

6.5% of the Neuro-Symbolic Literature Can Be Reproduced from Its Published Artifacts, a Six-Stage Audit Framework and First Instantiation

Brandon Colelough*, Vladimir Martirosyan, Ishan Tamrakar, William Regli, Aditya Kumar, Anh N. Nhu, Dhruv Dubey, Raj Ambavane, Haowei Deng

Department of Computer Science, University of Maryland
College Park, Maryland, USA
brandcol@umd.edu

Abstract

We present a six-stage framework for auditing the reproducibility of scientific claims across a research literature within the computer science domain, and instantiate our framework for the neuro-symbolic AI (NSAI) subdomain. Instantiating the framework on the NSAI subdomain produced a multi-year audit. Stage one retrieved 5,497 records and removed 3,018 duplicates. Stage two screened the 2,479 unique records at title and abstract, identifying 1,365 self-identified NSAI records, then removed a further 61 at full text for off-topic, non-research, no-quantitative-evaluation, or inaccessible-full-text reasons. Stage three sought a verifiable public code artifact for each of the 1,304 eligible records and found none for 849, leaving 455 to enter the artifact inventory and bounded rerun of stages four and five. We fully or partially reproduced 85 studies, 6.52% of the eligible corpus and 18.68% of attempted reruns. We found that 321 attempted reruns were blocked by missing non-code artifacts and 42 by missing or unusable code repositories. These figures quantify a persistent reproducibility deficit that survives even nominal “code available” declarations, and signal the need for enforced, versioned, and permanently archived artifact bundles in future NSAI publications. We argue that empirical NSAI papers should be required at submission time to provide complete, versioned, and permanently archived artifact bundles.

Introduction

Can an independent team rerun the computational pipeline released with a published paper and recover its main reported result? We put that question to 455 neuro-symbolic studies that advertised a public code artifact. Eighty-five reruns succeeded. In this paper, we undertake a study of the same-artifact rerun of published results in the emerging field of Neuro-Symbolic Artificial Intelligence. Specifically, we ask whether an independent team can rerun the computational pipeline released with a paper and recover the main reported result within a prespecified tolerance. NSAI is viewed by many as the emerging frontier that merges concepts from connectionist methods with those referred to informally as “Good Old Fashioned AI”, consisting of logic and formal techniques for reasoning. Integrations range from loose pipelines, where a neural component feeds a symbolic one, to tightly coupled systems trained jointly. The frontier science that aims

to integrate these two techniques is highly dynamic, and while the number of papers that claim to be creating contributions in this area is exploding in number, this paper reports that the results in the majority of these papers cannot be recovered by rerunning the released artifacts. This indicates that the research area comprising “Neuro-Symbolic AI” is experiencing a crisis of replication and lacks the community norms that would make released artifacts reliably executable as part of the publication of work. For the field to advance into the position of critical importance that many feel is inevitable, we should raise the bar on our scientific standards and require greater emphasis on how we can document reproducibility so that others may verify our claims. On the evidence below, we argue that empirical submissions in this field should carry a complete, versioned, and permanently archived artifact bundle at submission time.

Background

As has been documented often in the recent literature, many areas of science are currently experiencing a “reproducibility crisis” (Baker 2016) as studies have proven difficult or impossible to replicate (López-Nicolás et al. 2022), data sources are not available (Miłkowski, Hensel, and Hohol 2018), and experimental assumptions are not made explicit (Hensel 2020). Social sciences, in which these phenomena have been documented most extensively, have the additional issues posed by post hoc redesign of scientific hypotheses. Known as “Hypothesizing After the Results are Known”, or “p-hacking”, this occurs when a researcher forms or rewrites a hypothesis after seeing the data, and then presents that hypothesis as if it were specified before the data were collected (Rubin 2022). The field of computing, in theory, should be highly reproducible, as algorithms, code, data, and other artifacts can be easily shared and adopted. Conferences and journals have begun to require data sharing and other best practices to improve reproducibility (noa 2020). Without reproducible artifacts, computational research risks becoming unfalsifiable claims that cannot be independently verified and thus fall outside the bounds of science. We need to be able to rigorously and independently test hypotheses laid out in papers in the computer science community in the same way we do in other domains, so that the work can have more credibility.

*Corresponding author.

Contributions and Research Questions

We follow the ACM badging terminology (Association for Computing Machinery 2020) for reproducibility and refer to reproducibility as meaning that a team other than the original obtains a consistent result using the original team’s artifacts. We make two contributions. The first is a six-stage framework for auditing same-artifact rerun reproducibility across a literature. The second is an instantiation of the audit framework itself, the largest of its kind, employing the framework on 1,304 eligible NSAI records. The research questions we aim to address include: **RQ1** What proportion of NSAI papers releasing code can be fully or partially reproduced? **RQ2** How does artifact completeness (code / data / model weights) affect the probability of successful rerun? **RQ3** Do reproduction outcomes vary systematically by publication year or venue family (conference, journal, preprint)?

Related Work

Reproducibility failures are well-documented across empirical disciplines. Landmark studies in psychology and medicine have shown that a substantial fraction of published findings cannot be independently replicated (Open Science Collaboration 2015; Ioannidis 2005), and large-scale surveys across the social and biomedical sciences trace these failures to non-disclosed analytical choices, unavailable data, and selective reporting (Baker 2016; López-Nicolás et al. 2022; Hensel 2020). The interested reader is referred to that broader literature for a full treatment. Computer science presents a structurally different case, as algorithms, code, and data can in principle be shared exactly, meaning empirical claims ought to be among the most verifiable in science. In practice, however, the same failure modes recur, as the studies below illustrate. Vanderdonckt and Vatavu introduced *Amplitum*, a contextual framework that augments generic replication taxonomies with explicit descriptors of participants, devices, and physical settings (Vanderdonckt and Vatavu 2025). Their gesture-elicitation case study reproduced prior findings only after replicators matched the original laboratory environment and user cohort, underscoring that code and stimuli alone are insufficient when human behavior is the main dependent variable. Ferrari Dacrema *et al.* inspected 26 “state-of-the-art” (for the time) neural recommender papers (2015–2018) and were able to fully reproduce only 12 of the original 26, and in doing so discovered that *eleven of those twelve* fell behind well-tuned neighborhood, matrix-factorization, or sparse-linear baselines once a common experimental protocol was enforced (Ferrari Dacrema et al. 2021). Ferrari et al. attribute the apparent performance gains to methodological weaknesses, including baselines left at default settings, inadvertent test-set leakage during epoch selection, and undocumented data-split choices. The ReproNLP shared task series asks independent teams to repeat published human evaluations using materials supplied by the original authors, and reports that agreement with the original figures is frequently not achieved (Belz and Thomson 2023). The organisers attribute divergence to underspecified evaluation protocols and to details of the original setup that were never documented. Henderson *et al.* evaluated several widely cited DRL algorithms and found that benchmark rankings could reverse when random seeds, hardware platforms, or

training horizons were varied (Henderson et al. 2018). Their study therefore recommends reporting results over many seeds, applying formal significance tests, and disclosing every experimental detail. Pawlik *et al.* examined the longevity of public datasets and found that many links degrade, move, or silently change versions, rendering later experiments irreproducible despite nominal accessibility (Pawlik et al. 2019). They argue for immutable storage, rigorous version identifiers, and provenance metadata. Reproduction breaks down whenever any piece of the experimental context is missing, and Neuro-Symbolic AI is no exception. An open-code link without hardware details, solver commits, pre-processing scripts, dataset splits, and hyperparameter schedules offers little more than performative compliance. Gains attributed to the symbolic–neural fusion may vanish once baselines receive equal tuning or a knowledge base is revised; stochastic variation in the neural component could dominate the symbolic layer as well. Reliable reproduction, therefore, requires immutable, versioned artifact bundles, multi-seed evaluation, and full provenance for pretrained weights, logic programs, and curated knowledge graphs.

Methodological Audit Framework

We present our six-stage framework for auditing same-artifact rerun reproducibility across a literature and a realised example of our six-stage reproducibility framework below. Stage 1 constructs and screens the corpus. Stage 2 confirms full-text eligibility. Stage 3 identifies the code artifact and records a six-item artifact inventory before any execution is attempted. Stage 4 builds the released environment and checks executable integrity. Stage 5 reruns the primary experiment under a bounded repair allowance and a stated fidelity criterion. Stage 6 extracts data and audits outcome labels.

Corpus Construction & Screening, Audit Protocol Stages 1-3

Stage 1 constructs the corpus. We began with a deliberately broad bibliographic sweep, designed to capture the full breadth of the neuro-symbolic domain. Guided by a PRESS-validated query (McGowan et al. 2016) centered on “neuro-symbolic OR NeSy OR NSAI”, we queried nine major digital libraries on 23 May 2025: WEB OF SCIENCE, SCOPUS, PUBMED, EI COMPENDEX, IEEE XPLORE, ACM DL, SPRINGERLINK, GOOGLE SCHOLAR, and ARXIV. The query returned 5,497 records. This broad sweep increased coverage of emerging or lexically idiosyncratic work that narrower queries often overlook. A de-duplication pipeline removed 3,018 duplicates (55% of raw hits), leaving 2,479 unique records. The substantial overlap across sources illustrates the cross-posting norm in NSAI research and underscores the need for multi-pass de-duplication.¹ Relevance screening was performed in a single-blind title-and-abstract pass. After a 20-record calibration, reviewers screened titles and abstracts and retained any paper whose authors explicitly described the work as “neuro-symbolic,” regardless of application domain. 1,365 papers satisfied this broad criterion. Code availability

¹Pipeline tools: EndNote, Covidence, Zotero, SR-Accelerator, and Rayyan.

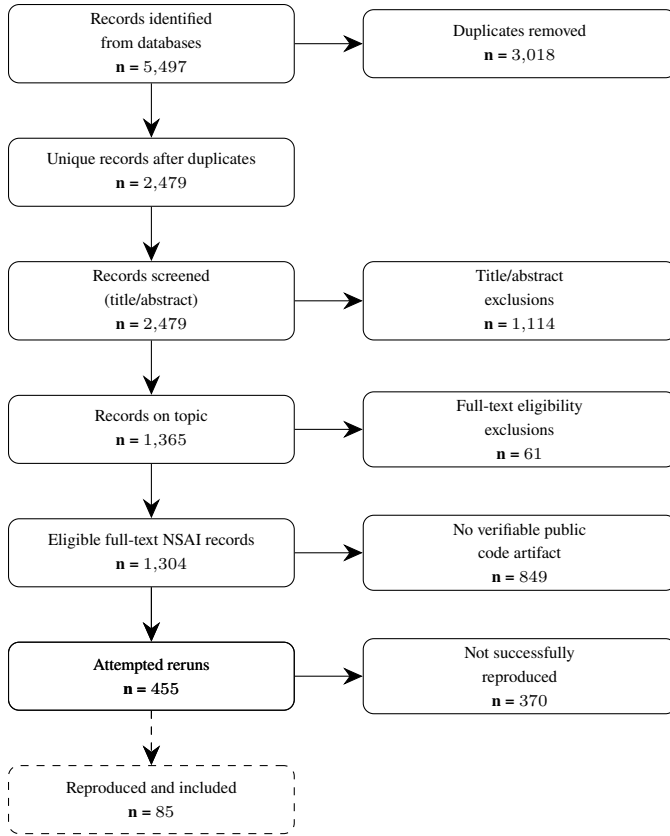


Figure 1: Selection and eligibility flow for the NSAI audit. Late full-text eligibility exclusions are shown separately and are not counted as rerun outcomes. The dashed bottom box reports the number of successful reruns within the attempted set and is shown as an audit outcome.

was not inferred from the abstract alone. Instead, repository verification was conducted during full-text eligibility assessment wherein annotators first inspected the full paper for repository or artifact links, and when none were present, they performed a structured external search using the paper title, author names, and method or domain keywords. This process eliminated 849 records for which no verifiable public code artifact could be identified and left 455 code-bearing studies for the rerun audit, an attrition of 65.11 % that exposes the gap between open-science claims and delivered artifacts. The stages reduced the literature from 2,479 unique records to 1,365 NSAI-relevant papers and further to 455 audit candidates that at minimum linked or could be matched to a code artifact. Figure 1 visualizes each reduction step, linking corpus construction directly to our objective of quantifying verifiable progress in Neuro-Symbolic AI.

