Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

The Invisible Infrastructure of Biomedical Research: An Exploratory Analysis of Software Mentions and Sentence-Level Citation Marking

Md Motasim billa

mbilla@ualr.edu

University of Arkansas at Little Rock

Research Article

Keywords: software citation, research software, software mentions, scientometrics, entity disambiguation, credit attribution

Posted Date: October 30th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11264021/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Abstract

Background

Research depends on software, yet software is seldom cited formally. Existing prevalence estimates come mainly from small, hand-coded samples and are usually reported as a single aggregate rate, which cannot show whether under-citation differs between classes of tool.

Methods

We conducted an exploratory, descriptive analysis of a deterministically defined subcorpus of the Chan Zuckerberg Initiative (CZI) Software Mentions dataset (799,756 mentions in 138,019 biomedical papers, 1980–2022). Mentions were assigned to five mention styles and six software types using published regular-expression and lexicon rules. The outcome, citation marking, was defined as the presence of a citation marker in the sentence that contains the software name; it is distinct from reference-list citation. We used contingency analysis (Cramér's V) and logistic regression with paper-clustered standard errors. A name-ambiguity diagnostic compared registry matching with a rule-based context classifier on 1,961 curated mentions of eight terms. The dataset's curation labels were not used as a filter: 15.0% of definitively labelled mentions are classified as not software.

Results

Overall, 10.7% of mentions were citation-marked and 69.7% were bare in-text names. Marking differed by software type (Cramér's V = 0.219), from 2.2% for statistical packages and 1.0% for reference managers (7 of 731 mentions) to 18.0% for machine-learning frameworks. Relative to domain-specific tools, the adjusted odds of marking were higher for machine-learning frameworks (OR = 1.70) and programming languages (OR = 1.47) and lower for statistical packages (OR = 0.23). Developer attribution was associated with lower odds of marking (OR = 0.29) and URLs with higher odds (OR = 1.28). The analysis found no robust time trend. The registry contained none of the eight ambiguous terms, and the context rules reached F1 = 0.591, a transparent baseline well below current transformer-based systems.

Conclusions

In this subcorpus, sentence-level citation marking of software is low and differs systematically by type of tool, being lowest for general-purpose statistical software. Because the instruments lack validation against human annotation and the subcorpus is not a probability sample, the findings are exploratory; they indicate where software-citation policy and validation work could most usefully be focused.

1. Introduction

Much contemporary research relies on software. Researchers perform sequence alignment, image quantification, statistical estimation and model training with computational tools, and these tools shape what a study can conclude. Surveys of computational practice report that researchers spend a substantial share of their working time developing, configuring and running such tools (Hannay et al., 2009; Prabhu et al., 2011).

The scholarly record, however, was designed to track texts rather than instruments, and it captures this dependence poorly. Software usually enters the record through a sentence in the methods section. A tool named in this way is visible to a human reader but largely invisible to the infrastructure that allocates credit, measures impact and reconstructs methodological provenance. Howison and Bullard (2016) documented this mismatch in the biology literature, showing that software is frequently mentioned in forms that cannot be resolved to a citable object. Comparable deficits have since been reported across disciplines, across tool families and in indexes of non-textual research outputs (Pan et al., 2016; Park & Wolfram, 2019; Schindler et al., 2022a). Software labour can also go unrecorded beyond the mention itself: recent work on contribution mapping reports that nearly 30% of articles with a linked code repository include code contributors who are not listed as authors (Brown et al., 2025).

This gap has three consequences. First, developers accrue little citation-based credit, which weakens the professional case for sustained maintenance in a reward system that counts references (Howison & Herbsleb, 2011; Singh Chawla, 2016). Second, funders cannot readily measure the return on investment in software, because the usage they finance leaves no indexed trace that can be linked back to the funding (Howison et al., 2015; Nesbitt et al., 2024). Third, reproducibility suffers, because a mention that omits version and provenance allows neither re-execution nor re-identification of the resource (Baker, 2016; Stodden et al., 2018).

