A RecSys Paper for $20: Why We Must Build, Evaluate, and Govern Autonomous Recommender Systems Research Labs (AutoRecLabs)

Joeran Beel[1,2,*], Bela Gipp[3], Tobias Vente[4], Moritz Baumgart[1,3], Philipp Meister[3] and Sinan Pourazari[1]

[1]University of Siegen, Siegen, Germany
[2]Recommender-Systems.com, Düsseldorf, Germany
[3]University of Göttingen, Göttingen, Germany
[4]University of Antwerp, Antwerp, Belgium

Abstract

Recommender systems research has automated parts of experimentation, but not yet the broader research process. We use *Automated Recommender Systems* (AutoRecSys) for tools that optimize recommenders inside largely human-defined pipelines. We define an *Autonomous Recommender Systems Research Lab* (AutoRecLab) as a system that connects multiple research stages and makes methodological decisions that researchers have not fully specified in advance. The "$20" in the title is a ballpark illustration of how inexpensive research production may become when work that traditionally takes weeks or months can increasingly be compressed into hours. Scientific validation, however, remains a separate task. This distinction is especially important in RecSys. Relevant methodological choices include implicit feedback, exposure, candidate construction, temporal ordering, user studies, online evaluation, and the relation between offline scores and user outcomes. A program can execute successfully while answering the wrong question or supporting an invalid claim.

We therefore call on the RecSys community to start building, evaluating, and governing AutoRecLabs now. Researchers should develop narrow prototypes for concrete tasks such as reproducing published experiments and auditing pipelines with known methodological faults. Benchmark organizers should compare these systems under shared task specifications, fixed resource conditions, protected checks, and expert adjudication where no reliable automatic ground truth exists. Tool developers should preserve the decisions, interventions, failures, and evidence produced during each run. Venues should establish disclosure and provenance requirements and pilot how AutoRecLab-generated research enters review and publication. The immediate challenge is not simply to generate more experiments, but to determine which research decisions can be delegated reliably and how their validity can be checked at scale. AutoRecLabs could reduce repetitive work, improve reproducibility, and make prior studies easier to extend. They could also amplify weak experimentation and submission volume. Progress should therefore be judged by the validity, provenance, and reuse of the resulting evidence rather than by the number of generated experiments or papers.

Keywords

AI4Science, AI4Research, AutoRecLab, Autonomous Research, Recommender Systems

1. Introduction

The ACM Recommender Systems (RecSys) conference reaches its 20th edition in 2026. Over the past two decades, the community has repeatedly adapted to new technologies and examined the methods used to evaluate recommender systems research [1, 2]. Our community advances algorithms and evaluation techniques and explores approaches such as large language models (LLMs) for generating recommendations. Research-process automation deserves similar attention. *Automated Recommender*

FRAME'26: Methodology First - Rethinking Research Assessment in RecSys Workshop, September 28, 2026, Minneapolis, Minnesota, USA

*Corresponding author.
joeran.beel@uni-siegen.de (J. Beel); bela.gipp@uni-goettingen.de (B. Gipp); tobias.vente@uantwerpen.be (T. Vente); moritz.baumgart@student.uni-siegen.de (M. Baumgart); Philipp.Meister@student.uni-siegen.de (P. Meister); Sinan.Pourazari@student.uni-siegen.de (S. Pourazari)
0000-0002-4537-5573 (J. Beel); 0000-0001-6522-3019 (B. Gipp); 0009-0003-8881-2379 (T. Vente); 0009-0007-1322-1450 (M. Baumgart); 0009-0008-6814-9668 (P. Meister); 0009-0001-8174-0259 (S. Pourazari)

© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-26

CEUR-WS.org/Vol-4341/paper1.pdf

Systems (AutoRecSys) tools such as Wang et al.'s AutoRec AutoML platform, BETA-Rec, LensKit-Auto, and LibRec-Auto support model search, tuning, or standardized experimentation within human-defined pipelines [3, 4, 5, 6].

Other research communities are exploring automation across larger parts of the research process. Autonomous research agents, sometimes framed as "AI Scientists", aim to generate research ideas, run experiments, and draft papers with limited human input [7, 8]. In computer science, related systems have discovered sorting routines that were integrated into the LLVM standard C++ sort library [9]. Program-search methods have also found improved mathematical constructions and algorithms [10]. In chemistry, research agents have planned and executed multistep experiments [11].

We use the concept of an *Autonomous Recommender Systems Research Lab* (AutoRecLab) to examine this broader automation in RecSys. An AutoRecLab connects research stages and participates in methodological decisions about the experimental setup and the interpretation of evidence. Section 3 gives an operational definition and distinguishes it from AutoRecSys. The relevant research extends from algorithmic experiments to user studies, interaction design, and online evaluation [12, 13]. Which decisions can be delegated, and how their quality should be assessed, are methodological questions for the RecSys community.

The title uses "$20" as a ballpark illustration of low-cost research production. For a workflow that otherwise takes weeks or months of human work, completion in hours could change both costs and research capacity. Whether the machine expenditure is $2, $20, or $200 matters less than the researcher time saved and the scientific value of the result. Evaluations should therefore account for setup, supervision, validation, and manuscript preparation alongside machine costs. The possibility of cheaper research production motivates this paper's attention to the quality and reuse of the resulting evidence.

Our contribution is a RecSys-specific framing and practical roadmap for research automation. We make three claims. First, AutoRecLab extends automation from optimizing a recommender to choosing and inspecting the surrounding research setup. Second, evaluation must distinguish technical execution from methodological validity. Third, near-term work should prioritize tasks with inspectable evidence, especially experiment implementation, reproduction, and methodological auditing. We call on RecSys researchers to build and test such systems, on tool developers to make their decisions and evidence inspectable, and on venue organizers to pilot evaluation and reporting procedures before autonomous research becomes routine.

Our own work combines the evaluation of general research agents with the development of RecSys-specific research automation. We first independently evaluated Sakana's AI Scientist [14, 15]. The study first appeared as a preprint and was subsequently published in *ACM SIGIR Forum*. It proposed benchmarks, research logs, pilot competitions, attribution mechanisms, and a strategic community discussion. An earlier arXiv version of the present paper adapted that agenda to recommender systems and introduced the AutoRecLab concept [16].

