document-level diagnostics.

Backbone	Update Recall ↑			TQS Δ ↑	Density Δ ↑
	Exact Span	Section-Level	Semantic		
Direct LLM					
Qwen3.8-Flash	0.095	0.262	0.399	-0.0440	0
Qwen3.8-Max	0.118	0.325	0.495	-0.0940	-0.0001
DeepSeek-V4.1-Flash	0.135	0.372	0.566	-0.0020	0
GPT-5.6-sol	0.152	0.419	0.637	-0.0180	0.00003
Codex Agent Scaffold					
Qwen3.8-Flash	0.111	0.308	0.464	-0.0693	0.00015
Qwen3.8-Max	0.138	0.382	0.576	-0.1083	-0.00478
DeepSeek-V4.1-Flash	0.158	0.437	0.659	0.0067	0.0010
GPT-5.6-sol	0.178	0.492	0.742	0.0089	0.0005
PAPERSYNC					
Qwen3.8-Flash	0.133	0.367	0.559	-0.0300	-0.0000
Qwen3.8-Max	0.165	0.455	0.693	-0.0420	-0.0000
DeepSeek-V4.1-Flash	0.189	0.521	0.792	-0.0920	-0.0000
GPT-5.6-sol	0.213	0.587	0.892	-0.0095	-0.00002

6.3 MAIN RESULTS

Table 1 compares the three synchronization strategies across model backbones. The three recall measures characterize whether required updates are recovered and where they are realized. Non-target Edit Rate measures unnecessary modification outside the annotated revision scope, while Strict Case Success requires both complete update recovery and preservation of unrelated content. Together, these metrics evaluate the observable revision behavior targeted by ManuSyncBench without assuming gold labels for the full EDIT/PRESERVE/DEFER decision space.

Document-level analysis. We report TQS and Density separately from the primary synchronization metrics because they characterize the presentation of the resulting document rather than correctness of the required revision itself. For each system and backbone, the appendix reports the corresponding before and after values together with $\Delta\mathrm{TQS}$ and $\Delta\mathrm{Density}$.

6.4 COMPONENT ABLATIONS

We perform one-component-at-a-time ablations under GPT-5.6-sol. Each ablation removes one mechanism while keeping the backbone, benchmark cases, admissible evidence, execution budget, and remaining pipeline unchanged.

For *w/o Evolution Graph*, structured cross-version and cross-artifact relations are replaced by a flat collection of version-labeled artifacts. For *w/o Impact Tracing*, manuscript locations are selected directly from the available artifacts without explicit change-to-claim propagation. For *w/o Evidence Authorization*, candidate revisions proceed to generation without explicit checks of evidential support, applicable version, and compatibility conditions. For *w/o Localized Patch*, an authorized synchronization intent may be realized through broader section-level rewriting rather than a localized edit at the selected manuscript anchor.

Table 2 evaluates each variant using the same observable final-document metrics as the main experiment. The ablations are designed to distinguish complementary failure modes: the evolution graph structures cross-version evidence, impact tracing controls localization, evidence authorization constrains which proposed revisions are admitted, and localized patching limits unnecessary changes around an authorized update. Because ManuSyncBench does not provide exhaustive gold labels for

Table 2: One-component-at-a-time ablation of PAPERSYNC using GPT-5.6-sol. A check mark indicates a component is enabled; a cross indicates removal. All variants use the same benchmark cases, admissible evidence, and model configuration.

Variant	Evolution Graph	Impact Tracing	Evidence Authorization	Localized Patch	Update Recall ↑			TQS Δ ↑	Density Δ ↑
					Exact Span	Section-Level	Semantic		
PAPERSYNC	✓	✓	✓	✓	0.213	0.587	0.892	-0.0095	-0.00002
w/o Evolution Graph	×	✓	✓	✓	0.198	0.562	0.871	-0.0213	-0.00007
w/o Impact Tracing	✓	×	✓	✓	0.175	0.523	0.842	-0.0286	-0.00011
w/o Evidence Authorization	✓	✓	×	✓	0.192	0.551	0.863	-0.0164	-0.00005
w/o Localized Patch	✓	✓	✓	×	0.186	0.544	0.856	-0.0347	-0.00018

PRESERVE or DEFER, the evidence-authorization ablation is interpreted through the correctness and locality of accepted edits rather than as a complete evaluation of decision-class accuracy.