Audit Protocol Stage 4-6

Inclusion criteria A paper entered the reproduction pipeline only when it satisfied *all* of the following; IC_1 - **Neuro-symbolic integration** – the study is self-described as Neuro-Symbolic. IC_2 - **Empirical evaluation** – the paper reports quantitative results on benchmarks, real-world

datasets, or synthetic tasks and compares against baselines or ablations. IC_3 - **CS relevance** – the work contributes technical insight into the CS domain. IC_4 - **Auditable code claim** – the paper provides a direct repository link or a uniquely identifiable public code artifact for the reported system, sufficient to permit artifact audit. IC_5 - **Full text Available** – the paper has an accessible full text for audit.

Exclusion criteria Papers were removed if they violated *any* of the following: E_1 Not written in English. E_2 Literature review, review, survey, editorial, or otherwise not original empirical research. E_3 No verifiable public repository or archival code artifact for the reported system could be identified from the paper or via a structured external search. E_4 Missing indispensable artifacts required to rerun the primary experiment could not be identified, accessed, or reconstructed under the study protocol. E_4 applies where the paper or its linked repository shows an indispensable artifact to be unavailable before any rerun is attempted and absences discovered during an attempted rerun are recorded as O_4 . E_5 Lacks quantitative evaluation. E_6 Outside the scope of neuro-symbolic methods. E_7 Duplicate, superseded, or version-of-record already retained. E_8 No full-text access (pay-walled or retracted). Exclusions E_1 – E_8 are applied only at title-and-abstract screening and at full-text eligibility assessment. A record failing any of them never enters the attempted rerun pool. Once a record does enter, every subsequent failure is recorded as an audit outcome under O_1 – O_5 and never as an exclusion, so that sample selection is never confounded with audit results. Figure 1 reports counts for both.

Reproduction Procedure Post-entry rerun outcomes are reported in Figure 2. Every study with a publicly accessible repository was evaluated under the six-stage protocol that compressed the workflow into discrete, auditable checkpoints². All annotators began with a one-hour onboarding workshop³ that introduced the audit workflow and extraction template. This was followed by two 4-hour live sessions (covering environment builds and dependency management, metric verification, and licensing constraints) and a 3-hour supervised drop-in lab in which each participant could reproduce exemplar studies end-to-end in a supervised environment. A worked example demonstrating application of the study’s data-extraction form is also available.⁴ All rerun attempts and artifact assessments were conducted by a team of eight trained graduate student annotators over a nine-month period, who completed the study’s standardized onboarding workshop and calibration protocol. Each annotator handled an average of 64 records, comprising the 455 attempted reruns and the 61 late eligibility exclusions distributed across eight annotators. Workload was allocated randomly across the team. A ten-paper calibration pilot produced Cohen’s $\kappa = 0.82$ (a standard measure of inter-rater agreement, where 1.0 is perfect agreement and values above 0.80 are considered strong), and outstanding disagreements were reconciled in group discus-

²NSAI survey and audit index

³annotator training and calibration materials in the [\[https://doi.org/10.5281/zenodo.21779234\]](https://doi.org/10.5281/zenodo.21779234)

⁴NSAI-2025 data-extraction worked example

sion. During the nine-month audit phase, the team met every second week to review edge cases and realign on the reproduction protocol. Outcome labels were assigned after entry into the attempted rerun pool as follows: **O1) Fully reproduced.** The primary experiment was executed successfully, and the reproduced result satisfied the study’s fidelity criterion. **O2) Partially reproduced.** The core pipeline executed, and the paper’s main qualitative claim was preserved, but one or more quantitative results fell outside the acceptance band, or only a subset of the headline experiments could be rerun. **O3) Executed but did not reproduce within tolerance.** The system ran to completion, but the reproduced results materially exceeded the acceptance threshold or contradicted the paper’s main quantitative claim. **O4) Not executable due to missing or inaccessible artifacts.** The attempted rerun could not proceed because one or more indispensable artifacts were unavailable, inaccessible, or not reconstructable under the study protocol. **O5) Not executable due to environment or code failure.** Attempted rerun failed because the released code or environment could not be built or executed under the protocol despite the permitted minimal fixes. For this study, an artifact was classified as missing only when it was an indispensable input to the primary experiment and could not be reconstructed from the paper and released materials under the study protocol. Indispensable inputs included fixed datasets or splits, preprocessing scripts that alter data semantics, checkpoints or weights when evaluation depended on a fixed trained model state, rule sets or knowledge bases, configuration files, environment descriptors, and evaluation assets. All reproduction logs and extracted data are version-controlled, and a consolidated record is available in the public reproduction and data-extraction sheet.

Evaluation Design

We report two outcomes and keep them separate. Full reproduction (O1) is the primary result and requires recovering the paper’s headline number. Partial recovery (O2) is reported alongside it. Only papers that entered the attempted rerun pool count toward either. To qualify as a full reproduction, a repository must execute under the author-supplied environment (or a minimally updated equivalent), with a max 7 day wall-clock time and on the audit team’s GPU pool, and reproduce the paper’s primary metric to within $\pm 5\%$ absolute error (or inside the authors’ 95 % confidence interval), yield outputs consistent with the paper’s headline claims, and require no correction of bugs intrinsic to the model architecture. Annotators were authorized to (i) update deprecated package versions or apply path fixes, provided it is minimal, (ii) adjust file paths, (iii) supply a lightweight evaluation harness when none was provided, and (iv) patch minor scripting errors. Malformed or undocumented environments, missing indispensable post-entry artifacts, or architecture-level defects were recorded as O4 or O5 audit outcomes, not as exclusions.

Protocol Stages

This study was designed around six central stages, which included:

Stage 1 – Full-text eligibility confirmation Using the eligibility criteria above and the reproduction component of the data extraction Form, annotators re-checked language, topical relevance, quantitative evaluation, and full-text accessibility. Failure on any item triggered eligibility exclusion.

Stage 2 – Repository identification and full-artifact verification For papers passing Stage 1, annotators identified the code artifact from the paper itself or via structured external search, recorded the repository URL, license, and commit hash where available, and then verified whether all artifacts required for reproduction (e.g., code, data, weights, environment files, and documentation) were publicly accessible. If an essential artifact was available but required gated access (e.g., proprietary data), up to two e-mail requests were sent over 14 days, and a lack of response led to a missing-artifact access outcome.

Stage 3 – Environment Build & Executable Integrity. Repositories passing Stage 2 were rebuilt with the authors’ environment file(s) (e.g., Dockerfile, `requirements.txt`, `environment.yml`, etc.). Builds failing due to irreconcilable dependencies were recorded as O5 outcomes. Successful builds proceeded to unit or smoke tests, and fatal code-level errors were likewise recorded as O5 outcomes.

Stage 4 – Result Re-execution. The primary experiment was rerun. A study was marked *Accepted* if reproduced metrics were faithful to the reported results from the authors and reported on a 5-point Likert scale of *Fully Reproducible*, *Partial*, *Not reproduced (large deviation / failed run)*, and *Unable to attempt/pending* whilst also reporting the exact percentage gap between reported and reproduced results.

Stage 5 – Data Extraction & Audit. Papers assigned O1 or O2 triggered data extraction utilizing Sections 4–8 of the reproduction and data extraction Form, capturing neural/symbolic design details, datasets, compute budget, metrics, and evaluation protocol. Upon completion of the reproduction work, one annotator later reviewed 100 % of O1/O2 papers and a stratified sample of O3/O4/O5 papers.

Technical Infrastructure, Limitations, and Data Management

All reproductions were executed on the University of Maryland Zaran high-performance computing cluster under its default quality-of-service limits. Jobs without GPUs ran in the standard partition on up to a single 128-core Zen 3 node with 512 GiB RAM (7-day wall-clock cap). GPU jobs used the GPU partition, each limited to one full GPU node consisting of either 4 × NVIDIA A100 (40 GiB each, 128 Zen 3 CPU cores, 512 GiB RAM) or 4 × NVIDIA H100 (80 GiB each, 96 Sapphire Rapids cores, 512 GiB RAM), also with a 7-day limit. Software stacks were loaded per job via the cluster’s module system. Author-supplied Dockerfiles were converted to Apptainer images using `docker2singularity`, and when conversion failed, the build was completed on a local machine.

Limitations The primary construct of this paper is reproducibility of reruns of same-artifacts to determine whether an independent team can obtain the reported result using the artifacts released by the authors. Under this construct, trained

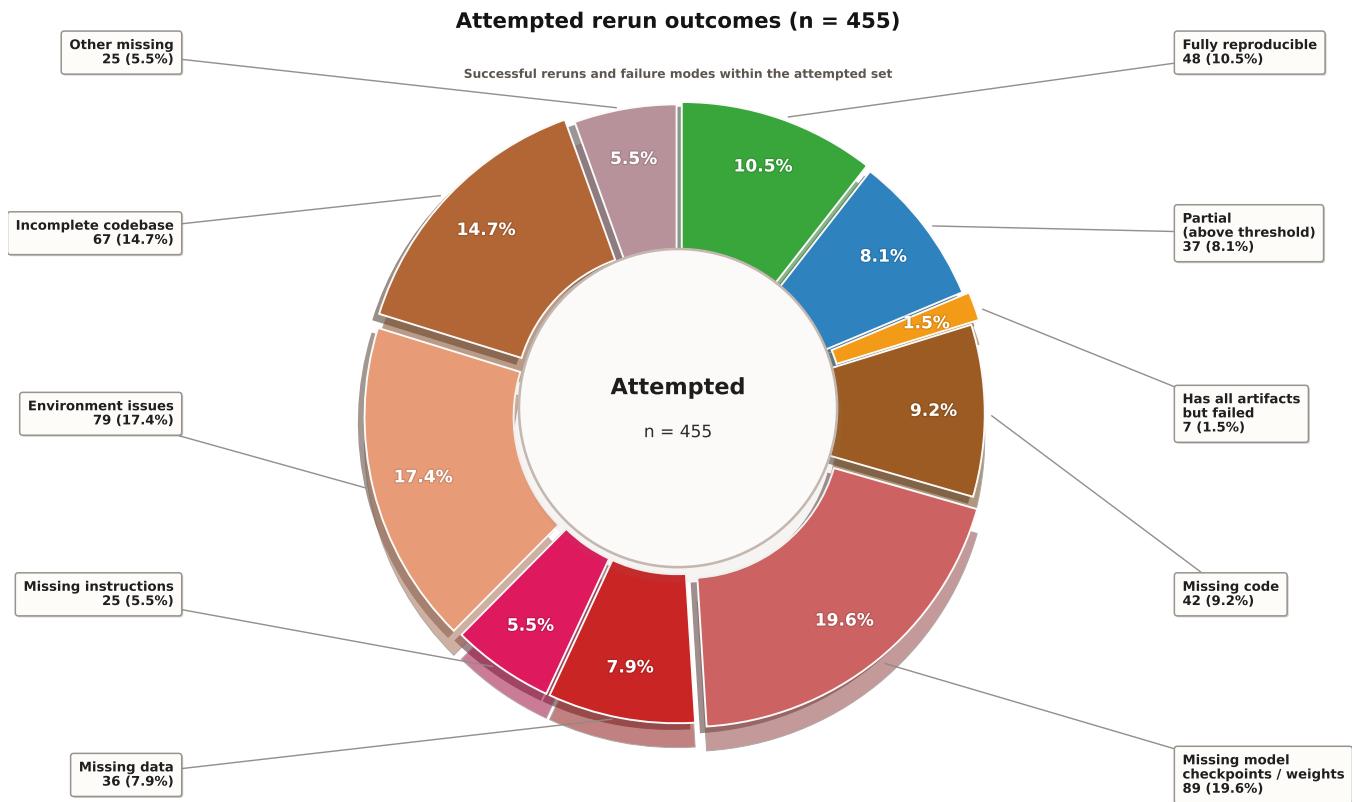


Figure 2: Rerun outcomes for the attempted same-artifact audit set ($n = 455$).

checkpoints were treated as indispensable when the published evaluation depended on a fixed trained model state. We did not substitute retraining in the primary analysis because retraining changes the experimental object and introduces additional stochastic, software, and platform variance. However, some papers without released checkpoints may remain reproducible in principle via retraining; we therefore interpret missing-checkpoint cases as non-rerunnable under this protocol rather than as proof that the underlying method is scientifically irreproducible in every broader sense. This framing is consistent with ACM’s current artifact terminology, under which reproducibility means an independent group obtaining the same result using the authors’ own artifacts, with agreement judged within an acceptable tolerance rather than exact identity.⁵ Additionally, although annotators could patch environment files and author evaluation harnesses when absent from the original release, severe lack of documentation for code and environments, together with architecture-level defects intrinsic to the framework itself (for example, a fresh clone that crashes with tensor shape mismatches or NaN losses despite the original environment), remained failure conditions under the protocol.