We subsequently released AutoRecLab Alpha and documented it in an ISG blog post [17]. This initial implementation translates natural-language RecSys research tasks into executable experiments. An accepted RecSys 2026 demo paper presents and evaluates an early AutoRecLab prototype [18]. The present FRAME version develops the methodological assessment, reproducibility, and governance aspects of this work. It builds on agenda items introduced in the earlier publications.

2. AI Research Agents

Recent surveys characterize AI research agents by their architecture, research stages, degree of autonomy, evaluation mechanisms, and validation requirements [19, 20, 21, 8]. Research automation predates the current LLM-agent wave. More than two decades ago, the Robot Scientist autonomously generated hypotheses, designed and executed functional-genomics experiments, interpreted the results, and used them to guide subsequent experiments [22]. Existing research agents differ in the stages they automate and in how their outputs are evaluated. Sakana AI's AI Scientist spans idea generation, experimental design, analysis, manuscript writing, and peer review [7]. Our independent evaluation of v1 examined

its use for RecSys research [15]. The system required a prepared experimental template. Five of twelve proposed experiments failed because of unresolved coding errors. Some executable experiments still produced logically flawed or misleading results. These observations concern the tested version and setup. They motivate assessing methodological validity separately from successful execution, a distinction also emphasized in work on the verification gap in autonomous research agents [21].

AI Scientist v2 later produced a fully AI-generated paper that passed peer review at an ICLR 2025 workshop [23]. Three manuscripts were submitted in that experiment. All were withdrawn after review under a preannounced transparency and ethics protocol.

Other systems cover overlapping parts of the research lifecycle. *AI-Researcher* combines literature review, hypothesis generation, experimentation, and manuscript drafting [24]. *Agent Laboratory* produces code and reports from a human-provided idea [25]. *Data-to-Paper* transforms raw data into a human-verifiable article [26]. *CodeScientist* combines idea generation with code-based experimentation [27], while AIGS adds an automated falsification stage [28]. AIDE focuses on machine-learning engineering through tree search over code implementations [29]. These systems provide possible components and comparison baselines for AutoRecLab studies.

Domain-specific systems connect research proposals to evidence from their application fields. Google's "AI Co-Scientist" generates hypotheses and research plans for biomedical questions [30]. In *Virtual Lab*, collaborating LLM-based scientists designed SARS-CoV-2 nanobodies that were subsequently validated experimentally [31]. For AutoRecLab, these examples motivate connecting general agent capabilities to domain-appropriate methods and validation.

Research automation also needs ways to build on earlier results. *AgentRxiv* enables agents to publish and retrieve intermediate findings across separate laboratories [32]. This addresses a function relevant to AutoRecLabs: preserving research outputs so that subsequent studies can use them.

Benchmarks assess capabilities that AutoRecLabs may rely on. MLAgentBench targets machine-learning experimentation [33], and MLE-bench targets machine-learning engineering [34]. ScienceAgentBench evaluates data-driven scientific discovery [35], while CORE-Bench targets computational reproducibility [36]. These benchmarks provide starting points for task design. The next section identifies the research decisions and RecSys-specific assumptions that an AutoRecLab evaluation should also examine.

3. From AutoRecSys to AutoRecLab

In this paper, we use *AutoRecSys* for systems that automate recommender development and evaluation within a largely human-defined experimental pipeline. The broader AutoML literature covers search over features, embedding dimensions, feature interactions, and model architectures [37].

AutoRec applies AutoML to model search and hyperparameter tuning for deep recommendation models [3]. BETA-Rec supports dataset preparation and a unified training, validation, tuning, and testing pipeline [4]. Further examples include LibRec-Auto [6], Auto-Surprise [38], Auto-CaseRec [39], and LensKit-Auto [5]. Such tools can reduce repetitive experimentation. In a comparison on 14 datasets, AutoML and AutoRecSys libraries often outperformed default configurations of conventional ML and RecSys libraries [40]. The same study found current AutoRecSys libraries less mature than established AutoML frameworks.

Definition. We define an *AutoRecLab* as a software system that carries out multiple connected stages of recommender systems research. Within a human-defined research remit, it makes methodological decisions that the researcher has not fully specified in advance. These decisions concern the research question, experimental design, or interpretation of evidence. This operational definition develops the concept introduced in the earlier version of this paper [16].

An AutoRecLab may analyze literature, design and execute experiments, interpret results, reproduce prior work, or prepare manuscripts. Its scope can cover a subset of these stages, with human approval at selected decision points. Evaluations should state which stages and decisions the system handles. The distinction concerns methodological discretion: AutoRecSys optimizes inside a human-defined

experimental sandbox; an AutoRecLab may also define, inspect, or revise that sandbox. AutoRecSys can therefore serve as a component of an AutoRecLab.

An AutoRecLab must account for the methodological diversity of recommender systems research. Algorithmic studies may compare learned models with popularity, recency, association-rule, nearest-neighbor, and other heuristic methods. The agent should select and configure methods according to the research question. Its experimental representation must therefore accommodate both model training and methods that require different forms of computation.

The data impose further requirements. Many RecSys datasets contain sparse interaction logs, and relevant tasks include ranking on implicit feedback [41]. Missing interactions are not reliable negatives. Observed interactions depend on prior exposure [42], while users, items, and candidate sets change over time [43]. Preprocessing and splitting choices can therefore alter the meaning of an experiment. Sampling, repeated interactions, and stochastic variation also need explicit treatment [44, 45, 46]. A pipeline may execute correctly while answering a different question from the one intended.

RecSys software makes many of these choices operational. LensKit [47], RecBole [48], and RecPack [49] provide domain-specific experimentation tools. Meta-frameworks such as OmniRec support execution across libraries [50]. Their settings and defaults affect filtering, negative sampling, candidate construction, temporal splitting, and metric computation. An agent using these tools must connect the implemented choices to the research question and the interpretation of the results.

The broader scope includes user studies, interaction design, behavioral questions, and long-term system effects. Offline, online, and user-study evaluations answer different questions [12]. Interface design can also change observed recommender effectiveness [13]. An AutoRecLab must therefore consider which evidence is appropriate for the intended claim, including when user-facing evaluation is needed.