The policy response is well developed in principle. The FORCE11 Software Citation Principles hold that software should be treated as a first-class research object (Smith et al., 2016), subsequent guidance has translated these principles into editorial requirements (Katz et al., 2021), and the FAIR (Findable, Accessible, Interoperable, Reusable) principles have been extended from data to research software (Wilkinson et al., 2016; Barker et al., 2022). The technical means also exist, in the form of machine-readable citation metadata, persistent and research-resource identifiers, and archival references to source code (Bandrowski & Martone, 2016; Di Cosmo et al., 2020). What remains comparatively scarce is large-scale empirical evidence on where, and for which classes of software, citation actually occurs.

Measurement has advanced along three lines. Gold-standard annotation has produced accurate but relatively small corpora, notably Softcite (Du et al., 2021) and SoMeSci (Schindler et al., 2021), building on earlier rule-based and bootstrapped studies of the biomedical literature (Duck et al., 2014, 2016; Pan et al., 2015) reviewed by Krüger and Schindler (2020). Large-scale extraction with supervised mention recognizers has been applied to millions of full texts (Lopez et al., 2021; Schindler et al., 2022b), and Istrate et al. (2022) released a machine-extracted dataset far larger than hand annotation permits, part of which carries human curation labels. Independent re-annotation of samples from machine-extracted

mention datasets has, however, found that a substantial share of extracted strings do not denote software (Druskat et al., 2024), so analyses built on such data must state how that noise is handled. Most recently, transformer-based joint frameworks for named-entity recognition (NER) and relation extraction (RE), sometimes trained with data generated by large language models, have reached a macro-F1 of about 0.89 on in-distribution shared-task data and about 0.63 on out-of-distribution data (Upadhyaya et al., 2025; Rastogi & Tiwari, 2025), and prompted large language models have been evaluated against gold-standard corpora (Pride et al., 2025). Open bibliographic infrastructure such as OpenAlex (Priem et al., 2022) now makes it possible, in principle, to attach disciplinary metadata to extracted mentions.

Two gaps remain. The first is descriptive. Prevalence is usually reported as a single aggregate rate, and where variation has been examined it has typically been by discipline or for a single exemplar tool (Li et al., 2016, 2017, 2019; Pan et al., 2018) rather than by class of instrument. An aggregate rate cannot distinguish a diffuse writing norm from a set of category-specific conventions, and the distinction matters for policy: a diffuse norm calls for general encouragement, whereas category-specific conventions call for targeted requirements. The second gap is methodological. Software names are often polysemous and highly variable in form (Duck et al., 2015), so pipelines that resolve mentions by matching names against registries may mis-resolve exactly the tools that are most widely used, a familiar failure mode in entity linking generally (Shen et al., 2015) and in biomedical resource disambiguation in particular (Ozyurt et al., 2016). Which names retain ambiguity after extraction and curation, and how far sentence context resolves it, have not been examined against curated labels at scale.

This exploratory study addresses both gaps within a defined subcorpus of 799,756 mentions and asks three research questions. RQ1: How are software mentions distributed across mention styles, and does that distribution differ by class of software? RQ2: Which mention-level features are associated with citation marking, and how large are those associations on the probability scale? RQ3: Where does residual name ambiguity lie in curated data, how completely does registry matching cover it, and how far do simple contextual rules resolve it?

The study makes three contributions, each subject to the limitations set out in Section 5. It provides a type-resolved description of sentence-level citation marking at a scale beyond hand annotation and shows that a single aggregate rate conceals large differences between tool classes. It documents an association between the form of attribution and citation marking: in-text naming of a developer co-occurs with the absence of a citation marker, whereas URLs co-occur with its presence. Finally, it reports that the names selected in advance as likely to be ambiguous show no label variation in the curated data, and that the terms that do show such variation are absent from a conventional tool registry. All instruments are published in full (Supplementary Table S6) so that the analysis can be reproduced and, in future work, validated.