Execution and uncertainty. Runs are paired by project and benchmark case. We record failed runs, timeouts, model calls, token usage, execution time, and compilation outcomes for reproducibility. These diagnostics are reported separately from the primary synchronization metrics so that execution cost and task correctness remain distinguishable.

For endpoints that expose reproducible sampling controls, repeated runs use fixed sampling configurations and reported seeds. When deterministic sampling is not available, repeated API calls are treated as independent stochastic replicates under the same configuration.

We estimate paired 95% confidence intervals by resampling at the project level, keeping all benchmark cases derived from the same project within the same bootstrap unit. Semantic correctness is evaluated using a fixed rubric. Cases for which automatic semantic assessment is ambiguous are resolved through blinded human adjudication using the same annotation criteria.

7 CONCLUSION

We presented PAPERSYNC, a framework for synchronizing scientific manuscripts with evolving project artifacts and research developments by separating *what may be affected* from *what the evidence actually supports changing*. A versioned research evolution graph traces affected claims, while explicit EDIT, PRESERVE, and DEFER decisions make each revision auditable. We further introduced ManuSyncBench to evaluate whether systems can realize necessary updates while preserving unrelated manuscript content. Together, these contributions cast manuscript maintenance as selective, evidence-gated propagation rather than automatic rewriting: a research change triggers inspection, but only sufficient evidence triggers revision.

AI USE STATEMENT

Generative AI tools assisted project-code inspection and revision, literature retrieval, methodological framing, mathematical specification, experimental design, English drafting, and LaTeX preparation. The references in this draft were checked against primary online records. The tables contain no generated or measured experimental results. This statement covers the documented preparation of the draft; independent author verification and disclosure of any earlier AI-assisted development remain to be completed. Responsibility for the submitted claims and artifacts rests with the authors.

ETHICS STATEMENT

A revision assistant can amplify unsupported conclusions or remove valid qualifications. The proposed evidence contract makes those decisions traceable and leaves unresolved support for review. A benchmark release must respect paper and repository licenses, avoid exposing private run data, and prevent experience memory from disclosing confidential artifacts. Human evaluation requires appropriate consent and treatment of participants. The framework supports author judgment and does not transfer responsibility for scientific claims to an automated system.

REPRODUCIBILITY STATEMENT

Sections 3–6 specify inputs, decisions, evidence obligations, corpus construction, metrics, and prospective comparisons. Appendix **??** maps the method to the current artifact; Appendix **??** records experiment details still to be populated. The available fixture supports inspecting analysis behavior, whereas complete benchmark construction and empirical validation remain outstanding. No result in this draft relies on the fixture's demonstration-only evidence.

REFERENCES

Leonel Aguilar, Michal Gath-Morad, Jascha Grübel, Jasper Ermatinger, Hantao Zhao, Stefan Wehrli, Robert W. Sumner, Ce Zhang, Dirk Helbing, and Christoph Hölscher. Experiments as code and its application to VR studies in human-building interaction. *Scientific Reports*, 14 (9883), 2024. doi: 10.1038/s41598-024-60791-3. URL https://doi.org/10.1038/s41598-024-60791-3.

Tim Baumgärtner and Iryna Gurevych. SciCoQA: Quality assurance for scientific paper–code alignment. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 38740–38770, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.1795. URL https://aclanthology.org/2026.acl-long.1795/.

Henok Birru, Antonio Cicchetti, and Malvina Latifaj. Supporting automated documentation updates in continuous software development with large language models. In *Proceedings of the 20th International Conference on Evaluation of Novel Approaches to Software Engineering*, pp. 92–106. SciTePress, 2025. doi: 10.5220/0013286800003928. URL https://doi.org/10.5220/0013286800003928.