We envision an integrated system that combines general research agents with AutoRecSys tools, RecSys-specific task representations, and methodological checks. It should link each research decision to the resulting implementation, evidence, and claim. AutoRecSys tools can therefore provide experimentation components inside the broader AutoRecLab workflow, while provenance and human oversight span the stages that the system automates.

An AutoRecLab can also be viewed as a recommender system for research decisions. At each stage, it must rank and select among hypotheses, datasets, methods, experimental designs, analyses, and follow-up actions under constraints such as time, cost, carbon, and ethics. This creates a two-way opportunity. AI-scientist systems may accelerate RecSys research, while RecSys methods for ranking, exploration–exploitation, multi-objective optimization, cold-start, counterfactual evaluation, and online learning may improve how autonomous research agents select and sequence research actions. Bandits, slates, constrained optimization, and causal inference are therefore not only tools that AutoRecLabs may use; they are areas in which the RecSys community may contribute to the broader development of autonomous research systems.

4. Roadmap: Build, Evaluate, and Govern

The RecSys community should not wait for end-to-end AutoRecLabs before studying research automation. Researchers can start with existing experiments whose inputs, outputs, and failure conditions are inspectable. Tool developers should expose the methodological decisions made by their systems. Benchmark organizers should test those decisions with protected checks where an executable answer exists and with independent RecSys experts where no single ground truth exists. They should also report the time and cost of expert, user-facing, or online validation separately from agent execution. The immediate objective is to identify which research decisions can be delegated reliably and which still require human control. The roadmap below turns this objective into concrete tasks for building, evaluating, and governing AutoRecLabs.

4.1. Build: Delegate One Research Decision at a Time

Reproduce one published experiment. Researchers should begin with a study that provides public data, code, and a concrete target result. The AutoRecLab should receive the paper, repository, data, and the result to reproduce. It should return an executable environment, the preprocessing and split, the candidate protocol, baseline and optimization settings, metric definitions, and the reproduced result. It should also list unresolved choices instead of silently inventing them. The evaluation should report the difference from the published result, every human correction, wall-clock time, and machine cost. Our RecSys 2026 demo provides an open-source starting point for this type of experiment[1] [18].

Audit one controlled methodological fault. A second task should test whether the system can detect a known problem rather than merely produce executable code. Benchmark creators can prepare variants of a valid RecSys experiment with one planted flaw, such as future information entering a temporal split, inconsistent candidate sets across algorithms, a changed metric definition, or a baseline evaluated under a different protocol. The system should identify the problem, explain which claim it affects, propose a correction, and rerun the affected analysis. Because the flaw is known in advance, the task has an inspectable answer while still requiring RecSys-specific methodological reasoning.

Increase autonomy one decision at a time. Researchers should vary the delegation boundary instead of comparing only "human" with "fully autonomous." In an initial condition, the human can fix the research question, data, split, candidate construction, baselines, metrics, and analysis. Further conditions can delegate one class of decisions at a time, for example split selection, baseline selection, or metric choice. Each condition should record incorrect decisions, human interventions, and the resulting change in cost and time. Only decisions that remain reliable under this test should be combined into broader autonomous workflows.

Include different RecSys method families. A prototype should not equate recommender research with training a neural model. Task sets should include simple popularity or recency baselines, nearest-neighbor methods, matrix-factorization approaches, and learned sequential or neural recommenders where appropriate. This exposes whether the system can select and configure methods according to the research question rather than applying one familiar training pipeline to every task.

4.2. Evaluate: Benchmark Claims, Not Just Runs

Start with three task families. A first AutoRecLab benchmark can focus on implementation, reproduction, and methodological auditing. An implementation task starts from a written experimental specification and checks whether the system realizes it correctly. A reproduction task starts from a published study and checks whether the system reconstructs the reported setup and result. An audit task starts from an executable study and checks whether the system detects a known methodological problem. Each task package should state the provided artifacts, the decisions the system may make, the stopping rule, and the artifacts that must be returned.

Measure four outcomes separately. Benchmark reports should distinguish task completion, methodological validity, reproducibility, and resource use. Task completion asks whether the requested artifact was produced and executed. Methodological validity asks whether the design and implementation answer the intended research question. Reproducibility asks whether another researcher or system can rerun the returned artifacts and recover the result. Resource use should include model calls or token use where available, compute, wall-clock time, monetary cost, and human intervention. A single success rate should not collapse these outcomes. Our initial AutoRecLab evaluation illustrates the reason: eight of nine runs produced code classified as bug-free, yet successful execution did not guarantee scientifically meaningful analysis [18].

Hold comparison conditions constant. Comparisons should use the same task specification, allowed sources, execution environment, resource budget, and intervention rules. To test whether RecSys-specific components add value, evaluators should compare the same underlying agent with and without those components under matched conditions. Human-designed pipelines and AutoRecSys tools can

[1] Source code: https://github.com/ISG-Siegen/AutoRecLab

provide additional baselines. Reports should separate gains from better methodological support from gains obtained simply through more compute, more model calls, or more human correction. Energy or environmental estimates should state their accounting assumptions [51, 52, 53].

Use protected checks where the answer is executable. Some properties can be checked automatically. Benchmark maintainers can test whether held-out interactions or future events leak into the training data, whether temporal ordering is respected, whether candidate sets follow the declared protocol, whether metric implementations match the specification, and whether required baselines were actually executed. Reproduction tasks can compare returned results with declared target ranges. Audit tasks can use a protected answer key for planted faults. These checks should establish conformance to the task, not replace scientific judgment.

Use expert adjudication where the claim is methodological. Questions such as whether an offline metric supports a user-benefit claim or whether two evaluation protocols answer the same research question do not have a simple executable ground truth. Such tasks should be reviewed by independent RecSys experts using a short, predeclared rubric. The benchmark should retain disagreements and the reasons for them instead of converting every judgment into one opaque score. This makes the cost and uncertainty of scientific validation visible.

Publish failure distributions, not only aggregate scores. Benchmark maintainers should classify failures at the task level. Useful categories include execution failure, incorrect protocol, invalid interpretation, missing evidence, and required human rescue. They should retain successful and failed runs, stochastic repetitions, and human corrections. Dataset-selection rationales should also be recorded [54], and repeated runs should remain visible rather than being reduced to one selected outcome [45]. This allows later work to identify which decisions are becoming reliable and which still require human control.