2. Data and Methods

2.1 Design and target of inference

This is a cross-sectional, observational, and exploratory analysis of machine-extracted software mentions. The unit of analysis is the mention, that is, a single occurrence of a software name within a sentence, rather than the paper or the software product. The analytic population is a defined subcorpus of the CZI dataset (Section 2.2), not the dataset as a whole. Every reported quantity is therefore a descriptive statistic of that subcorpus as extracted, and it inherits whatever extraction error the source dataset carries. Confidence intervals express estimation uncertainty about the process generating the observed outcomes, not sampling uncertainty about a wider corpus, and generalisation beyond the subcorpus must be argued rather than assumed.

Because all software classes are measured within one corpus by one instrument, comparisons between classes are less sensitive to corpus composition than absolute levels are, provided that measurement error does not differ systematically between classes (Section 2.3). The main conclusions are therefore stated as relative contrasts. The design is associational and supports no causal claims. No human participants were involved, and all inputs are openly licensed.

2.2 Data source and subcorpus definition

Mentions were obtained from the CZ Software Mentions dataset (Istrate et al., 2022), which contains software names extracted from PubMed Central Open Access full texts by a fine-tuned SciBERT recogniser (Beltagy et al., 2019), together with publication dates, document identifiers and, for part of the data, curation labels. The analysis required three fields: sentence text, publication date and curation label. Of the 24 tables in the release, four carried all three fields. One table was retained per corpus subset, with the disambiguated variant preferred where one existed. The two tables analysed were therefore the disambiguated table of the PMC-OA commercial-use subset and the raw table of the non-commercial subset; because the two tables differ in name normalisation, this choice is revisited in Section 5.

Reading was capped at 400,000 records per table, taken sequentially from the start of each file, and sentence text in the source tables was limited to 200 characters. After removal of empty and non-informative software strings, 399,918 mentions remained from the commercial-use table and 399,838 from the non-commercial table, giving a subcorpus of 799,756 mentions of 107,248 distinct software strings from 138,019 papers. This selection rule is deterministic and reproducible from the public files, but it depends on file position rather than on any property of the mention. File order in the source release was not audited; if position correlates with year, journal or software family, the subcorpus may not be representative of the full dataset. For this reason the subcorpus is treated as the analytic population rather than as a probability sample (Section 2.1).

The dataset is known to contain strings that are not software (Druskat et al., 2024). We did not apply a separate filtering step but retained the dataset's own curation labels. Of the 799,756 mentions, 431,375 (53.9%) are labelled as software, 75,945 (9.5%) as not software and 25,322 (3.2%) as unclear, and

267,114 (33.4%) carry no curation label. Among mentions with a definite label, 15.0% are labelled as not software. The labels define the reference set for the disambiguation diagnostic (Section 2.5). They were not used to filter the prevalence and regression analyses (Sections 3.1–3.3), which use all 799,756 mentions. The absolute rates reported there, therefore, include an unknown share of non-software strings, and the comparisons between classes are unaffected only to the extent that this noise is distributed evenly across them (Section 5).

Publication year was parsed from the date field and accepted if it fell within a plausibility window from 1950 to one year after the year of analysis. A valid year was obtained for 799,748 mentions (> 99.99%). The eight mentions without a parseable date were retained in the prevalence tables (Table 1; N = 799,756) and excluded from the regression, which requires a year (N = 799,748). Observed years range from 1980 to 2022. Papers were identified by their normalised Digital Object Identifier (DOI) where available (782,244 mentions, 97.8%) and otherwise by their PubMed Central identifier (17,512 mentions, 2.2%). Figure 1 shows the flow of mentions through the analyses.

2.3 Outcome: sentence-level citation marking

A mention is defined as citation-marked if the sentence carrying the software name contains a citation marker: a bracketed or parenthetical numeral or numeral range, a superscript numeral, or an author–year construction (patterns in Supplementary Table S6). The reference list was not parsed. The outcome therefore measures co-occurrence of a software name with the textual apparatus that a reference-list citation produces and that a bare mention does not. Sentence- and mention-level operationalisations of this kind have been used in earlier large-scale work (Schindler et al., 2022a), and the underlying distinction between formal citation and informal mention goes back to Howison and Bullard (2016).