Data Management & Transparency All repositories, weights, patched scripts, environment manifests, and execution logs required for reproduction of examined manuscripts

⁵<https://www.acm.org/publications/policies/artifact-review-and-badging-current>

remain on the University of Maryland Zaran project scratch file system under institutional retention policy. Summary statistics are mirrored in the public extraction sheet⁶. The complete audit trail for this systematic reproduction review (including per-paper reproduction logs, annotator tagging logs, and the reproduction protocol and supporting documentation) is archived in a long-term online repository⁷, together with the scripts required to regenerate every reported aggregate and figure. We additionally publish the raw outcomes via an interactive webpage⁸ to enable paper-level inspection and independent recomputation of the headline counts reported in the manuscript.

Results

System-Level Reproducibility

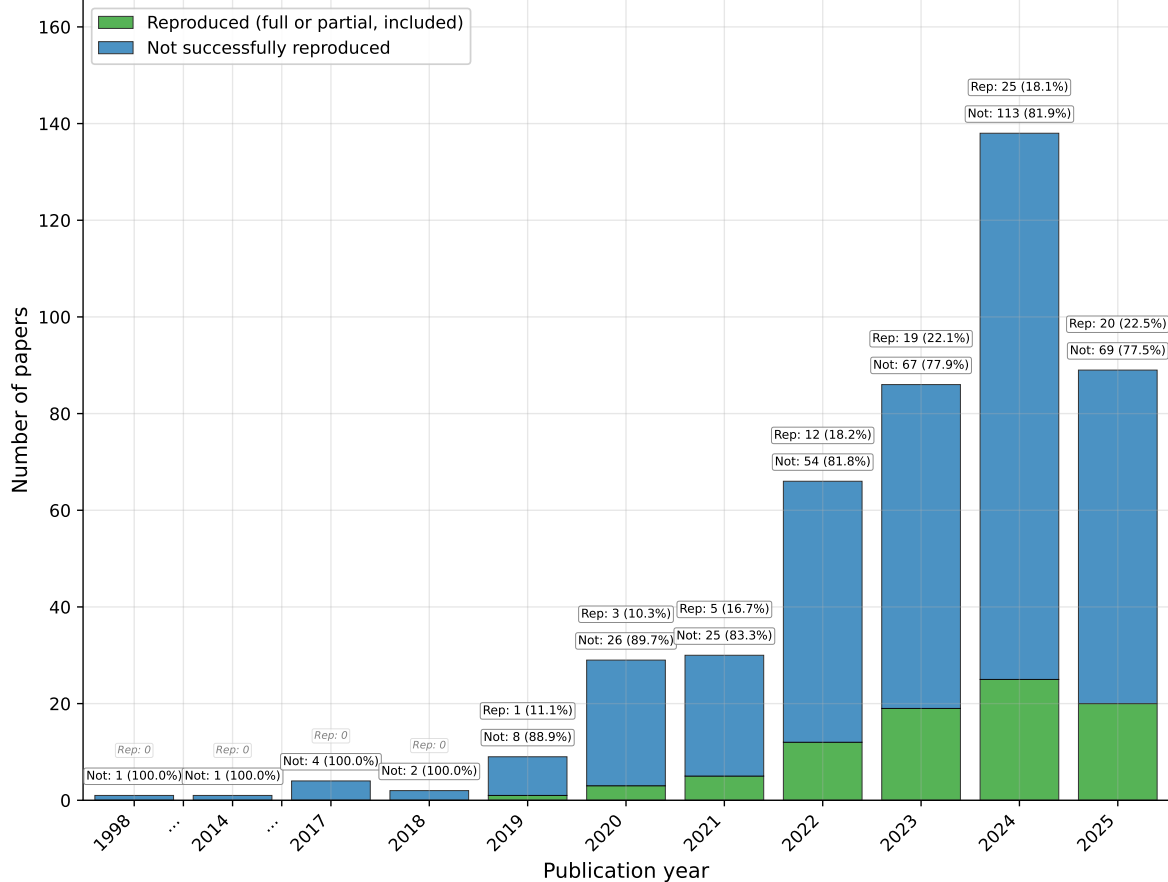
Figure 2 reports outcomes for the attempted rerun set only ($n = 455$). Within this set, **48** studies (10.55 %) were fully reproduced and **37** (8.13 %) were partially reproduced, yielding **85** successful reruns overall (18.68 %). Failures within the attempted set comprised **321** papers (70.55 %) blocked by missing non-code artifacts, **42** (9.23 %) blocked by unavailable or unusable code repositories, and **7** (1.54 %) that had the nominal artifact set but failed because of environment or

⁶NSAI reproduction and data-extraction results

⁷[\[https://doi.org/10.5281/zenodo.21779234\]](https://doi.org/10.5281/zenodo.21779234)

⁸interactive results browser

Attempted Rerun Outcomes by Publication Year (Reproduced & Included vs Not Successfully Reproduced)



Timeline not to scale

Figure 3: Reproduction outcomes over time for the attempted NSAI papers. Stacked bars show, for each publication year, the number of papers that were successfully reproduced (full or partial) versus not reproduced under the study protocol, illustrating the growth of the field alongside a reproduction rate that shows no sustained upward trend.

code defects. The **61** records that failed late full-text eligibility checks are reported separately in Figure 1 and are not counted as rerun outcomes. The rerun success rate is therefore $85/455 = 18.68\%$ (Wilson 95% CI 15.4–22.5). Relative to the eligible corpus ($n = 1,304$), the rate is $85/1304 = 6.52\%$ (Wilson 95% CI 5.3–8.0). The primary obstacle to successful same-artifact reruns was incomplete artifact availability. Within the attempted set, **321** papers were blocked by missing non-code artifacts, and **42** by missing or unusable code repositories. A residual **7** studies shared the nominal artifact set yet still failed because of environment or code defects. By contrast, when code, data, and weights were all available, reruns succeeded in **85** of **92** cases (92.4 %, Wilson 95% CI 85.1–96.3). Missing artifacts concentrated in three components. Absent model checkpoints or weights blocked **89** attempts (19.56 % of attempted reruns), environment or specification problems blocked **79** (17.36 %), and incomplete codebases blocked **67** (14.73 %). These three categories account for **235** of **321** missing-artifact failures (73.2 %), leaving **86** failures distributed across missing datasets (**36**, 7.91 %), missing or

incomplete documentation (**25**, 5.49 %), and other missing items (**25**, 5.49 %). The skew implies that most failures arose from predictable artifact-release gaps.

Year-Wise Reproducibility Patterns

Figure 3 tracks NSAI reproduction outcomes by publication year and shows a steep rise in paper volume, from single digits before 2019 to 138 publications in 2024 and 89 in the partial-year 2025 cohort. From 2019 onward the annual full-or-partial rate ranges from 10.3 % (2020) to 22.1 % (2023), with intermediate values of 11.1 % (2019), 16.7 % (2021), 18.2 % (2022), and 18.1 % (2024). Because the 2025 cohort covers only January–May, its denominator is still evolving; we therefore treat the 22.5 % figure as provisional and exclude 2025 when describing the trend. The years before 2019 contributed 8 attempted reruns in total, none of which were reproduced, and we exclude them as well. Within the 2019 to 2024 window, the proportion of studies reproduced (fully or partially) remains in a narrow band of roughly 10 % to 22 % of the yearly output, and the stacked-bar profiles reveal

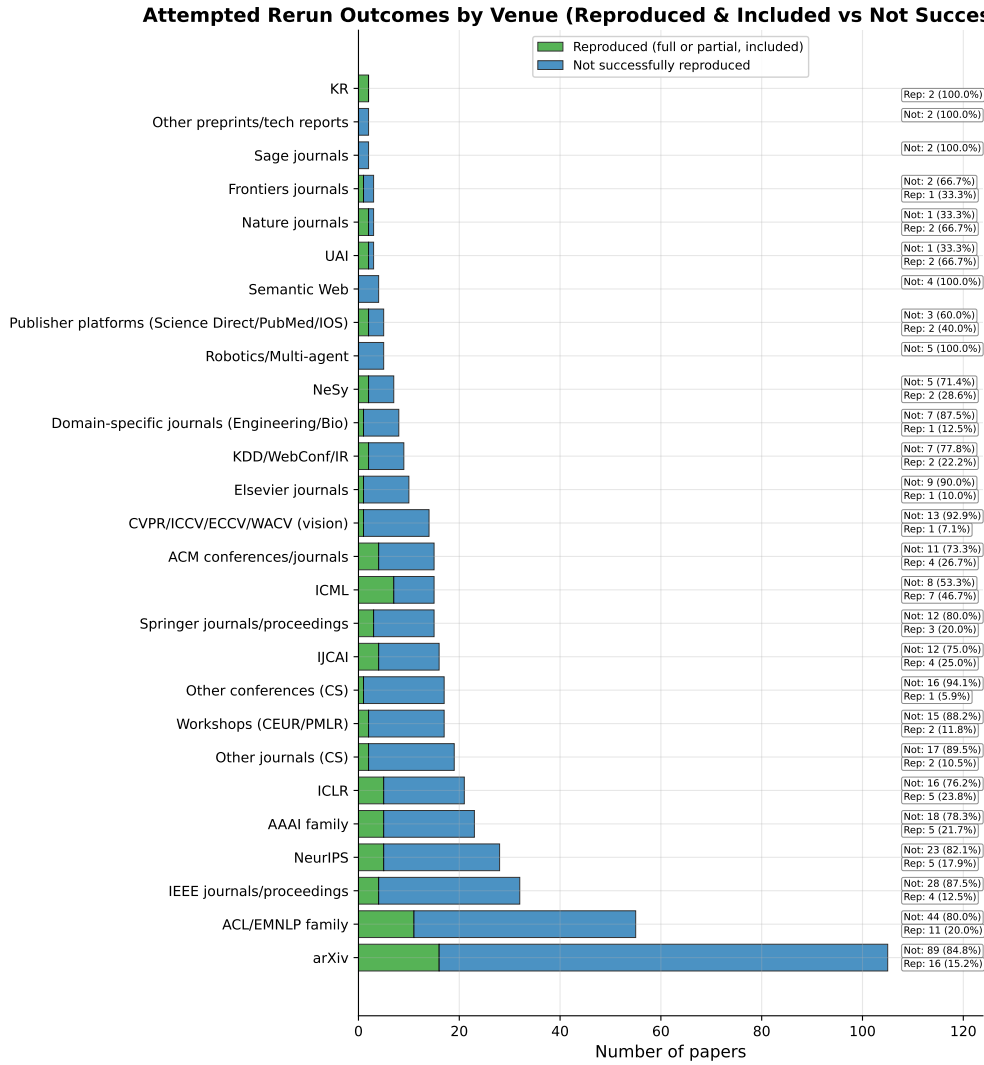


Figure 4: Reproduction outcomes by venue group for the attempted NSAI papers. Stacked bars show, for each venue family, the number of papers that were successfully reproduced (full or partial) versus not reproduced. Venues are ordered by number of attempted reruns.

no sustained upward trajectory. In practical terms, the field is publishing substantially more work each year, but the likelihood that any given paper can be reproduced under our protocol has not materially improved, suggesting that the surge in publications has not, by itself, improved real-world reproducibility under our protocol.