4.3. Govern: Make Generated Evidence Reviewable

Define a minimum AutoRecLab record. Every reported AutoRecLab result should have a machine-readable record that is sufficient to reconstruct how it was produced. At minimum, the record should identify the research task and its version, model and tool versions, retrieved sources, dataset and preprocessing versions, split and candidate protocol, baselines, metrics, seeds, resource budget, attempted runs, failed runs, and human interventions. Each main claim should link to the result and artifact that support it. A changed methodological decision should create a new recorded condition rather than silently overwrite the previous one.

Give reviewers structured evidence, not raw agent logs. Reviewers should not be expected to read complete model traces. A submission that uses an AutoRecLab can provide a compact summary of delegated decisions, human corrections, resource use, unresolved checks, and links from claims to executable artifacts. Automated pre-checks can test whether the required fields and artifacts are present and internally consistent. Detailed traces can remain available for targeted inspection when a discrepancy or methodological question requires them. This keeps the review burden bounded while preserving inspectability.

Pilot one review workflow before changing venue policy. A workshop, artifact track, or reproducibility challenge can test the process before a conference makes broader rules. Authors can submit the manuscript, executable artifact, and AutoRecLab record. Automated checks can run first; reviewers then receive the structured summary and inspect selected evidence. Organizers should measure additional review time, problems detected, false alarms, author effort, and disagreements between automated and human assessment. The result of the pilot should be a revised reporting schema and a list of checks that were actually useful, not a new mandatory policy based only on speculation.

Keep responsibility with human authors. Human authors should remain responsible for the research question, the claims submitted for publication, and the decision to release or deploy a study. For user-facing or online experiments, the record should also identify who approved the study, consent or privacy procedures, stopping criteria, and any changes proposed by the agent. Privacy, fairness, exposure, behavioral effects, and long-term feedback should be treated as research constraints rather than optional post-hoc checks.

Create one shared community package. A concrete first community deliverable would be a small public package rather than an open-ended "AI Researcher" competition. It should contain task specifications for implementation, reproduction, and auditing; executable environments and reference artifacts; protected checks where appropriate; a common run-record schema; and baseline results for at least one general research agent and one RecSys-specific system. Once such a package exists, RecSys workshops or challenges can compare systems under the same tasks and budgets. Related competitions have already been proposed in information retrieval [15], and existing RecSys infrastructure such as OmniRec can support standardized execution across frameworks [50].

5. Discussion

5.1. More Research Is Not Necessarily More Knowledge

Cheaper research production can increase output without increasing knowledge. Conventional studies require researcher time for implementation, baseline configuration, and hyperparameter optimization. Executing experiments, analyzing results, and preparing manuscripts add further work. An AutoRecLab could examine more hypotheses, models, datasets, preprocessing choices, splits, metrics, and robustness conditions. It could continue experiments without continuous supervision, react to intermediate results, and revisit earlier decisions. It could also retain negative and inconclusive results, reducing repeated work on unsuccessful ideas.

AutoRecLabs may generate many minor model variations, benchmark-specific improvements, or technically complete studies that resolve little uncertainty. Prior work has questioned apparent algorithmic progress when comparisons rely on weak baselines [55]. Automation could amplify that problem by generating more offline improvements without addressing exposure, interaction, longitudinal effects, or deployment. Evaluation should therefore distinguish increased research output from advances in cumulative knowledge.

AutoRecLabs could make it easier to build directly on prior work. Extending a published study often requires reconstructing implementation details and resolving dependencies. Researchers must also locate the data, repeat preprocessing, and identify the configurations behind the reported results. An AutoRecLab could treat a paper, repository, and artifact package as the starting point of a new research process. It could reconstruct the experiment, compare reproduced and reported results, identify unresolved choices, and prepare a verified environment for extension.

Prior methods could then be rerun under common datasets, splits, candidate sets, metrics, and baseline implementations. Comparisons could be updated as new methods and datasets become available. This would move the research record closer to a maintained body of evidence than a sequence of weakly connected papers. Reproducing, consolidating, or invalidating earlier findings may become less laborious, but these activities would remain scientific contributions. The value of an AutoRecLab may therefore depend less on the number of studies it produces than on whether it helps determine which findings remain valid across experimental conditions.

5.2. Inspectable Evidence for Reproducibility and Research Evaluation

Research automation can improve reproducibility when the process remains inspectable. For computational studies, executable artifacts and their configurations allow another researcher or AutoRecLab to rerun the experimental steps. Retained decisions, failures, and outputs document how the evidence was produced. Discrepancies can then be traced to software versions, parameters, preprocessing choices, candidate definitions, or stochastic outcomes.

AutoRecLabs could also make robustness analysis more routine. A reproduction could test alternative seeds [45, 46], datasets [54], temporal splits [43], and preprocessing choices [44]. Further tests could vary candidate sets, hyperparameter budgets, baseline implementations, or metric definitions. Reproduction would then move beyond verifying one reported number toward identifying the conditions under which a conclusion holds.

Reproducibility in RecSys also requires reconstructing the assumptions behind the metrics. Candidate construction, exposure, temporal ordering, and the treatment of missing interactions shape the meaning of a result. The interpretation must connect those choices to the stated claim. Two implementations may report similar scores while representing different recommendation problems. Reproducibility therefore includes reconstructing the assumptions under which the numbers are meaningful. This potential depends on provenance: an AutoRecLab that discards failed experiments, intermediate changes, retrieved sources, and human interventions may be no easier to reproduce than current work.

The same artifacts can also support research evaluation. Automated checks can reproduce selected experiments and test internal consistency, while expert reviewers assess novelty, claim scope, user impact, and scientific importance. This division could redirect reviewer time from reconstructing pipelines toward interpreting evidence.

The boundary is methodological. Recommendation quality has no single ground truth, and the selected proxy may not support the intended claim. Some questions therefore require expert judgment, simulation, an online experiment, or a user study. User studies and qualitative research also require forms of assessment that cannot be reduced to executable checks.

5.3. Automation Shifts the Publication Bottleneck

If experiments and manuscripts become inexpensive to produce, validation and review may become the limiting resources. Submission volume may exceed the capacity of current review systems. AutoRecLabs could generate many variations of one idea, convert exploratory analyses into complete papers, or resubmit related work across venues with little additional effort. More submissions would not necessarily represent more knowledge. The publication bottleneck would shift from producing research to determining which results are valid, novel, reproducible, and worth retaining.