This construct is not interchangeable with reference-list citation. Howison and Bullard (2016) hand-coded every mention in 90 biology articles against the reference list and reported a formal-citation share of roughly 31–43%. Our instrument credits a mention only when a marker occurs in the same sentence, so citations placed in an adjacent sentence, a table, a figure legend, or supplementary material are not counted. The difference between their figure and ours is discussed in Section 4.1.

Four sources of measurement error are inherent in the instrument. First, a marker in the sentence may belong to an adjacent claim rather than to the software, which inflates the count. Second, citations outside the sentence, or beyond the 200-character text limit, are missed, which deflates it. Third, some patterns match strings that are not citations: parenthetical numerals can denote enumerations or quantities, and the superscript pattern also matches unit exponents such as cm^2 or 10^6, which are common in biomedical text; this inflates the count. Fourth, where the extractor has returned a string that is not software, the mention still enters the denominator. The net direction of these errors is not identified. The absolute level is therefore an estimate of sentence-level citation marking as measured by this instrument, not of reference-list citation.

Comparisons between software classes are less exposed, because the same fixed pattern set is applied to every mention regardless of class, year, or corpus subset. If misclassification is non-differential, it tends to attenuate contrasts towards the null, although attenuation is not guaranteed in any single study (Jurek et al., 2005). Non-differentiality cannot be verified here and is plausibly violated in at least two ways: the 200-character limit may affect classes differently if software names occupy different positions in the sentence, and unit exponents and decimal quantities may be more frequent in some kinds of sentence than others. These possibilities are discussed in Section 5.

2.4 Classification of mention style and software type

Each mention was assigned to exactly one of five mention styles by applying four regular-expression detectors to its sentence: citation marker, version string (dotted numbers with optional prefixes), URL (web protocols and common code-hosting and identifier domains), and developer attribution (parenthetical company, institute or vendor designations). These attributes correspond to metadata relations annotated in gold-standard software-mention corpora (Schindler et al., 2021) and recur throughout the extraction literature (Krüger & Schindler, 2020). Styles were assigned in the priority order citation-marked, name with version, developer attribution, URL only, and in-text name only. The three structural markers (version, URL and developer attribution) were also retained as separate binary covariates, independently of the assigned style.

Two properties of the detectors affect interpretation. The version pattern matches any dotted number, so it detects genuine version strings but also decimal quantities such as *p*-values, concentrations, and effect sizes; the name-with-version style and the version covariate are therefore best read as ‘a dotted number occurs in the sentence’. The developer pattern is keyed to corporate designators and under-detects academic and community developers. Both properties are documented with the patterns in Supplementary Table S6.

Software strings were classified into six types (statistical package, programming language, machine-learning framework, GIS or visualisation tool, reference manager, and domain-specific tool) using a curated lexicon applied by exact and then word-boundary matching to the software string alone. Strings matching no lexicon entry were assigned to the residual domain-specific category. The lexicon assigns each string to exactly one type in a fixed priority order (Supplementary Table S6), so the type categories are mutually exclusive.

No second human coder annotated a validation sample, and no inter-rater reliability coefficient is available. Every pattern and lexicon entry is reproduced verbatim in Supplementary Table S6, so the classification can be reproduced exactly, but reproducibility is not a substitute for validated accuracy. Validation against an independently annotated sample and against the SoMeSci and Softcite gold standards was part of the original study plan and was not carried out (Supplementary Table S1).

2.5 Entity-disambiguation diagnostic

The 507,320 mentions with a definite curation label (software or not software) form the reference set for this diagnostic. The diagnostic was planned for six names chosen in advance as likely to be polysemous. None of these names carried a single negative label in the reference set (Supplementary Table S2), so no discrimination task existed for them; this result is reported in Section 3.4. A pre-specified fallback rule therefore selected the eight most frequent strings with at least 30 labelled mentions and a positive (software) rate strictly between 5% and 95%: zoom, string, star, oncomine, quest, cluster, prime and prosite (1,961 mentions).