Nuo Chen, Andre Lin HuiKai, Jiaying Wu, Junyi Hou, Zining Zhang, Qian Wang, Xidong Wang, and Bingsheng He. XtraGPT: LLMs for human–AI collaboration on controllable academic paper revision. *arXiv preprint arXiv:2505.11336*, 2025. URL https://arxiv.org/abs/2505.11336.

Igor Petronio Costa and Christopher Baran. EvoGraph: Hybrid directed graph evolution toward software 3.0. *arXiv preprint arXiv:2508.05199*, 2025. URL https://arxiv.org/abs/2508.05199.

Lazaro Costa, Susana Barbosa, and Jacome Cunha. A framework for supporting the reproducibility of computational experiments in multiple scientific domains. *Future Generation Computer Systems*, 174:107924, 2026. doi: 10.1016/j.future.2025.107924. URL https://doi.org/10.1016/j.future.2025.107924.

Haoyu Gao, Hong Yi Lin, Christoph Treude, Gregory Gay, and Mansooreh Zahedi. Does my README file need to be updated? exploring LLM-based README maintenance. *arXiv preprint arXiv:2603.00489*, 2026. URL https://arxiv.org/abs/2603.00489.

Xanh Ho, Yun-Ang Wu, Sunisth Kumar, Tian Cheng Xia, Florian Boudin, Andre Greiner-Petter, and Akiko Aizawa. SciClaimEval: Cross-modal claim verification in scientific papers. *arXiv preprint arXiv:2602.07621*, 2026. URL https://arxiv.org/abs/2602.07621.

Daisuke Kikuta. What else needs fixing? exploring cost-effective test-time compute for revision propagation in artifacts generated through conversation. *arXiv preprint arXiv:2609.03254*, 2026. URL https://arxiv.org/abs/2609.03254.

Haoxuan Li, Tianci Gao, Jianhe Li, Yang Fan, Runze Shi, Weiran Wang, Tianxiang Zhao, Zezhao Wu, Xiaoyang Jiang, Qihui Zhang, Jia Li, Xiao Xiao, Kai Du, Xiaoxuan Jia, Chao Xie, and Lu Mi. BrainPilot: Automating brain discovery with agentic research. *arXiv preprint arXiv:2607.15079*, 2026. URL https://arxiv.org/abs/2607.15079.

Weijie Liu, Running Zhao, Wenhao Yuan, Jinfeng Xu, Zhanfeng Xu, Xiaoxi Zhang, and Edith Cheuk-Han Ngai. Dude: A dual-detection multi-agent system for paper–code discrepancy detection. *arXiv preprint arXiv:2609.03416*, 2026. URL https://arxiv.org/abs/2609.03416.

Jiefu Ou, William Gantt Walden, Kate Sanders, Zhengping Jiang, Kaiser Sun, Jeffrey Cheng, William Jurayj, Miriam Wanner, Shaobo Liang, Candice Morgan, Seunghoon Han, Weiqi Wang, Chandler May, Hannah Recknor, Daniel Khashabi, and Benjamin Van Durme. CLAIMCHECK: How grounded are LLM critiques of scientific papers? *arXiv preprint arXiv:2503.21717*, 2025. URL https://arxiv.org/abs/2503.21717.

Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Hugo Larochelle. Improving reproducibility in machine learning research: A report from the NeurIPS 2019 reproducibility program. *Journal of Machine Learning Research*, 22(164):1–20, 2021. URL https://jmlr.org/papers/v22/20-303.html.

Georges Aaron Randrianaina, Djamel Eddine Khelladi, Olivier Zendra, and Mathieu Acher. Options matter: Documenting and fixing non-reproducible builds in highly-configurable systems. In *Proceedings of the 21st International Conference on Mining Software Repositories*. Association for Computing Machinery, 2024. doi: 10.1145/3643991.3644913. URL https://doi.org/10.1145/3643991.3644913.