Venue-Level Impacts on Reproducibility

Figure 4 compares reproduction outcomes across twenty-seven venue families and shows no detectable advantage for journals, conferences, or preprint servers at the class level ($\chi^2 = 2.26$, $df = 2$, $p = .32$). Large outlets illustrate this point as arXiv achieves **16/105** successes (15.2%), ACL/EMNLP conference proceedings **11/55** (20.0%), IEEE journals **4/32** (12.5%), and NeurIPS **5/28** (17.9%). Comparable rates are observed for AAAI family journals/conferences

(21.7%) and for ICLR (23.8%) despite differing review models, while domain-specific journals, vision conferences, and publisher platforms all cluster in the low-to-mid-teens. The only apparent outlier is ICML at **7/15** (46.7%) however, its small denominator limits generality. Aggregating by broad class, conferences reproduce at 21.1% (53/251, Wilson 95% CI 16.5–26.6), journals at 16.5% (16/97, CI 10.4–25.1), and preprints at 15.0% (16/107, CI 9.4–22.9). The three intervals overlap substantially. Publication venue alone is not a reliable predictor of NSAI reproducibility. Artifact completeness, not outlet type, is what distinguishes the attempts that succeeded from those that did not.

Failure modes behind missing artifacts

Figure 5 assigns each of the $n = 363$ papers that were non-reproducible due to incomplete artifact provision to a

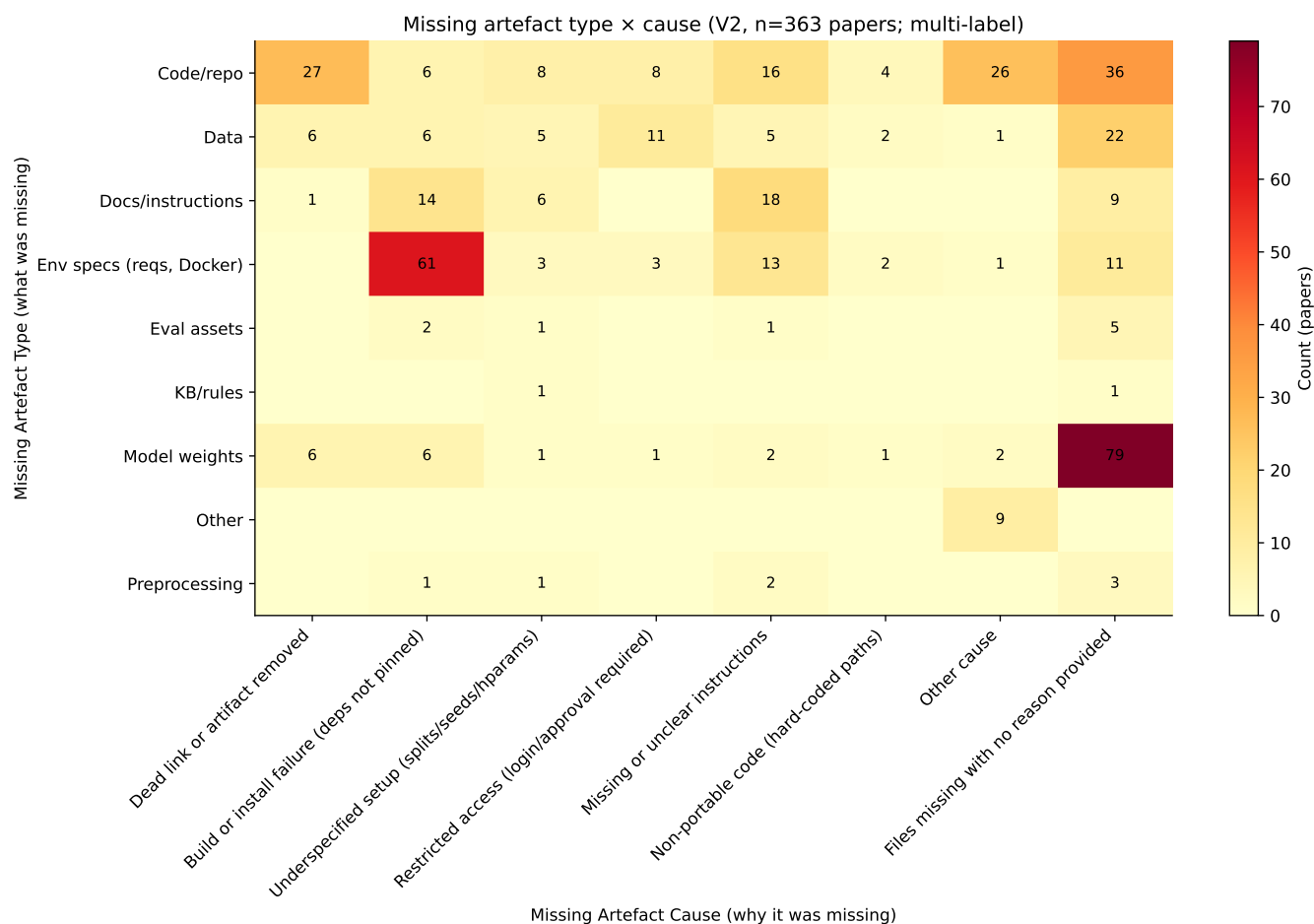


Figure 5: Rows indicate *what* was missing (e.g., data, model weights, environment specifications, documentation), while columns indicate *why* it was missing (e.g., dead links, restricted access, an under-specified setup, or files absent with no explanation)

missing artifact type and a primary cause. The largest cell is missing model weights with files missing with no reason provided (79 papers), and the second is missing environment specifications linked to build or install failure from unpinned dependencies (61 papers). For model weights, restricted access and dead links are rare (1 and 6 papers, respectively), so the bottleneck is usually not permissioning but non-release or decay, and the claim that a repository enables rerunning the reported pipeline is falsified by inspection in the majority of weight-missing cases. Environment failures concentrate on dependency drift, which is consistent with repositories that omit a locked descriptor or ship one that no longer resolves under current tooling. Code and repository issues are also common, with files missing with no reason provided (36) and dead links or removed artifacts (27), plus a substantial remainder attributed to other causes (26), so “code available” is an unreliable proxy for an executable codebase. Data failures split between restricted access (11) and unexplained absence (22), while documentation failures cluster in missing or unclear instructions (18) and build failures (14).

Citation impact vs reproducibility

Figure 6 demonstrates whether citation impact is a useful proxy for practical reproducibility by comparing DOI-resolved citation counts across the same reproduction outcome and exclusion buckets shown in figures 2 and 5 above. Citation counts were obtained by DOI lookup for 427 of the 455 attempted records. The remaining 28 lacked a resolvable DOI and are excluded here. The distribution of citations from the remaining 427 papers, as shown in Figure 6, indicates substantial overlap across all buckets, with a large mass of zero and low citation papers in every outcome and a long right tail that inflates the average number of citations per bucket. The fully reproducible bucket has the highest mean citation count at 14.1 but a median of 2, and comparably high means occur in dominant non-reproducible buckets such as missing model checkpoints or weights with mean 10.0, median 1, max 234 and environment or specification issues with mean 10.5, median 2, max 233, showing that incomplete artifact release shows no clear relationship with number of citations that a paper may receive.

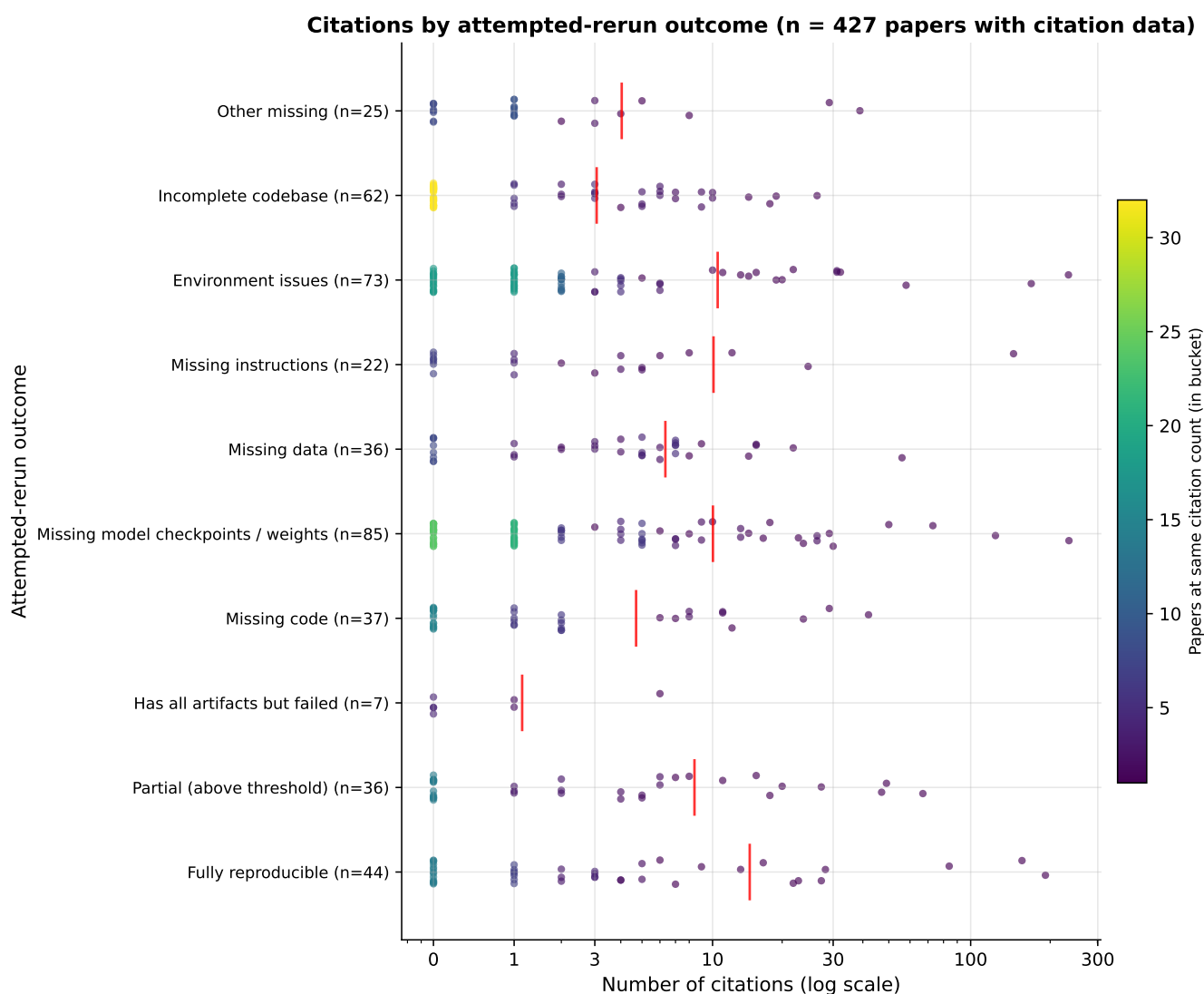


Figure 6: Citation counts by reproduction outcome/exclusion reason for NSAI papers with available citation data ($n = 427$) shown on a log scale. The red marker indicates the mean citation count for each bucket.