Two approaches were compared. In Approach A (registry matching), a mention is classified as software if and only if its name appears in a registry assembled from the type lexicon plus a small set of additional well-known tools; context is ignored. This is the dictionary-lookup baseline conventional in this literature (Duck et al., 2015). In Approach B (contextual rules), a mention is classified as software if its sentence contains at least one term from a software-context lexicon and no term from a competing-sense lexicon (both listed in Supplementary Table S6). Approach B is a simple rule-based classifier; it is neither an embedding model (Beltagy et al., 2019) nor a transformer-based joint NER and RE system of the kind that now defines the state of the art (Upadhyaya et al., 2025; Rastogi & Tiwari, 2025; Pride et al., 2025).

Two features of the comparison limit what it can show. The registry was assembled before the eight evaluated terms were known and contains none of them, so Approach A has zero coverage by construction and its metrics are fixed at zero. The competing-sense lexicon was likewise written for the six originally selected names and contains no entries specific to the eight evaluated terms; for these terms, Approach B's decision therefore rests almost entirely on the software-context lexicon. We retain the comparison as a coverage-and-context diagnostic rather than a contest between competitive systems: it shows whether a conventional registry contains the names that curated data identify as ambiguous, and to what extent generic software vocabulary in the sentence separates the two senses. We retained the rule-based form because every decision can be traced to a lexicon entry and because the rules operate on the 200-character sentence windows available here.

For each term, precision, recall, and F1 were computed for the binary software versus not-software decision on a pre-extracted string, where precision = TP/(TP + FP), recall = TP/(TP + FN), and F1 = 2 × precision × recall/(precision + recall), with TP, FP and, FN denoting true positives, false positives, and false negatives. No relation extraction was attempted, so no RE score is reported. The difference $\Delta F1 = F1(B) - F1(A)$ was tested by approximate randomisation of paired per-mention predictions within each term, a test recommended for paired system comparisons on this class of measure (Dror et al., 2018). Because F1(A) is zero by construction, ΔF1 equals F1(B) and the test indicates only whether Approach B performs better than a classifier with no coverage.

2.6 Statistical analysis

Dependence between software type and mention style was tested with Pearson's chi-square, and its strength was summarised with Cramér's V, because chi-square statistics are uninformative about

magnitude at this sample size. Citation marking was modelled by logistic regression:

$$\text{logit}(p \sim i\sim) = \beta_0 + \beta_1 \text{Type}_i + \beta_2 \text{Year}_{ci} + \beta_3 \text{Version}_i + \beta_4 \text{URL}_i + \beta_5 \text{Developer}_i \quad (1)$$

where Type is a set of indicator variables with domain-specific tool as the reference category and Year_c is publication year centered on its mean. The model was estimated by maximum likelihood on the 799,748 mentions with a valid year. Predictors were screened for perfect separation, and none was dropped. These five predictors are the only mention-level attributes available in the subcorpus without bibliographic linkage. The type indicators are mutually exclusive by construction, and the structural markers are binary and rare (Section 3.1); no collinearity problem arose.

Because multiple mentions frequently come from the same paper, standard errors were clustered by paper (Cameron & Miller, 2015), and the paper-clustered results are reported in the main text (Table 2); model-based standard errors are given for comparison in Supplementary Table S3. Coefficients are reported as odds ratios, $\text{OR}_k = \exp(\beta \sim k\sim)$. Because odds ratios are a poor guide to magnitude when the outcome is uncommon, model-implied probabilities for illustrative covariate profiles were obtained by inverse-logit transformation of the linear predictor. Overall fit was assessed with the likelihood-ratio statistic $G = -2(\ln L_0 - \ln L_1)$ against the intercept-only model, and Nagelkerke, McFadden, and Cox–Snell pseudo-R^2 values are reported (Nagelkerke, 1991). The model is a descriptive summary of mention-level regularities, not a causal or complete model: author, funder, journal, publisher and disciplinary characteristics are unobserved, and the coefficients are marginal with respect to them.