AutoRecLabs may partly address the problem they create. The same systems used to produce studies could reproduce submissions, compare them with prior work, inspect artifacts, and test whether main claims survive alternative evaluations. Automated triage could identify incomplete or internally inconsistent submissions before they consume substantial reviewer time. For RecSys triage, the task formulation and the relation between metrics and claims both need inspection. The evaluator must also consider whether offline evidence is sufficient and whether relevant user or stakeholder outcomes were omitted.

Technical filtering leaves the underlying incentives unchanged. If career progression, venue prestige, and research assessment continue to reward publication counts, lower production costs may increase fragmentation and redundancy. Maintained artifacts, successful reproductions, consolidated evidence, and verified extensions may need greater recognition than another standalone manuscript.

Access to AutoRecLabs may also be uneven. Organizations with proprietary models, larger compute budgets, commercial scholarly databases, and extensive execution infrastructure may outperform smaller research groups. Research automation could therefore lower some barriers while concentrating influence among organizations that can operate the most capable systems.

Transparent resource reporting would make these differences visible; unequal access would remain. Open-source systems, public execution infrastructure, controlled-budget benchmarks, and open scholarly resources could provide alternative conditions for participation and comparison. The community should compare efficiency separately from unrestricted scale. A system may obtain stronger results by executing thousands of experiments. Its methodological quality and cost still need comparison with systems operating under smaller budgets. Cost, access, and researcher effort should also remain visible in system comparisons. Energy use and environmental impact are additional parts of the scientific assessment [51, 52, 53].

6. Conclusion

AutoRecLab places methodological decisions at the center of research automation. A useful system should make defensible experimental choices, preserve their evidence, and keep claims within the

scope of that evidence. These requirements apply to partial workflows as well as more autonomous laboratories.

Several actions should start now. Researchers should formulate inspectable automation tasks, report resource budgets and human interventions, and retain provenance that links decisions to results and claims. Tool builders should expose RecSys-specific choices such as splits, candidate construction, baselines, and evaluation protocols instead of hiding them behind defaults. Benchmark organizers should establish shared implementation, reproduction, and methodological-audit tasks under controlled resource conditions. Conference and workshop organizers should pilot how AutoRecLab-generated evidence is disclosed and reviewed before attempting end-to-end "AI Researcher" competitions. None of these steps requires full autonomy.

AutoRecLabs could accelerate experimentation, strengthen reproducibility, and make prior studies easier to extend. They could also amplify benchmark-driven incrementalism, submission volume, opaque assessment, and unequal access to research infrastructure. Building, evaluating, and governing AutoRecLabs should therefore proceed together. Research automation is already changing how scientific artifacts are produced. The RecSys community should now actively shape how AutoRecLabs are built, how their claims are validated, and how their outputs enter the scientific record.

Declaration on Generative AI

During the preparation of this work, the authors used OpenAI ChatGPT for language editing, structural revision, and assistance with literature search and reference checking. The authors reviewed and edited the content and take full responsibility for the publication.

References

[1] J. Beel, C. Breitinger, S. Langer, A. Lommatzsch, B. Gipp, Towards reproducibility in recommender-systems research, User Modeling and User-Adapted Interaction 26 (2016) 69–101. URL: https://doi.org/10.1007/s11257-016-9174-x. doi:10.1007/s11257-016-9174-x.

[2] J. Beel, Proposal for evidence-based best-practices for recommender systems evaluation, in: C. Bauer, A. Said, E. Zangerle (Eds.), Evaluation Perspectives of Recommender Systems: Driving Research and Education (Dagstuhl Seminar 24211), Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, pp. 66–68. URL: https://doi.org/10.4230/DagRep.14.5.58. doi:10.4230/DagRep.14.5.58, contribution in Dagstuhl Reports, volume 14, issue 5; the DOI resolves to the complete seminar report.

[3] T.-H. Wang, X. Hu, H. Jin, Q. Song, X. Han, Z. Liu, AutoRec: An automated recommender system, in: Proceedings of the 14th ACM Conference on Recommender Systems, Association for Computing Machinery, 2020, pp. 582–584. URL: https://doi.org/10.1145/3383313.3411529. doi:10.1145/3383313.3411529.

[4] Z. Meng, R. McCreadie, C. Macdonald, I. Ounis, S. Liu, Y. Wu, X. Wang, S. Liang, Y. Liang, G. Zeng, J. Liang, Q. Zhang, BETA-Rec: Build, evaluate and tune automated recommender systems, in: Proceedings of the 14th ACM Conference on Recommender Systems, Association for Computing Machinery, 2020, pp. 588–590. URL: https://doi.org/10.1145/3383313.3411524. doi:10.1145/3383313.3411524.

[5] T. Vente, M. D. Ekstrand, J. Beel, Introducing lenskit-auto, an experimental automated recommender system (AutoRecSys) toolkit, in: Proceedings of the 17th ACM Conference on Recommender Systems, Association for Computing Machinery, 2023, pp. 1212–1216. URL: https://doi.org/10.1145/3604915.3610656. doi:10.1145/3604915.3610656.

[6] N. Sonboli, M. Mansoury, Z. Guo, S. Kadekodi, W. Liu, Z. Liu, A. Schwartz, R. Burke, librec-auto: A tool for recommender systems experimentation, in: Proceedings of the 30th ACM International Conference on Information and Knowledge Management, Association for Computing

Machinery, 2021, pp. 4584–4593. URL: https://doi.org/10.1145/3459637.3482006. doi:10.1145/3459637.3482006.

[7] C. Lu, C. Lu, R. T. Lange, J. Foerster, J. Clune, D. Ha, The AI scientist: Towards fully automated open-ended scientific discovery, 2024. URL: https://arxiv.org/abs/2408.06292. doi:10.48550/arXiv.2408.06292. arXiv:2408.06292.