Answers to Research Questions RQ1 – RQ3

For **RQ1** [proportion of NSAI papers that can be reproduced], within the attempted rerun set ($n = 455$), 48 studies met the strict full-reproduction criterion (10.55 %, Wilson 95% CI 8.1–13.7) and a further 37 met the partial criterion (8.13 %). The full-or-partial composite is 85/455 (18.68 %). Relative to the full eligible corpus ($n = 1304$), these fall to 3.68% (full; Wilson 95% CI 2.8–4.8) and 6.52% (full or partial; Wilson 95% CI 5.3–8.0). For **RQ2** [effect of artifact completeness on reproduction success], artifact completeness is the dominant factor. When code, data, and weights were all present, reproduction succeeded in 85 of 92 cases (92.4%). When artifacts were incomplete, 321 of 455 reruns (70.55%) were blocked, with missing model weights ($n = 89$), environment issues ($n = 79$), and incomplete codebases ($n = 67$) accounting for 73.2% of those failures. For **RQ3** [variation in outcomes by

publication year and venue family], neither publication year nor venue predicts reproducibility. Success rates remained flat between 10.3% and 22.1% from 2019 to 2024 despite rapid growth in paper volume, and large outlets cluster in a similar narrow band, with arXiv at 15.2%, ACL/EMNLP at 20.0%, IEEE at 12.5%, and NeurIPS at 17.9%, and aggregate class rates of 21.1% for conferences, 16.5% for journals, and 15.0% for preprints ($\chi^2 = 2.26$, $df = 2$, $p = .32$)

Discussion

Summary of Principal Findings in Context

Every study in the attempted set ($n = 455$) carried a verifiable public code artifact at screening. Fewer than one in five could be rerun.

Trends and Patterns

Despite adopting an audit protocol that permits limited dependency updates, path corrections, and lightweight evaluation harnesses to be supplied by auditors, the proportion of NSAI papers that reproduced remained stubbornly flat, oscillating between 10.3 % and 22.1 % per year from 2019 through 2024. Figure 2 attributes most failures to missing non-code artifacts (70.55 %) and, to a lesser extent, to absent code (9.23 %), yet when the full range of artifacts required for reproduction is present, reproduction succeeds in 92.4 % of cases. Perhaps most striking is the insignificance of venues as a determinant for whether a manuscript will provide the resources required to reproduce the work. Conferences, journals, and preprints cluster at 21.1 %, 16.5 %, and 15.0 % respectively, and the three confidence intervals overlap (Figure 4). The evidence points to artifact availability, rather than venue type or methodological novelty, as the main bottleneck to reproducibility in current NSAI research. The analysis, therefore, shifts the conversation from improving experimental technique to enforcing comprehensive artifact release across all publication outlets.

“Dead” Codebase Links

Despite passing the initial NSAI relevance screen and subsequent repository-identification step, 42 of the in-scope papers ultimately fell into the Missing Code category because the referenced repository was dead, private, empty, unrelated, or otherwise unusable at audit time. In most instances, the hyperlink supplied in the manuscript was (i) dead or resolving to a private or removed repository, (ii) redirected to a project homepage that described the system but held no source files, (iii) pointed to a repository that never held any source code, or (iv) led to code that was unrelated or insufficient to implement the reported method. These false-positive disclosures illustrate that nominal compliance with “code available” guidelines is not enough, and that persistent, content-verified repositories and explicit version tags are essential if claims of openness are to translate into practical reproducibility.

Reproducibility requirements for the domain of Neuro-Symbolic AI as a science

Practical implications. Systematic examination of the AAAI Author Reproducibility Checklist⁹, the NeurIPS Paper-Checklist Guidelines¹⁰, the IJCAI Reproducibility Guidelines¹¹, the NeurIPS report (Pineau et al. 2021), and the AI Magazine survey of reproducibility barriers and drivers (Semmelrock et al. 2025) shows that the five documents converge on a common baseline of artifacts that an empirical paper must provide: (i) runnable source code; (ii) the exact datasets or immutable links to them; (iii) pre-trained checkpoints; (iv) a single command or script that reproduces the reported metrics (an evaluation harness) (v) a machine-readable environment file (`environment.yml`, `requirements.txt`, `Dockerfile`, etc.) and (vi) documentation in the form of instructions

that integrate items (i)–(v) into one executable workflow. All checklists emphasize persistent identifiers, explicit version locking, and permissive licensing.

Policy levers Existing venue initiatives fall into two categories. ACM artifact badges¹² include an external review of the repository, whereas the AAAI, NeurIPS, and IJCAI checklists are self-attested by the authors. Each submission to a peer-reviewed conference, journal, and even preprint venues should undergo an automated repository audit before peer review, confirming that all six artifacts are present and executable. Only 7 of 455 attempts failed with a complete artifact set, so such a check would address 363 of the 370 failures we observed.

Future directions for NSAI Research Neuro-Symbolic models integrate multiple components in complicated systems to deliver functional systems, so a single missing artifact can invalidate the entire pipeline. An absent checkpoint severs logic bindings, and undocumented rule sets could compromise evaluation and as such, authors must archive the complete artifact bundle (codebase, datasets, model weights, evaluation scripts, environment files and documentation) in a DOI-minting repository (e.g., Zenodo, HuggingFace-Hub, Figshare, OSF, or Mendeley Data).¹³ Computer science research is only science when its results withstand independent execution. A manuscript and a transient GitHub link do not satisfy this condition. Requiring the six-item artifact bundle, bound by immutable URIs for all publication venues (including pre-publication), is the most immediate, low-cost intervention available to the community. The good news is that the barrier to improvement is low. The included papers from this study demonstrate that full reproducibility is achievable at any publication venue when authors commit to a complete artifact bundle from the outset. The community can act now by depositing code, data, model weights, environment files, and a single-command evaluation script in a DOI-minting repository such as Zenodo or HuggingFace-Hub at submission time, not as an afterthought. Venues can reinforce this by requiring automated artifact checks before peer review begins, a low-cost intervention relative to the effort wasted reproducing or discarding non-reproducible work.

Case Studies

Case Study 1: Scallop-A Badge-Awarded Benchmark for Reproducibility The Scallop PLDI 2023 paper (Li, Huang, and Naik 2023) earned both the ACM *Artifacts Available* and *Artifacts Evaluated—Reusable* badges, and its release exemplifies full-stack reproducibility. A Zenodo snapshot (DOI 7804200) freezes the exact commit, dataset splits, pretrained weights, and SHA-256 hashes. Docker and Conda manifests plus a Rust nightly lockfile reconstitute the software stack, while a single script (`run_all.sh`) rebuilds, trains, and evaluates the eight-task benchmark. GitHub CI recompiles the core library, runs unit tests, and executes a

⁹<https://aaai.org/conference/aaai/aaai-26/reproducibility-checklist/>

¹⁰<https://neurips.cc/public/guides/PaperChecklist>

¹¹<https://www.ijcai.org/reproducibility>

¹²<https://www.acm.org/publications/policies/artifact-review-and-badging-current>

¹³These services issue persistent identifiers, maintain long-term storage, and provide metadata suitable for citation.

smoke benchmark on every push, preventing configuration drift. Our audit reproduced all eight tasks on a single A100 GPU with a median absolute deviation $\leq 3\%$ from the reported metrics and required no manual intervention. Scallop, therefore, is a fantastic demonstration that badge-level artifact curation and automated validation can turn a complex NSAI system into a reliably re-runnable research object.

Case Study 2: LogiCity-Conference-Level Reproducibility without Formal Badging Unlike Scallop, LogiCity (Li et al. 2024) earned no external badge, yet its NeurIPS 2024 Datasets and Benchmarks package replicated on the first attempt. The authors froze code, data, and checkpoints in a version-tagged release, shipped Docker and Conda manifests, and wired a one-command launcher that rebuilds the simulator, trains agents, and scores results. Continuous-integration runs the full smoke benchmark on every commit, and SHA-256 checks ensure downloads match the snapshot. On our A100 test node, the Safe-Path-Following and Visual-Action-Prediction tasks reproduced within ± 2 pp of the reported scores and retained the original baseline ordering. LogiCity therefore demonstrates that rigorous curation and automated checks can deliver badge-level reproducibility at a premier conference—even when no formal badging program exists.

Case Study 3: MARS-Preprint-Level Reproducibility through Curated artifacts The *Mechanism-of-Action Retrieval System* (MARS) (DeLong et al. 2024) appears only as an arXiv preprint posted in March 2024, yet its artifact package met every criterion for a gold standard reproducible package. A version-tagged GitHub repository provides Docker and Conda descriptors, immutable data archives, and pre-trained checkpoints, and a single `run.sh` script rebuilds the biomedical-graph pipeline, trains the models, and computes evaluation metrics. Re-execution on one NVIDIA A100 GPU reproduced the MoA-Net benchmark with MRR 0.315 versus the reported 0.318 and Hits@10 0.672 versus 0.685 (absolute deviation ≤ 2 percentage points), preserving the published ranking of baselines. All code, data, and weights are protected by SHA-256 hashes, permitting table verification without retraining. Although the release has not yet undergone an external artifact audit, this example indicates that a fully versioned, one-command artifact stack can deliver robust reproducibility irrespective of publication venue.

Cross-Case Comparison and Venue Implications The three case studies span the principal publication strata in computer science, as a badge-audited archival proceedings (Scallop), a top-tier conference track without external certification (LogiCity), and an open-access arXiv preprint (MARS), yet each enabled low-effort, reliable replication by adhering to the same practical principles. Each artifact package (i) freezes the software stack with Docker or Conda, (ii) archives immutable data and pre-trained checkpoints under permanent identifiers, (iii) offers a one-command script that rebuilds and evaluates the full pipeline, (iv) discloses every configuration parameter, and (v) offers a good level of documentation with instructions on environment build requirements. With these ingredients in place, our audit reproduced all reported metrics with an absolute error no greater than three percentage points

for all three frameworks. Scallop achieves this through a formal ACM badge review, LogiCity relies on conference-driven community norms and continuous-integration tests, and MARS depends on author-integrity. These results indicate that enforceable artifact standards, not venue prestige, are the decisive factor in achieving reliable reproducibility.

Conclusion

This study provides, to the best of our knowledge, the first longitudinal, large-scale assessment of same-artifact rerun reproducibility within neuro-symbolic AI. Across 1,365 self-identified NSAI records identified by our search, 1,304 met full-text eligibility and 455 entered the attempted same-artifact rerun audit and 85 were fully or partially reproduced. Failure analysis shows that reproducibility typically collapses when essential artifacts such as full codebases, datasets, pretrained model weights, evaluation harness scripts, environmental files or documentation are absent at release. Computer science research attains scientific legitimacy only when independent investigators can verify its empirical claims. To that end, we recommend that *all* empirical submissions, regardless of venue, be required to pass an automated artifact audit *prior* to peer review. The audit should confirm the availability of six items including (i) full source code, (ii) immutable datasets, (iii) pretrained model checkpoints, (iv) an executable evaluation script, (v) a machine-readable environment descriptor, and (vi) concise documentation sufficient to invoke the pipeline. Persistent DOI-minting repositories such as Zenodo, HuggingFace-Hub, Figshare, OSF, or Mendeley Data already provide the necessary infrastructure to host these artifacts at scale and with minimal cost to authors. Adopting this minimal standard would align computer science publication practice with the norms of experimental science and render future advances in Neuro-Symbolic AI transparent, verifiable, and readily extensible. The community must therefore institutionalize immutable artifact disclosure, requiring and archiving scientific works prior to publication, and thereby ensure that progress is measurable and cumulative by lowering the barrier for full pipeline reproduction of Neuro-Symbolic systems.

References

- 2020. Research, reuse, repeat. *Nat. Mach. Intell.*, 2(12): 729–729.
- Association for Computing Machinery. 2020. Artifact Review and Badging, Version 1.1. <https://www.acm.org/publications/policies/artifact-review-and-badging-current>.
- Baker, M. 2016. 1,500 scientists lift the lid on reproducibility. *Nature*, 533(7604): 452–454.
- Belz, A.; and Thomson, C. 2023. The 2023 ReproNLP Shared Task on Reproducibility of Evaluations in NLP: Overview and Results. In Belz, A.; Popović, M.; Reiter, E.; Thomson, C.; and Sedoc, J., eds., *Proceedings of the 3rd Workshop on Human Evaluation of NLP Systems*, 35–48. Varna, Bulgaria: INCOMA Ltd., Shoumen, Bulgaria.
- DeLong, L. N.; Gadiya, Y.; Galdi, P.; and Fleuriot, J. D. 2024. Mars: A neurosymbolic approach for interpretable drug discovery. *arXiv*.