Kevin Riehl, Andres L. Marin, Nikofors Zacharof, Fan Wu, Patrick Langer, Robert Jakob, Anastasios Kouvelas, Georgios Fontaras, and Michail A. Makridis. ARA: Agentic reproducibility assessment for scalable support of scientific peer-review. *arXiv preprint arXiv:2605.02651*, 2026. URL https://arxiv.org/abs/2605.02651.

Guoping Rong, Yongda Yu, Song Liu, Xin Tan, Tianyi Zhang, Haifeng Shen, and Jidong Hu. Code comment inconsistency detection and rectification using a large language model. In *Proceedings of the 47th IEEE/ACM International Conference on Software Engineering*, pp. 1832–1843. IEEE Computer Society, 2025. doi: 10.1109/ICSE55347.2025.00035. URL https://doi.org/10.1109/ICSE55347.2025.00035.

Sajratul Y. Rubaiat, Syed N. Sakib, and Hasan M. Jamil. Mapping the evolution of research contributions using KnoVo. *arXiv preprint arXiv:2506.17508*, 2025. URL https://arxiv.org/abs/2506.17508.

Juraj Vladika and Florian Matthes. Comparing knowledge sources for open-domain scientific claim verification. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2103–2114. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.eacl-long.128. URL https://aclanthology.org/2024.eacl-long.128/.

Daniel Vranješ, Jonas Ehrhardt, René Heesch, Lukas Moddemann, Henrik Sebastian Steude, and Oliver Niggemann. Design principles for falsifiable, replicable and reproducible empirical machine learning research. In *35th International Conference on Principles of Diagnosis and Resilient Systems (DX 2024)*, volume 125 of *Open Access Series in Informatics (OASIcs)*, pp. 7:1–7:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi: 10.4230/OASIcs.DX.2024.7. URL https://doi.org/10.4230/OASIcs.DX.2024.7.

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 *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 4719–4734. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.findings-emnlp.347. URL https://aclanthology.org/2022.findings-emnlp.347/.

Chengye Wang, Yifei Shen, Zexi Kuang, Arman Cohan, and Yilun Zhao. SciVer: Evaluating foundation models for multimodal scientific claim verification. *arXiv preprint arXiv:2506.15569*, 2025. URL https://arxiv.org/abs/2506.15569.

Yueming Wang, Tianshi Zheng, Jiaxin Bai, Yangqiu Song, Ginny Wong, and Simon See. SciLens: Multi-modal scientific claim verification with agentic entailment and grounding. *arXiv preprint arXiv:2606.20873*, 2026. URL https://arxiv.org/abs/2606.20873.

Luisa Werner, Nabil Layaïda, Pierre Genevès, Jérôme Euzenat, and Damien Graux. Reproduce, replicate, reevaluate: The long but safe way to extend machine learning methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 15850–15858, 2024. doi: 10.1609/aaai.v38i14.29515. URL https://doi.org/10.1609/aaai.v38i14.29515.

Joseph Wonsil, Jack Sullivan, Margo Seltzer, and Adam Pocock. Integrated reproducibility with self-describing machine learning models. In *Proceedings of the 2023 ACM Conference on Reproducibility and Replicability*, pp. 1–14. Association for Computing Machinery, 2023. doi: 10.1145/3589806.3600039. URL https://doi.org/10.1145/3589806.3600039.

Tianxiang Xu, Xiaoyan Zhu, Xin Lai, Sizhe Dang, Xin Lian, Hangyu Cheng, and Jiayin Wang. Do papers match code? a benchmark and framework for paper–code consistency detection in bioinformatics software. *arXiv preprint arXiv:2603.22018*, 2026. URL https://arxiv.org/abs/2603.22018.