[8] J. Beel, Two decades after the robot scientist: A survey of AI scientists, autonomous research, and why validation still sets the pace, ResearchGate preprint, 2026. URL: https://www.researchgate.net/publication/414552473_Two_Decades_After_the_Robot_Scientist_A_Survey_of_AI_Scientists_Autonomous_Research_and_Why_Validation_Still_Sets_the_Pace. doi:10.13140/RG.2.2.11480.66569, preprint published 21 September 2026.

[9] D. J. Mankowitz, A. Michi, A. Zhernov, M. Gelmi, M. Selvi, C. Paduraru, E. Leurent, S. Iqbal, J.-B. Lespiau, A. Ahern, T. Köppe, K. Millikin, S. Gaffney, S. Elster, J. Broshear, C. Gamble, K. Milan, R. Tung, M. Hwang, T. Cemgil, M. Barekatain, Y. Li, A. Mandhane, T. Hubert, J. Schrittwieser, D. Hassabis, P. Kohli, M. Riedmiller, O. Vinyals, D. Silver, Faster sorting algorithms discovered using deep reinforcement learning, Nature 618 (2023) 257–263. URL: https://doi.org/10.1038/s41586-023-06004-9. doi:10.1038/s41586-023-06004-9.

[10] B. Romera-Paredes, M. Barekatain, A. Novikov, M. Balog, M. P. Kumar, E. Dupont, F. J. R. Ruiz, J. S. Ellenberg, P. Wang, O. Fawzi, P. Kohli, A. Fawzi, Mathematical discoveries from program search with large language models, Nature 625 (2024) 468–475. URL: https://doi.org/10.1038/s41586-023-06924-6. doi:10.1038/s41586-023-06924-6.

[11] D. A. Boiko, R. MacKnight, B. Kline, G. Gomes, Autonomous chemical research with large language models, Nature 624 (2023) 570–578. URL: https://www.nature.com/articles/s41586-023-06792-0. doi:10.1038/s41586-023-06792-0.

[12] J. Beel, S. Langer, A comparison of offline evaluations, online evaluations, and user studies in the context of research-paper recommender systems, in: Research and Advanced Technology for Digital Libraries, Springer International Publishing, 2015, pp. 153–168. URL: https://doi.org/10.1007/978-3-319-24592-8_12. doi:10.1007/978-3-319-24592-8_12.

[13] J. Beel, H. Dixon, The "unreasonable" effectiveness of graphical user interfaces for recommender systems, in: Adjunct Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization, Association for Computing Machinery, 2021, pp. 22–28. URL: https://doi.org/10.1145/3450614.3461682. doi:10.1145/3450614.3461682.

[14] J. Beel, M.-Y. Kan, M. Baumgart, Evaluating sakana's AI scientist: Bold claims, mixed results, and a promising future?, 2025. URL: https://arxiv.org/abs/2502.14297. doi:10.48550/arXiv.2502.14297. arXiv:2502.14297, preprint first posted 20 February 2025; later published in ACM SIGIR Forum.

[15] J. Beel, M.-Y. Kan, M. Baumgart, Evaluating sakana's AI scientist: Bold claims, mixed results, and a promising future?, ACM SIGIR Forum 59 (2025) 1–20. URL: https://doi.org/10.1145/3769733.3769747. doi:10.1145/3769733.3769747.

[16] J. Beel, B. Gipp, T. Vente, M. Baumgart, P. Meister, From AutoRecSys to AutoRecLab: A call to build, evaluate, and govern autonomous recommender-systems research labs, 2025. URL: https://arxiv.org/abs/2510.18104. doi:10.48550/arXiv.2510.18104. arXiv:2510.18104, first posted on arXiv on 20 October 2025.

[17] J. Beel, M. Baumgart, P. Meister, AutoRecLab Alpha: Bringing the AI scientist to recommender systems, Intelligent Systems Group blog, 2025. URL: https://isg.beel.org/blog/2025/10/28/autoreclab-alpha-bringing-the-ai-scientist-to-recommender-systems/.

[18] M. Baumgart, P. Meister, J. Krell, M. Schmidt, B. Gipp, J. Beel, AutoRecLab: An autonomous recommender systems lab, in: Proceedings of the 20th ACM Conference on Recommender Systems, Association for Computing Machinery, 2026. URL: https://doi.org/10.1145/3773078.3841273. doi:10.1145/3773078.3841273.

[19] S. Ren, C. Xie, P. Jian, Z. Ren, C. Leng, J. Zhang, Towards scientific intelligence: A survey of LLM-based scientific agents, 2025. URL: https://arxiv.org/abs/2503.24047. doi:10.48550/arXiv.2503.24047. arXiv:2503.24047, first posted 31 March 2025; metadata follows arXiv v3, revised 2

February 2026.

[20] T. Zheng, Z. Deng, H. T. Tsang, W. Wang, J. Bai, Z. Wang, Y. Song, From automation to autonomy: A survey on large language models in scientific discovery, in: Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2025, pp. 17733–17750. URL: https://aclanthology.org/2025.emnlp-main.895/. doi:10.18653/v1/2025.emnlp-main.895.

[21] T. Ding, A. Nannapaneni, B. Liu, L. Zhang, Autonomous research agents: A survey of AI scientists and the verification gap, 2026. URL: https://arxiv.org/abs/2608.05179. doi:10.48550/arXiv.2608.05179. arXiv:2608.05179.

[22] R. D. King, K. E. Whelan, F. M. Jones, P. G. K. Reiser, C. H. Bryant, S. H. Muggleton, D. B. Kell, S. G. Oliver, Functional genomic hypothesis generation and experimentation by a robot scientist, Nature 427 (2004) 247–252. URL: https://doi.org/10.1038/nature02236. doi:10.1038/nature02236.

[23] Y. Yamada, R. T. Lange, C. Lu, S. Hu, C. Lu, J. Foerster, J. Clune, D. Ha, The AI scientist-v2: Workshop-level automated scientific discovery via agentic tree search, 2025. URL: https://arxiv.org/abs/2504.08066. doi:10.48550/arXiv.2504.08066. arXiv:2504.08066.

[24] J. Tang, L. Xia, Z. Li, C. Huang, AI-researcher: Autonomous scientific innovation, in: Advances in Neural Information Processing Systems, volume 38, 2025, pp. 10477–10516. URL: https://papers.nips.cc/paper_files/paper/2025/hash/0d904d300a105809a2114d727851e759-Abstract-Conference.html. doi:10.52202/085713-0320.