Ferrari Dacrema, M.; Boglio, S.; Cremonesi, P.; and Jan-nach, D. 2021. A Troubling Analysis of Reproducibility and Progress in Recommender Systems Research. *ACM Trans. Inf. Syst.*, 39(2).

Henderson, P.; Islam, R.; Bachman, P.; Pineau, J.; Precup, D.; and Meger, D. 2018. Deep reinforcement learning that matters. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press. ISBN 978-1-57735-800-8.

Hensel, W. M. 2020. Double trouble? The communication dimension of the reproducibility crisis in experimental psychology and neuroscience. *Eur. J. Philos. Sci.*, 10(3).

Ioannidis, J. P. A. 2005. Why most published research findings are false. *PLoS Med.*, 2(8): e124.

Li, B.; Li, Z.; Du, Q.; Luo, J.; Wang, W.; Xie, Y.; Stepputtis, S.; Wang, C.; Katia, P. S.; Ravikumar, P. K.; Gray, A. G.; Si, X.; and Scherer, S. 2024. LogiCity: Advancing Neuro-Symbolic AI with Abstract Urban Simulation. *Neurips*.

Li, Z.; Huang, J.; and Naik, M. 2023. Scallop: A language for neurosymbolic programming. *ACM*.

López-Nicolás, R.; López-López, J. A.; Rubio-Aparicio, M.; and Sánchez-Meca, J. 2022. A meta-review of transparency and reproducibility-related reporting practices in published meta-analyses on clinical psychological interventions (2000-2020). *Behav. Res. Methods*, 54(1): 334–349.

McGowan, J.; Sampson, M.; Salzwedel, D. M.; Cogo, E.; Foerster, V.; and Lefebvre, C. 2016. PRESS peer review of electronic search strategies: 2015 guideline statement. *J. Clin. Epidemiol.*, 75: 40–46.

Miłkowski, M.; Hensel, W. M.; and Hohol, M. 2018. Replicability or reproducibility? On the replication crisis in computational neuroscience and sharing only relevant detail. *J. Comput. Neurosci.*, 45(3): 163–172.

Open Science Collaboration. 2015. PSYCHOLOGY. Estimating the reproducibility of psychological science. *Science*, 349(6251): aac4716.

Pawlik, M.; Hütter, T.; Kocher, D.; Mann, W.; and Augsten, N. 2019. A Link is not Enough – Reproducibility of Data. *Datenbank-Spektrum*, 19.

Pineau, J.; Vincent-Lamarre, P.; Sinha, K.; Larivière, V.; Beygelzimer, A.; d'Alché Buc, F.; Fox, E.; and Larochelle, H. 2021. Improving reproducibility in machine learning research (a report from the NeurIPS 2019 reproducibility program). *J. Mach. Learn. Res.*, 22(1).

Rubin, M. 2022. The costs of HARKing. *Br. J. Philos. Sci.*, 73(2): 535–560.

Semmelrock, H.; Ross-Hellauer, T.; Kopeinik, S.; Theiler, D.; Haberl, A.; Thalmann, S.; and Kowald, D. 2025. Reproducibility in machine-learning-based research: Overview, barriers, and drivers. *AI Mag.*, 46(2).

Vanderdonckt, J.; and Vatavu, R.-D. 2025. Context is Key for Reproducibility of Empirical Studies in Human-Computer Interaction. In *Proceedings of the 3rd ACM Conference on*

Reproducibility and Replicability, ACM REP '25, 41–50. New York, NY, USA: Association for Computing Machinery. ISBN 9798400719585.

Appendix

Data Availability. The supplementary materials supporting this study are deposited in the [<https://doi.org/10.5281/zenodo.21779234>]. The archive contains the paper-level reproduction and data-extraction records, missing-artifact annotations, audit protocols, annotator documentation, analysis scripts, generated aggregate results, and an offline copy of the results browser. A worked example demonstrating application of the data-extraction form is available through the NSAI-2025 data-extraction worked example. Paper-level outcomes can also be inspected through the interactive results browser.

Broader impacts

This work can positively affect the machine-learning community by encouraging more transparent, verifiable, and cumulative Neuro-Symbolic AI research. At the same time, stronger artifact-release requirements may increase burdens on authors, especially under-resourced groups, and may create pressure to release sensitive, proprietary, or legally restricted artifacts. These risks can be mitigated through anonymized review artifacts, persistent metadata, gated access, and documented exemptions where full public release is not ethically or legally possible.

Final List of Reproduced Works

A full list of the papers that were able to be reproduced for this study can be found here: NSAI-2025-reproducibility-study.

Citation	Title	Description
Acharya et al. (2025) Codebase	Neurosymbolic AI for Travel Demand Prediction: Integrating Decision Tree Rules into Neural Networks	This paper introduces a neuro-symbolic AI framework that blends decision tree rules with neural networks to predict travel demand. By combining interpretability with deep learning power, it achieves more accurate and explainable results for transportation planning and resource optimization.
Ishay et al. (2023) Codebase	Leveraging Large Language Models to Generate Answer Set Programs	They use LLMs to work through logic puzzle solving in a step-by-step manner.
Murali et al. (2019) Codebase	Composing Neural Learning and Symbolic Reasoning with an Application to Visual Discrimination	propose a compositional neurosymbolic framework that combines a neural network to detect objects and relationships with a symbolic learner that finds interpretable discriminators.
Ahmed et al. (2022) Codebase	Semantic probabilistic layers for neuro-symbolic learning	The paper presents Semantic Probabilistic Layers (SPLs), a neural network module that ensures structured-output predictions are always consistent with logical constraints, enabling accurate and tractable neuro-symbolic learning. SPLs modularly combine probabilistic inference and logical reasoning, outperforming previous methods in tasks requiring strict output validity.
Ahmed et al. (2022) Codebase	Neuro-symbolic entropy regularization	This paper presents neuro-symbolic entropy regularization, a unified framework that combines entropy regularization and neuro-symbolic learning for structured prediction tasks. By constraining entropy minimization to outputs that form valid structures (as defined by logical circuits), the approach yields models that are both more accurate and more likely to produce valid predictions, demonstrated across semi-supervised and fully-supervised experiments.
Akl (2024) Codebase	NeSy is alive and well: A LLM-driven symbolic approach for better code comment data generation and classification	This paper presents a neuro-symbolic workflow combining semantic rule-based decomposition with an LLM to generate controlled synthetic data for C-code comment classification. Empirical results show that this augmentation improves ML models (Voting Classifier, Random Forest, MLP) performance measured in F1 score.
Alam et al. (2024) Codebase	A Walsh Hadamard Derived Linear Vector Symbolic Architecture	The paper introduces the Hadamard-derived Linear Binding (HLB), a novel vector symbolic architecture that leverages properties of the Walsh-Hadamard transform for efficient, numerically stable vector binding in neuro-symbolic AI. HLB achieves state-of-the-art performance on both classical VSA benchmarks and selected deep learning tasks, outperforming previous VSA methods in terms of computational complexity, accuracy, and differentiability for modern neural architectures.
Daniele and Luciano (2022) Codebase	Knowledge Enhanced Neural Networks for relational domains	Paper extends knowledge enhanced NNs to handle relational data and shows that stacking multiple KE layers deals with rule dependencies, achieving better accuracy than baseline NNS on Citeseer citation network classification while also being faster than baseline Semantic Based Regularization (SBR) and Relational Neural Machines (RNM)
Ibrahimzada et al. (2024) Codebase	AlphaTrans: A Neuro-Symbolic Compositional Approach for Repository-Level Code Translation and Validation	Solves the task of code translation from one programming language to another. It does so using neural symbolic framework that breaks down the source code into fragments and also utilized the test code to ensure code was properly translated.
Alon et al. (2022) Codebase	Neuro-symbolic language modeling with automaton-augmented retrieval	The paper presents RETOMATON, a neuro-symbolic system that approximates costly nearest-neighbor datastore searches in retrieval-based language models using automaton states and pointer links, enabling substantial speed-ups while maintaining or improving perplexity.

Citation	Title	Description
Asai and Muise (2020) Codebase	Learning neural-symbolic descriptive planning models via cube-space priors: The voyage home (to STRIPS)	neuro-symbolic architecture is trained end-to-end to produce a succinct and effective discrete state transition model from images alone.
Aspis et al. (2022) Codebase	Embed2sym-scalable neuro-symbolic reasoning via clustered embeddings	The paper presents Embed2Sym, a scalable framework that combines neural perception and symbolic reasoning through clustered embeddings, enabling fast training, interpretability, and generalization in tasks that exceed the scalability of prior neuro-symbolic systems. Embed2Sym achieves state-of-the-art results and significantly reduces training time on complex reasoning tasks involving visual and symbolic inputs.
Badreddine et al. (2022) Codebase	Logic Tensor Networks	This paper presents Logic Tensor Networks (LTN), a neurosymbolic framework that supports querying, learning and reasoning with both rich data and abstract knowledge about the world. LTN introduces a fully differentiable logical language, called Real Logic, whereby the elements of a first-order logic signature are grounded onto data using neural computational graphs and first-order fuzzy logic semantics
Bairi et al. (2024) Codebase	CodePlan: Repository-Level Coding using LLMs and Planning	The paper introduces CodePlan, a framework that treats large-scale repository-level code edits (e.g., API migrations or temporal edits across many files) as a planning problem: it uses an LLM guided by static dependency and impact analysis to generate a chain of edits until the repository satisfies a correctness oracle. They evaluate on C# and Python codebases and show that it outperforms no-planning baselines in build success rate and matching ground truth edits.
Barbiero et al. (2023) Codebase	Interpretable neural-symbolic concept reasoning	The paper proposes the Deep Concept Reasoner (DCR), an interpretable concept-based model that uses neural networks to generate fuzzy logic rules from concept embeddings, and then executes those rules on concept truth degrees to make semantically meaningful and differentiable predictions.
Barnaby et al. (2023) Codebase	ImageEye: Batch Image Processing using Program Synthesis	Batch editing of images such as cropping out a desired object/person in a batch of 100+ images has not been an easy task. The paper describes a neuro symbolic approach to generate programs from user demonstrations that perform such tasks. They show the program can automate 96% of these tasks.
Biggio et al. (2021) Codebase	Neural symbolic regression that scales	The paper presents NeSymReS, a neural symbolic regression model that leverages large-scale pre-training of Transformers on procedurally generated equations and data, enabling scalable, efficient, and robust discovery of symbolic equations from input-output data pairs. The approach outperforms traditional and neural symbolic regression methods across diverse evaluation benchmarks.
Li et al. (2024) Codebase	LogiCity: Advancing Neuro-Symbolic AI with Abstract Urban Simulation	LogiCity models diverse urban elements using semantic and spatial concepts, such as IsAmbulance(X) and IsClose(X, Y). These concepts are used to define FOL rules that govern the behavior of various agents. Since the concepts and rules are abstractions, they can be universally applied to cities with any agent compositions, facilitating the instantiation of diverse scenarios
Carraro et al. (2024) Codebase	Mitigating data sparsity via neuro-symbolic knowledge transfer	Paper uses Logic Tensor Networks (LTN) to mitigate data sparsity issues in recommender systems. It combines Matrix Factorization (neural) models with First-Order Logic axioms (symbolic).