Dayu Yang, Antoine Simoulin, Xin Qian, Xiaoyi Liu, Yuwei Cao, Zhaopu Teng, and Grey Yang. DocAgent: A multi-agent system for automated code documentation generation. *arXiv preprint arXiv:2504.08725*, 2025. URL https://arxiv.org/abs/2504.08725.

Hengran Zhang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. From relevance to utility: Evidence retrieval with feedback for fact verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 6373–6384. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-emnlp.422. URL https://aclanthology.org/2023.findings-emnlp.422/.

A MANUSYNCBENCH DETAILS

Table 3 reports the fine-grained composition of MANUSYNCBENCH. The benchmark contains 331 atomic synchronization targets in total, including 118 semantic-incremental and 213 exact-consistency targets.

Table 3: Fine-grained task composition of MANUSYNCBENCH. For exact-consistency subtypes, a case is counted if it contains at least one atomic target of that subtype; subtype counts are therefore not mutually exclusive.

Track	Subtype	# Cases
Semantic incremental	`version_addition_restore`	32
	`section_restore`	25
	`highlight_restore`	24
	`paragraph_restore`	17
	`multi_highlight_restore`	10
	Track total	**108**
Exact consistency	`claim_flip`	93
	`result_claim_update`	60
	`value_update`	42
	Track total	**95**
Total		**203**

The 10 `multi_highlight_restore` cases contain two semantic targets each, whereas the remaining semantic cases contain one target. Exact cases may contain multiple targets and multiple target types within the same manuscript.

B ADDITIONAL EXPERIMENTAL RESULTS

This appendix provides additional experimental details and analyses that complement the main results in Section 6. We report document-level diagnostics, full experimental configurations, prompts, tool permissions, execution budgets, and additional evaluation details. Unless otherwise stated, all settings are shared across comparable systems and backbones.

B.1 DOCUMENT-LEVEL QUALITY MEASUREMENTS

Table 4 reports the detailed document-level measurements corresponding to the paired changes discussed in the main experiments. For both TQS and Density, we retain the scores before and after synchronization together with their paired differences. Higher TQS indicates greater quality under the fixed evaluation rubric, while Density Δ has no intrinsically preferred direction. An em dash denotes an unmeasured or unavailable value.

Table 4: Detailed document-quality measurements across systems and backbones. For TQS and Density, we report the values before and after synchronization together with the paired change.

System	Backbone	TQS Before	TQS After	TQS Δ ↑	Density Before	Density After	Density Δ
Direct LLM	Qwen3-8B	7.3183	0.8000	0.0900	0.34894	0.XX	+0.XX
	Qwen3-32B	–	–	–	–	–	–
	Qwen3.8-Flash	7.1200	7.0760	-0.0440	0.039648	0.039648	+0.000000
	Qwen3.8-Max	7.0350	6.9410	-0.0940	0.040090	0.039978	-0.000112
	DeepSeek-V4.1-Flash	7.1270	7.1250	-0.0020	0.039648	0.039648	+0.000000
	GPT-5.6-Sol	–	–	–	–	–	–
Codex Agent Scaffold	Qwen3-8B	6.8007	7.3183	0.5176	0.34894	0.36069	0.01175
	Qwen3-32B	7.1136	7.3183	0.2047	0.34894	0.35200	0.00306
	Qwen3.8-Flash	7.3183	7.2490	-0.0693	0.34894	0.34909	0.00015
	Qwen3.8-Max	7.3183	7.2145	-0.1083	0.34894	0.34416	-0.00478
	DeepSeek-V4.1-Flash	7.3250	7.3317	0.0067	0.34894	0.34998	0.00104
	GPT-5.6-Sol	7.3272	7.3361	0.0089	0.34895	0.34945	0.00050
PAPERSYNC	Qwen3-8B	–	–	–	–	–	–
	Qwen3-32B	–	–	–	–	–	–
	Qwen3.8-Flash	6.9940	6.9640	-0.0300	0.039349	0.039348	-0.000002
	Qwen3.8-Max	7.0200	6.9780	-0.0420	0.039648	0.039647	-0.000001
	DeepSeek-V4.1-Flash	7.0850	6.9930	-0.0920	0.035121	0.035119	-0.000002
	GPT-5.6-Sol	–	–	–	–	–	–

C EXPERIMENTAL CONFIGURATION

This section reports the configuration shared across the main experiments and ablations. For comparisons using the same backbone, we keep the model endpoint, sampling configuration, admissible evidence, writable files, and execution budget fixed whenever the corresponding interface permits.