[25] S. Schmidgall, Y. Su, Z. Wang, X. Sun, J. Wu, X. Yu, J. Liu, M. Moor, Z. Liu, E. Barsoum, Agent laboratory: Using LLM agents as research assistants, in: Findings of the Association for Computational Linguistics: EMNLP 2025, Association for Computational Linguistics, Suzhou, China, 2025, pp. 5977–6043. URL: https://aclanthology.org/2025.findings-emnlp.320/. doi:10.18653/v1/2025.findings-emnlp.320.

[26] T. Ifargan, L. Hafner, M. Kern, O. Alcalay, R. Kishony, Autonomous LLM-driven research — from data to human-verifiable research papers, NEJM AI 2 (2025). URL: https://doi.org/10.1056/AIoa2400555. doi:10.1056/AIoa2400555.

[27] P. Jansen, O. Tafjord, M. Radensky, P. Siangliulue, T. Hope, B. Dalvi Mishra, B. P. Majumder, D. S. Weld, P. Clark, CodeScientist: End-to-end semi-automated scientific discovery with code-based experimentation, in: Findings of the Association for Computational Linguistics: ACL 2025, Association for Computational Linguistics, Vienna, Austria, 2025, pp. 13370–13467. URL: https://aclanthology.org/2025.findings-acl.692/. doi:10.18653/v1/2025.findings-acl.692.

[28] Z. Liu, K. Liu, Y. Zhu, X. Lei, Z. Yang, Z. Zhang, P. Li, Y. Liu, AIGS: Generating science from AI-powered automated falsification, 2024. URL: https://arxiv.org/abs/2411.11910. doi:10.48550/arXiv.2411.11910. arXiv:2411.11910.

[29] Z. Jiang, D. Schmidt, D. Srikanth, D. Xu, I. Kaplan, D. Jacenko, Y. Wu, AIDE: AI-driven exploration in the space of code, 2025. URL: https://arxiv.org/abs/2502.13138. doi:10.48550/arXiv.2502.13138. arXiv:2502.13138.

[30] J. Gottweis, W.-H. Weng, A. Daryin, T. Tu, P. Sirkovic, A. Myaskovsky, G. Glowaty, F. Weissenberger, A. Orlandi, D. Popovici, A. Palepu, K. Rong, R. Tanno, K. Saab, F. Zhang, J. Blum, A. Carroll, K. Kulkarni, N. Tomasev, D. Zverinski, I. Rendulic, E. Vedadi, F. Hasler, L. Rimanic, M. Boia, I. Budiselic, B. Feinstein, M. Bellaiche, T. Sheffer, J. Freyberg, J. Ratcliff, O. Bertolli, K. Chou, A. Hassidim, B. Gokturk, A. Vahdat, Y. Guan, V. Dhillon, E. D. Vaishnav, B. Lee, T. R. D. Costa, J. R. Penadés, G. Peltz, Y. Matias, J. Manyika, D. Hassabis, Y. Xu, P. Kohli, A. Pawlosky, A. Karthikesalingam, V. Natarajan, Accelerating scientific discovery with co-scientist, Nature 655 (2026) 487–496. URL: https://doi.org/10.1038/s41586-026-10644-y. doi:10.1038/s41586-026-10644-y.

[31] K. Swanson, W. Wu, N. L. Bulaong, J. E. Pak, J. Zou, The virtual lab of AI agents designs new SARS-CoV-2 nanobodies, Nature 646 (2025) 716–723. URL: https://doi.org/10.1038/s41586-025-09442-9. doi:10.1038/s41586-025-09442-9.

[32] S. Schmidgall, M. Moor, Agentrxiv: Towards collaborative autonomous research, 2025. URL: https://arxiv.org/abs/2503.18102. doi:10.48550/arXiv.2503.18102. arXiv:2503.18102.

[33] Q. Huang, J. Vora, P. Liang, J. Leskovec, MLAgentBench: Evaluating language agents on machine

learning experimentation, in: Proceedings of the 41st International Conference on Machine Learning, volume 235 of *Proceedings of Machine Learning Research*, 2024, pp. 20271–20309. URL: https://proceedings.mlr.press/v235/huang24y.html.

[34] J. S. Chan, N. Chowdhury, O. Jaffe, J. Aung, D. Sherburn, E. Mays, G. Starace, K. Liu, L. Maksin, T. Patwardhan, A. Madry, L. Weng, MLE-bench: Evaluating machine learning agents on machine learning engineering, in: International Conference on Learning Representations, 2025. URL: https://proceedings.iclr.cc/paper_files/paper/2025/hash/7e3767db483c942b883eb4f8cfb74e31-Abstract-Conference.html.

[35] Z. Chen, S. Chen, Y. Ning, Q. Zhang, B. Wang, B. Yu, Y. Li, Z. Liao, C. Wei, Z. Lu, V. Dey, M. Xue, F. N. Baker, B. Burns, D. Adu-Ampratwum, X. Huang, X. Ning, S. Gao, Y. Su, H. Sun, ScienceAgentBench: Toward rigorous assessment of language agents for data-driven scientific discovery, in: International Conference on Learning Representations, 2025. URL: https://proceedings.iclr.cc/paper_files/paper/2025/hash/f12b4df26344f3be803c06b555252efe-Abstract-Conference.html.

[36] Z. S. Siegel, S. Kapoor, N. Nadgir, B. Stroebl, A. Narayanan, CORE-Bench: Fostering the credibility of published research through a computational reproducibility agent benchmark, Transactions on Machine Learning Research (2024). URL: https://openreview.net/forum?id=BsMMc4MEGS.

[37] R. Zheng, L. Qu, B. Cui, Y. Shi, H. Yin, AutoML for deep recommender systems: A survey, ACM Transactions on Information Systems 41 (2023) 101:1–101:38. URL: https://doi.org/10.1145/3579355. doi:10.1145/3579355.

[38] R. Anand, J. Beel, Auto-Surprise: An automated recommender-system (AutoRecSys) library with tree of parzens estimator (TPE) optimization, in: Proceedings of the 14th ACM Conference on Recommender Systems, Association for Computing Machinery, 2020, pp. 585–587. URL: https://doi.org/10.1145/3383313.3411467. doi:10.1145/3383313.3411467.