Citation	Title	Description
Chanin and Hunter (2023) Codebase	Neuro-symbolic commonsense social reasoning	present a novel system for taking social rules of thumb (ROTs) in natural language from the Social Chemistry 101 dataset and converting them to first-order logic where reasoning is performed using a neuro-symbolic theorem prover.
Dickens et al. (2024) Codebase	Convex and Bilevel Optimization for Neuro-Symbolic Inference and Learning	The paper presents a bilevel, gradient-based optimization framework for neural-symbolic (NeSy) learning, introducing a dual block coordinate descent algorithm and smooth formulation for efficient and scalable parameter learning, empirically validated on multiple datasets.
Chen and Jabbarvand (2024) Codebase	Neurosymbolic Repair of Test Flakiness	Introduces FlakyDoctor, a neuro-symbolic repair system that couples LLM-based patch synthesis with static analysis, compilation “stitching,” and targeted validation to repair order-dependent (OD) and implementation-dependent (ID) flaky tests. Evaluated on 873 flaky tests from 243 projects, achieving ~57% OD and 59% ID repair success and producing 79 previously un-fixed patches (19 merged PRs)
Chen et al. (2022) Codebase	ConvFinQA: Exploring the Chain of Numerical Reasoning in Conversational Finance Question Answering	Introduces a new dataset to study chain of numerical reasoning in question-answering.
Glanois et al. (2021) Codebase	Neuro-Symbolic Hierarchical Rule Induction	This paper proposes an efficient and interpretable neuro-symbolic model called Hierarchical Rule Induction (HRI) to solve Inductive Logic Programming (ILP) problems. This model is built from a set of meta-rules organized in a hierarchical structure, and it invents first-order rules by learning embeddings to match facts and predicates.
Yin et al. (2024) Codebase	MDD-5k: A New Diagnostic Conversation Dataset for Mental Disorders Synthesized via Neuro-Symbolic LLM Agents	Introduces a new framework to generate a dataset of diagnostic conversations between a patient and a doctor. It also provides a dataset with 5k high-quality long conversations with diagnosis results and treatment options.
Pryor et al. (2022) Codebase	NeuPSL: Neural Probabilistic Soft Logic	Introduce Neural Probabilistic Soft Logic (NeuPSL), a novel neurosymbolic (NeSy) framework that unites state-of-the-art symbolic reasoning with the low-level perception of deep neural networks. To model the boundary between neural and symbolic representations, they propose a family of energy-based models, NeSy Energy-Based Models, and show that they are general enough to include NeuPSL and many other NeSy approaches.
Dagan et al. (2023) Codebase	Dynamic Planning with a LLM	presents LLM Dynamic Planner (LLM-DP): a neuro-symbolic framework where an LLM works hand-in-hand with a traditional planner to solve an embodied task. Given action-descriptions, LLM-DP solves Alfworl faster and more efficiently than a naive LLM ReAct baseline.
Daggitt et al. (2024) Codebase	Vehicle: Bridging the embedding gap in the verification of neuro-symbolic programs	This paper presents Vehicle, a tool designed to bridge the "embedding gap" in the verification of neuro-symbolic programs—programs combining neural (machine learning) and symbolic (traditional code) components. Vehicle provides a unified, dependently-typed language and compiler that enables formal specifications for neural components to be integrated across different verification and training backends, demonstrated by verifying a neural network controller for an autonomous car.
Li et al. (2022) Codebase	Learning Signal Temporal Logic through Neural Network for Interpretable Classification	design a novel time function and sparse softmax function to improve the soundness and precision of the neural-STL framework. As a result, we can efficiently learn a compact STL formula for the classification of time-series data through off-theshelf gradient-based tools.

Citation	Title	Description
DeLong et al. (2024) Codebase	Mars: A neurosymbolic approach for interpretable drug discovery	Mechanism of Action Retrieval System (MARS) is a neurosymbolic approach for drug discovery that uses logical rules with learned rule weights. The model has better interpretability through dynamic weight learning from logical rules.
Dhanraj and Eliasmith (2025) Codebase	Improving Rule-based Reasoning in LLMs via Neurosymbolic Representations	introduces a novel neurosymbolic method that improves LLM reasoning by encoding hidden states into neurosymbolic vectors, enabling problem-solving within a neurosymbolic vector space
Done et al. (2023) Codebase	CORRPUS: Code-based Structured Prompting for Neurosymbolic Story Understanding	CoRRPUS shows the usefulness of code-based symbolic representations for enabling LLMs to perform better on story reasoning tasks.
Dong et al. (2019) Codebase	Neural Logic Machines	propose the Neural Logic Machine (NLM), a neural-symbolic architecture for both inductive learning and logic reasoning. NLMs exploit the power of both neural networks—as function approximators, and logic programming—as a symbolic processor for objects with properties, relations, logic connectives, and quantifiers.
Eiter et al. (2023) Codebase	A Modular Neurosymbolic Approach for Visual Graph Question Answering	This paper presents a modular neuro-symbolic architecture that processes images of graph-structured data (rather than symbolic graphs) by first using optical graph recognition and OCR to extract nodes/edges and labels, then parsing the question and using answer-set programming (ASP) for logical reasoning. This paper also introduces a new VGQA task and dataset (in 3 sets: tiny, small, medium) and establishes a baseline of 73% accuracy.
Misino et al. (2022) Codebase	VAEL: Bridging Variational Autoencoders and Probabilistic Logic Programming	This paper presents VAEL, a neuro-symbolic generative model integrating variational autoencoders (VAE) with the reasoning capabilities of probabilistic logic (L) programming. Besides standard latent subsymbolic variables, their model exploits a probabilistic logic program to define a further structured representation, which is used for logical reasoning.
Marconato et al. (2024) Codebase	BEARS Make Neuro-Symbolic Models Aware of their Reasoning Shortcuts	They propose to ensure NeSy models are aware of the semantic ambiguity of the concepts they learn, thus enabling their users to identify and distrust low-quality concepts. Starting from three simple desiderata, they derive bears (BE Aware of Reasoning Shortcuts), an ensembling technique that calibrates the model’s concept-level confidence without compromising prediction accuracy, thus encouraging NeSy architectures to be uncertain about concepts affected by RSSs.
van Krieken et al. (2025) Codebase	Neurosymbolic Diffusion Models	To overcome the limitations of the independence assumption, this paper introduces neurosymbolic diffusion models (NESYDMS), a new class of NeSy predictors that use discrete diffusion to model dependencies between symbols. their approach reuses the independence assumption from NeSy predictors at each step of the diffusion process, enabling scalable learning while capturing symbol dependencies and uncertainty quantification.
van Krieken et al. (2022) Codebase	A-NeSI: A Scalable Approximate Method for Probabilistic Neurosymbolic Inference	Introduce Approximate Neurosymbolic Inference (A-NESI): a new framework for PNL that uses neural networks for scalable approximate inference. A-NESI 1) performs approximate inference in polynomial time without changing the semantics of probabilistic logics; 2) is trained using data generated by the background knowledge; 3) can generate symbolic explanations of predictions; and 4) can guarantee the satisfaction of logical constraints at test time, which is vital in safety-critical applications.

Citation	Title	Description
Endo et al. (2023) Codebase	Motion Question Answering via Modular Motion Programs	Proposes a neuro symbolic framework to reason about motion sequences in human actions via symbolic reasoning and modular design. They say that it grounds motion through learning motion concepts, attribute neural operators and temporal relations. The task they chose is human motion QA for evaluate their NSPose method.
Karabulut et al. (2025) Codebase	Neurosymbolic Association Rule Mining from Tabular Data	Aerial+ trains an autoencoder on tabular data, then extracts association rules by feeding in test vectors with features set to 1. If the autoencoder reconstructs other features with high probability, it creates a rule. Generates way fewer rules than FP-Growth (2-10x less) with better coverage and runs faster on big datasets, plus makes rule-based classifiers like CORELS train faster without losing accuracy.
Evans et al. (2021) Codebase	https://www.sciencedirect.com/science/article/pii/S0004370221000722?via%3Dihub Making sense of raw input	The central contribution of this paper is a neuro-symbolic framework for distilling interpretable theories out of streams of raw, unprocessed sensory experience.
Feinman and Lake (2020) Codebase	Learning task-general representations with generative neuro-symbolic modeling	develop a generative neuro-symbolic (GNS) model of hand-written character concepts that uses the control flow of a probabilistic program, coupled with symbolic stroke primitives and a symbolic image renderer, to represent the causal and compositional processes by which characters are formed.
Feng et al. (2022) Codebase	Neuro-symbolic natural logic with introspective revision for natural language inference	Presents a NeSy framework to integrate natural logic with NNs to do natural language inference. They use RL to help guide this natural logic. Overall this improves the intuitive inference understandability of humans.
Gong et al. (2024) Codebase	Neuro-Symbolic Embedding for Short and Effective Feature Selection via Autoregressive Generation	Proposes a neuro-symbolic framework that models feature selection as an autoregressive token generation problem, learning embeddings for short and effective feature subsets with superior predictive performance compared to classical baselines.
Hakim et al. (2025) Codebase	ANSR-DT: An Adaptive Neuro-Symbolic Learning and Reasoning Framework for Digital Twins	ANSR-DT presents an adaptive neuro-symbolic framework integrating deep learning (CNN-LSTM) and symbolic reasoning (Prolog) for digital twins. The system generates interpretable insights, learns rules from data, and provides knowledge graph visualizations to support decision-making in industrial scenarios.
Princis et al. (2024) Codebase	Enhancing SQL Query Generation with Neurosymbolic Reasoning	Find a way to use language models to generate SQL queries. They introduce a new tool called Xander which helps
Ho Fung et al. (2024) Codebase	SymbolNet: Neural Symbolic Regression with Adaptive Dynamic Pruning	SymbolNet combines neural networks with symbolic regression through adaptive pruning to extract interpretable mathematical expressions while achieving high model compression. The method uses a custom training loop with threshold-based pruning to identify and eliminate less important network connections, enabling recovery of human-readable symbolic formulas from trained models without sacrificing predictive accuracy.
Howard et al. (2023) Codebase	Neurocomparatives: Neuro-symbolic distillation of comparative knowledge	This paper presents NeuroComparatives, a dataset of 8.8M comparative commonsense statements (e.g., "cats are typically smaller than dogs") generated using constrained beam search with GPT-2 and filtered using a discriminator trained on human annotations. The dataset aims to provide high-validity comparative knowledge for commonsense reasoning tasks.
Hu et al. (2023) Codebase	A multi-grained self-interpretable symbolic-neural model for single/multi-labeled text classification	propose a Symbolic-Neural model that can learn to explicitly predict class labels of text spans from a constituency tree without requiring any access to spanlevel gold labels

Citation	Title	Description
Ishay et al. (2024) Codebase	Think before You Simulate: Symbolic Reasoning to Orchestrate Neural Computation for Counterfactual Question Answering	The paper discusses a novel approach to do causal and temporal reasoning in video particularly for counterfactual reasoning. They make use of a causal graph and ASP to coordinate between perception and simulation modules.
Maene et al. (2024) Codebase	KLay: Accelerating Arithmetic Circuits for Neurosymbolic AI	The paper presents KLay, a scalable data structure and layerization algorithm that accelerates arithmetic circuit computation in neurosymbolic AI systems, enabling efficient forward and backward passes. It demonstrates near-linear runtime scaling across five orders of magnitude, significantly outperforming baseline methods on logic-based inference tasks.
Ying et al. (2025) Codebase	A Neural Symbolic Model for Space Physics	PhyE2E presents an end-to-end transformer-based framework for physics-informed symbolic regression that incorporates dimensional analysis as an inductive bias through oracle-guided divide-and-conquer strategies combined with MCTS refinement. The method achieves state-of-the-art performance on the Feynman equations benchmark, improving symbolic accuracy by 10-29% over existing methods like PySR, uDSR, TPSR, and LaSR.
Kricheli et al. (2024) Codebase	Error Detection and Constraint Recovery in Hierarchical Multi-Label Classification without Prior Knowledge	his paper presents an approach that uses Error Detection Rules (EDR) to learn the failure modes of machine learning models, relaxing the common assumption that hierarchical constraints must exist beforehand. These learned rules are effective at detecting classifier errors and can be leveraged as constraints for Hierarchical Multi-label Classification (HMC), allowing for the recovery of explainable constraints even when they are not provided initially.
Ma et al. (2020) Codebase	Knowledge-driven Data Construction for Zero-shot Evaluation in Commonsense Question Answering	vary the set of language models, training regimes, knowledge sources, and data generation strategies, and measure their impact across tasks. Extending on prior work, we devise and compare four constrained distractor-sampling strategies
Katz et al. (2021) Codebase	Tunable Neural Encoding of a Symbolic Robotic Manipulation Algorithm	present a neurocomputational controller for robotic manipulation based on the recently developed “neural virtual machine” (NVM). Authors program the NVM with a symbolic algorithm that solves blocks-world restacking problems, and execute it in a robotic simulation environment
Kelly et al. (2023) Codebase	There and back again: extracting formal domains for controllable neurosymbolic story authoring	Demonstrate that languagemodels can be used to author narrative planning domainsfrom natural language stories with minimal human intervention. Second, authors explore the reverse, demonstrating that we can use logical story domains and plans to produce storiesthat respect the narrative commitments of the planne
Kohaut et al. (2024) Codebase	Probabilistic Mission Design in Neuro-Symbolic Systems	Authors propose Probabilistic Mission Design (ProMis), a novel neuro-symbolic approach to navigating UAS within legal frameworks. ProMis is an interpretable and adaptable system architecture that links uncertain geospatial data and noisy perception with declarative, Hybrid Probabilistic Logic Programs (HPLP) to reason over the agent’s state space and its legality
Kohaut et al. (2024) Codebase	The Constitutional Filter	introduces an approach for Bayesian estimation of agents expected to comply with a human-interpretable neuro-symbolic model we call its Constitution. Hence, autors present the Constitutional Filter (CoFi), leading to improved tracking of agents by leveraging expert knowledge, incorporating deep learning architectures, and accounting for environmental uncertainties