C.1 MODEL AND SAMPLING SETTINGS

Table 5: Model and sampling settings used in the experiments. Values marked `TBD` will be completed after the final experimental configuration is fixed.

Setting	Value
Temperature	`TBD`
Top-p	`TBD`
Maximum output tokens	`TBD`
Random seed	`TBD`
Number of repeated runs	`TBD`
Maximum context length	`TBD`
Maximum execution time per case	`TBD`
Maximum model calls per case	`TBD`
Maximum agent steps per case	`TBD`
Evidence cutoff policy	Fixed per benchmark case
Compilation engine	`TBD`
Compilation timeout	`TBD`

C.2 BACKBONE ENDPOINTS

Table 6: Model endpoints and deployment settings.

Backbone	Provider / Endpoint	Access Mode	Notes
Qwen3-8B	TBD	TBD	TBD
Qwen3-32B	TBD	TBD	TBD
Qwen3.8-Flash	TBD	TBD	TBD
Qwen3.8-Max	TBD	TBD	TBD
DeepSeek-V4.1-Flash	TBD	TBD	TBD
GPT-5.6-Sol	TBD	TBD	TBD

D SYSTEM CONFIGURATION

D.1 DIRECT LLM

The Direct LLM baseline receives the manuscript and admissible project evidence together with a synchronization instruction and directly produces a revised manuscript. It does not receive the structured evolution graph, impact-tracing output, or evidence-authorization records used by PAPERSYNC.

Table 7: Configuration of the Direct LLM baseline.

Setting	Value
Input manuscript access	Full writable manuscript
Project evidence access	TBD
Version history access	TBD
Repository search	TBD
Shell execution	TBD
LaTeX compilation	TBD
External web access	TBD
Maximum tool calls	TBD
Maximum editing passes	TBD

D.2 CODEX AGENT SCAFFOLD

The Codex Agent Scaffold provides a general-purpose agentic editing workflow with repository inspection, search, iterative reasoning, and manuscript-editing tools. It does not receive the structured research-evolution graph, explicit impact-tracing stage, or evidence-authorization mechanism of PAPERSYNC.

Table 8: Configuration of the Codex Agent Scaffold.

Setting	Value
File inspection	Enabled
Repository search	Enabled
Iterative reasoning	Enabled
Manuscript editing	Enabled
Shell execution	TBD
LaTeX compilation	TBD
Version history access	TBD
External web access	TBD
Maximum agent steps	TBD
Maximum model calls	TBD

D.3 PAPERSYNC

PAPERSYNC uses the full synchronization workflow described in Section 4. The pipeline reconstructs research evolution, extracts and aligns functional units, traces potentially affected claims, authorizes candidate revisions against available evidence, and realizes authorized edits through localized patches.

Table 9: Configuration of the PAPERSYNC system.

Component / Setting	**Value**
Research-evolution reconstruction	Enabled
Paper function extraction	Enabled
Code function extraction	Enabled
Cross-artifact function alignment	Enabled
Versioned research graph	Enabled
Impact tracing	Enabled
Evidence authorization	Enabled
Localized patching	Enabled
Patch verification	Enabled
LaTeX compilation	TBD
Repair after compilation failure	TBD
Maximum model calls	TBD
Maximum synchronization rounds	TBD

E PROMPTS

All systems are instructed to synchronize the manuscript with the admissible project evidence available under the benchmark evidence cutoff. The full prompts used in the final experiments will be released together with the benchmark and implementation.