[39] S. Gupta, J. Beel, Auto-CaseRec: Automatically selecting and optimizing recommendation-systems algorithms, OSF Preprints, 2020. URL: https://doi.org/10.31219/osf.io/4znmd. doi:10.31219/osf.io/4znmd.

[40] T. Vente, L. Wegmeth, J. Beel, The potential of AutoML for recommender systems, in: Adjunct Proceedings of the 33rd ACM Conference on User Modeling, Adaptation and Personalization, Association for Computing Machinery, 2025, pp. 371–378. URL: https://doi.org/10.1145/3708319.3734173. doi:10.1145/3708319.3734173.

[41] L. Wegmeth, T. Vente, J. Beel, Recommender systems algorithm selection for ranking prediction on implicit feedback datasets, in: Proceedings of the 18th ACM Conference on Recommender Systems, Association for Computing Machinery, 2024, pp. 1163–1167. URL: https://doi.org/10.1145/3640457.3691718. doi:10.1145/3640457.3691718.

[42] A. Collins, D. Tkaczyk, A. Aizawa, J. Beel, Position bias in recommender systems for digital libraries, in: Transforming Digital Worlds, Springer International Publishing, 2018, pp. 335–344. URL: https://doi.org/10.1007/978-3-319-78105-1_37. doi:10.1007/978-3-319-78105-1_37.

[43] T. Scheidt, J. Beel, Time-dependent evaluation of recommender systems, in: Proceedings of the 1st Workshop on the Perspectives on the Evaluation of Recommender Systems (PERSPECTIVES 2021), volume 2955 of *CEUR Workshop Proceedings*, 2021. URL: https://ceur-ws.org/Vol-2955/paper10.pdf.

[44] J. Beel, V. Brunel, Data pruning in recommender systems research: Best-practice or malpractice?, in: Late-Breaking Results of the 13th ACM Conference on Recommender Systems, volume 2431 of *CEUR Workshop Proceedings*, 2019, pp. 26–30. URL: https://ceur-ws.org/Vol-2431/paper6.pdf.

[45] L. Wegmeth, T. Vente, L. Purucker, J. Beel, The effect of random seeds for data splitting on recommendation accuracy, in: Proceedings of the 3rd Workshop on the Perspectives on the Evaluation of Recommender Systems (PERSPECTIVES 2023), co-located with the 17th ACM Conference on Recommender Systems (RecSys 2023), volume 3476 of *CEUR Workshop Proceedings*, CEUR-WS.org, Singapore, Singapore, 2023. URL: https://ceur-ws.org/Vol-3476/paper4.pdf, iSSN 1613-0073.

[46] S. Schulz, The effect of random seeds for data splitting on recommendation accuracy [reproduced], OSF Preprints, 2024. URL: https://doi.org/10.31219/osf.io/5vu3c. doi:10.31219/osf.io/5vu3c.

[47] M. D. Ekstrand, LensKit for python: Next-generation software for recommender systems ex-

periments, in: Proceedings of the 29th ACM International Conference on Information and Knowledge Management, Association for Computing Machinery, 2020, pp. 2999–3006. URL: https://doi.org/10.1145/3340531.3412778. doi:10.1145/3340531.3412778.

[48] W. X. Zhao, S. Mu, Y. Hou, Z. Lin, Y. Chen, X. Pan, K. Li, Y. Lu, H. Wang, C. Tian, Y. Min, Z. Feng, X. Fan, X. Chen, P. Wang, W. Ji, Y. Li, X. Wang, J.-R. Wen, RecBole: Towards a unified, comprehensive and efficient framework for recommendation algorithms, in: Proceedings of the 30th ACM International Conference on Information and Knowledge Management, Association for Computing Machinery, 2021, pp. 4653–4664. URL: https://doi.org/10.1145/3459637.3482016. doi:10.1145/3459637.3482016.

[49] L. Michiels, R. Verachtert, B. Goethals, RecPack: An(other) experimentation toolkit for top-N recommendation using implicit feedback data, in: Proceedings of the 16th ACM Conference on Recommender Systems, Association for Computing Machinery, 2022, pp. 648–651. URL: https://doi.org/10.1145/3523227.3551472. doi:10.1145/3523227.3551472.

[50] L. Wegmeth, M. Baumgart, P. Meister, B. Gipp, J. Beel, OmniRec: The all-in-one solution for reproducible and interoperable recommender systems experimentation, in: Advances in Information Retrieval, Springer Nature Switzerland, 2026, pp. 129–135. URL: https://doi.org/10.1007/978-3-032-21321-1_18. doi:10.1007/978-3-032-21321-1_18.

[51] T. Vente, L. Wegmeth, A. Said, J. Beel, From clicks to carbon: The environmental toll of recommender systems, in: Proceedings of the 18th ACM Conference on Recommender Systems, Association for Computing Machinery, 2024, pp. 580–590. URL: https://doi.org/10.1145/3640457.3688074. doi:10.1145/3640457.3688074.

[52] J. Beel, A. Said, T. Vente, L. Wegmeth, Green recommender systems: A call for attention, ACM SIGIR Forum 58 (2024) 1–5. URL: https://doi.org/10.1145/3722449.3722468. doi:10.1145/3722449.3722468.

[53] L. Wegmeth, T. Vente, A. Said, J. Beel, Green recommender systems: Understanding and minimizing the carbon footprint of AI-powered personalization, ACM Transactions on Recommender Systems 4 (2026) 1–38. URL: https://doi.org/10.1145/3768626. doi:10.1145/3768626.

[54] J. Beel, L. Wegmeth, L. Michiels, S. Schulz, Informed dataset selection with "algorithm performance spaces", in: Proceedings of the 18th ACM Conference on Recommender Systems, Association for Computing Machinery, 2024, pp. 1085–1090. URL: https://doi.org/10.1145/3640457.3691704. doi:10.1145/3640457.3691704.

[55] M. Ferrari Dacrema, P. Cremonesi, D. Jannach, Are we really making much progress? a worrying analysis of recent neural recommendation approaches, in: Proceedings of the 13th ACM Conference on Recommender Systems, Association for Computing Machinery, 2019, pp. 101–109. URL: https://doi.org/10.1145/3298689.3347058. doi:10.1145/3298689.3347058.