Citation	Title	Description
Kouvaros and Botoeva (2024) Codebase	Formal verification of parameterised neural-symbolic multi-agent systems	This paper presents VENMAS, a formal verification toolkit for multi-agent systems with neural components, enabling bounded temporal logic verification of parameterized neural-symbolic agents. The work extends existing verification methods to handle systems with an arbitrary number of homogeneous agents using symbolic abstraction techniques.
Kulmanov et al. (2021) Codebase	Semantic similarity and machine learning with ontologies	provide an overview over the methods that use ontologies to compute similarity and incorporate them in machine learning methods; in particular, the authors outline how semantic similarity measures and ontology embeddings can exploit the background knowledge in ontologies and how ontologies can provide constraints that improve machine learning models.
Kulmanov et al. (2024) Codebase	Protein function prediction as approximate semantic entailment	The Gene Ontology (GO) is a formal, axiomatic theory with over 100,000 axioms that describe the molecular functions, biological processes and cellular locations of proteins in three subontologies. Developed DeepGO-SE, a method that predicts GO functions from protein sequences using a pretrained large language model. DeepGO-SE generates multiple approximate models of GO, and a neural network predicts the truth values of statements about protein functions in these approximate models.
Lee et al. (2024) Codebase	Diakop: Dialogue-based knowledge-oriented programming for neural-symbolic knowledge base question answering	present Dialogue-based Knowledge-oriented Programming system (DiaKoP), a system with a chat interface designed for multi-turn knowledge base question answering (KBQA). DiaKoP enables users to decompose complex questions into multiple simpler follow-up questions and interact with the system to obtain answers.
Li et al. (2020) Codebase	Closed loop neural-symbolic learning via integrating neural perception, grammar parsing, and symbolic reasoning	This paper proposes a Neural-Grammar-Symbolic (NGS) model that improves upon inefficient reinforcement learning approaches by using a grammar model to bridge neural perception and symbolic reasoning, along with a novel back-search algorithm to learn from incorrect predictions. The experiments, conducted on handwritten formula recognition and visual question answering tasks, demonstrate that this method significantly outperforms reinforcement learning models in terms of performance, convergence speed, and data efficiency.
Li et al. (2022) Codebase	AdaLoGN: Adaptive Logic Graph Network for Reasoning-Based Machine Reading Comprehension	present a neural-symbolic approach which, to predict an answer, passes messages over a graph representing logical relations between text units.
Li et al. (2023) Codebase	Scallop: A language for neurosymbolic programming	present Scallop, a language which combines the benefits of deep learning and logical reasoning. Scallop enables users to write a wide range of neurosymbolic applications and train them in a data- and compute-efficient manner. It achieves these goals through three key features: 1) a flexible symbolic representation that is based on the relational data model; 2) a declarative logic programming language that is based on Datalog and supports recursion, aggregation, and negation; and 3) a framework for automatic and efficient differentiable reasoning that is based on the theory of provenance semirings.

Citation	Title	Description
Li et al. (2024) Codebase	Relational Programming with Foundation Models	implement VIEIRA by extending the SCALLOP compiler with a foreign interface that supports foundation models as plugins. We implement plugins for 12 foundation models including GPT, CLIP, and SAM. We evaluate VIEIRA on 9 challenging tasks that span language, vision, and structured and vector databases. Our evaluation shows that programs in VIEIRA are concise, can incorporate modern foundation models, and have comparable or better accuracy than competitive baselines.
Murphy et al. (2024) Codebase	Autoformalizing Euclidean Geometry	use theorem provers to fill in such diagrammatic information automatically, so that the LLM only needs to autoformalize the explicit textual steps, making it easier for the model.
Manginas et al. (2024) Codebase	NeSyA: Neurosymbolic Automata	how that symbolic automata can be integrated with neural-based perception, under probabilistic semantics towards an end-to-end differentiable model. Their proposed hybrid model, termed NESYA (Neuro Symbolic Automata) is shown to either scale or perform more accurately than previous NeSy systems in a synthetic benchmark and to provide benefits in terms of generalization compared to purely neural systems in a realworld event recognition task
Mejri et al. (2024) Codebase	LARS-VSA: A Vector Symbolic Architecture For Learning with Abstract Rules	The paper presents LARS-VSA, a neuro-symbolic framework leveraging hyperdimensional computing for abstract rule learning with compositional generalization. It introduces a high-dimensional attention mechanism and demonstrates superior generalization, efficiency, and accuracy over contemporary neural and neuro-symbolic baselines on multiple relational reasoning and math tasks.
Olausson et al. (2023) Codebase	(sic) LINC: A Neurosymbolic Approach for Logical Reasoning by Combining Language Models with First-Order Logic Provers	investigate the validity of instead reformulating such tasks as modular neurosymbolic programming, which we call LINC: Logical Inference via Neurosymbolic Computation. In LINC, the LLM acts as a semantic parser, translating premises and conclusions from natural language to expressions in first-order logic. These expressions are then offloaded to an external theorem prover, which symbolically performs deductive inference.
Post et al. (2024) Codebase	Accelerating UMR adoption: Neuro-symbolic conversion from AMR-to-UMR with low supervision	The paper proposes a neuro-symbolic method for converting AMR (Abstract Meaning Representation) roles to UMR (Uniform Meaning Representation) roles by integrating animacy parsing and logic rules with a neural network. The approach addresses non-deterministic role mappings with minimal human supervision, achieving 75.81% accuracy compared to a 62.35% baseline neural network.
Premisri and Kordjamshidi (2024) Codebase	Neuro-symbolic Training for Reasoning over Spatial Language	This paper presents SpaRTUNQChain, a neuro-symbolic framework that combines BERT embeddings with spatial logic constraints for question-answering over spatial language. The system uses the DomiKnowS framework to enforce compositional reasoning rules during training, improving accuracy on spatial reasoning benchmarks.
Quan et al. (2025) Codebase	Peirce: Unifying material and formal reasoning via llm-driven neuro-symbolic refinement	introduce PEIRCE, a neuro-symbolic framework designed to unify material and formal inference through an iterative conjecture–criticism process.
Roig Vilamala et al. (2023) Codebase	DeepProbCEP: A neuro-symbolic approach for complex event processing in adversarial settings	Introduces DeepProbCEP, a differentiable ProbLog + CNN hybrid that trains end-to-end from complex-event labels.

Citation	Title	Description
Skryagin et al. (2024) Codebase	Scalable Neural-Probabilistic Answer Set Programming	This paper introduces Answer Set Networks (ASNs), a novel neural-symbolic solver based on Graph Neural Networks (GNNs) designed to overcome the high computational costs and CPU-bound nature of traditional Answer Set Programming (ASP) solvers. By translating ASP problems into a graph format that leverages GPU parallelization, ASNs outperform existing systems and enable new applications like fine-tuning Large Language Models (LLMs) with logic and solving large-scale drone navigation tasks.
Tammet et al. (2023) Codebase	An Experimental Pipeline for Automated Reasoning in Natural Language (Short Paper)	The paper presents NLPsSolver, an experimental end-to-end pipeline that parses English text with a neural UD parser, converts it into extended first-order logic with defaults and confidences, performs reasoning with a high-performance default logic engine, and converts proofs back into natural language answers and explanations. The authors evaluate the system on toy NLI/QA examples, a subset of HANS, and AllenAI ProofWriter demos, showing strong performance there and highlighting the pipeline as a basis for future hybrid neuro-symbolic systems.
Winters et al. (2021) Codebase	DeepStochLog: Neural Stochastic Logic Programming	This paper introduces DeepStochLog, a novel neural-symbolic framework that integrates neural networks into stochastic definite clause grammars (SDCGs) to define a probability distribution over possible derivations. The experimental evaluation shows this approach scales significantly better than methods based on neural probabilistic logic programs and achieves state-of-the-art results on several challenging neural-symbolic tasks.
Trinh et al. (2024) Codebase	Solving olympiad geometry without human demonstrations	AlphaGeometry is a neuro-symbolic system that uses a neural language model, trained from scratch on our large-scale synthetic data, to guide a symbolic deduction engine through infinite branching points in challenging problems.
Wu et al. (2024) Codebase	LLM Meets Bounded Model Checking: Neuro-symbolic Loop Invariant Inference	proposes LaM4Inv, a neuro-symbolic framework that combines large language models (Llama-3-8B, GPT-3.5, GPT-4, GPT-4-Turbo) with bounded model checking (ESBMC + SMT solvers) to automatically infer loop invariants for C programs. It evaluates LaM4Inv on an expanded benchmark of 316 loop-invariant problems, showing large gains in the number of verified loops compared to classical invariant generators and recent LLM-based baselines.
Wu and Nakayama (2024) Codebase	MILE: Memory-Interactive Learning Engine for Neuro-Symbolic Solutions to Mathematical Problems	MILE is a neuro-symbolic math word problem solver that predicts formulas using a memory-based decoder instead of tree-structured decoding, letting it handle more complex computation graphs. It beats existing methods on Math23K by learning formulas as rules.
Kang et al. (2025) Codebase	Neuro-Conceptual Artificial Intelligence: Integrating OPM with Deep Learning to Enhance Question Answering Quality	introduceNeuro-ConceptualArtificial Intelligence(NCAI),aspecializationof the neuro-symbolicAI approach that integrates conceptual modeling usingObject-Process Methodology (OPM) ISO19450:2024with deeplearningtoenhancequestion-answering (QA)quality
Yang et al. (2023) Codebase	Learning to Solve Constraint Satisfaction Problems with Recurrent Transformer	Constraint satisfaction problems (CSPs) are about finding values of variables that satisfy the given constraints. We show that Transformer extended with recurrence is a viable approach to learning to solve CSPs in an end-to-end manner, having clear advantages over state-of-the-art methods such as Graph Neural Networks, SATNet, and some neuro-symbolic models.

Citation	Title	Description
Yang et al. (2023) Codebase	Injecting Logical Constraints into Neural Networks via Straight-Through Estimators	This paper introduces CL-STE, a method to inject discrete logical constraints into neural networks by representing the constraints as a loss function and using a Straight-Through Estimator (STE) to enable gradient-based optimization. By leveraging GPUs and avoiding heavy symbolic computation, this technique scales significantly better than existing neuro-symbolic methods and allows various architectures, like CNNs and GNNs, to learn from constraints with fewer or no labeled data.
Li et al. (2025) Codebase	Proving Olympiad Inequalities by Synergizing LLMs and Symbolic Reasoning	we introduce a neuro-symbolic tactic generator that synergizes the mathematical intuition learned by LLMs with domain-specific insights encoded by symbolic methods.