Two further sensitivity analyses were performed. First, the model was refitted after excluding the single most frequent software string, SPSS, to check whether the statistical package coefficient depends on one product (Supplementary Table S4). Second, because the corpus is very unevenly distributed across years, the temporal association was re-examined within the well-populated window 2008–2021 by binomial regression of annual counts on year. This check is unadjusted and ecological, since it uses annual aggregates reconstructed from denominators and reported percentages (Supplementary Table S5), and it is intended as a diagnostic of the analysis window rather than a competing estimate.

3. Results

3.1 Corpus composition and overall prevalence

The subcorpus is concentrated in recent years. Mentions dated 2020 and 2021 together make up 31.3% of the total, and no year before 2001 contributes as many as 200 mentions. Only 41 mentions carry a 2022 date, which reflects the timing of the corpus snapshot rather than a decline in publishing. Year-to-year counts are also irregular in places (for example, 6,470 mentions in 2004 but 262 in 2005; Supplementary Table S5), which is consistent with the position-based selection rule (Section 2.2).

The distribution across software types is highly skewed. Domain-specific tools account for 86.2% of mentions and statistical packages for 8.7%. The remaining four categories together account for less

than 5%, and two are small in absolute terms: machine-learning frameworks contribute 1,269 mentions and reference managers 731. These small denominators widen the intervals reported in Section 3.3.

Citation marking is uncommon. It is present in 85,279 mentions (10.7%). Most mentions (69.7%) consist of a bare name in running text, and a further 16.3% give the name together with a dotted number such as a version. Developer attributions (2.4%) and URLs (0.9%) are rare. Close to nine mentions in ten therefore appear without any of the apparatus that a reference-list citation leaves in the sentence.

3.2 Mention style by software type (RQ1)

Mention style varies with software type (Table 1; Fig. 2). The full contingency table departs strongly from independence, $\chi^2(20) = 153{,}217.3$, $p < .001$, Cramér's V = 0.219. Restricting the comparison to citation-marked versus all other styles gives a weaker association, $\chi^2(5) = 6{,}523.6$, $p < .001$, V = 0.090. At this sample size, the chi-square statistics carry little information, and we base our interpretation on the V values. Programming languages (15.8%) and machine-learning frameworks (18.0%) are marked more often than the corpus average of 10.7%; domain-specific tools are close to it (11.4%); and GIS and visualisation tools (7.8%), statistical packages (2.2%) and reference managers (1.0%) are marked less often.

Table 1
Mention style by software type: counts and row percentages.

Software type	Citation-marked	Name + version	Developer attr.	URL only	In-text only	Total
Statistical package	1,561 (2.2)	41,521 (59.5)	7,857 (11.3)	121 (0.2)	18,726 (26.8)	69,786
Programming language	3,864 (15.8)	8,122 (33.2)	464 (1.9)	688 (2.8)	11,339 (46.3)	24,477
ML framework	228 (18.0)	184 (14.5)	4 (0.3)	20 (1.6)	833 (65.6)	1,269
GIS/visualization	1,077 (7.8)	2,417 (17.4)	1,149 (8.3)	158 (1.1)	9,077 (65.4)	13,878
Reference manager	7 (1.0)	121 (16.6)	21 (2.9)	7 (1.0)	575 (78.7)	731
Domain-specific tool	78,542 (11.4)	77,916 (11.3)	9,555 (1.4)	6,331 (0.9)	517,271 (75.0)	689,615
All mentions	**85,279 (10.7)**	**130,281 (16.3)**	**19,050 (2.4)**	**7,325 (0.9)**	**557,821 (69.7)**	**799,756**

Note. Row percentages in parentheses. Styles are mutually exclusive and assigned in the priority order shown (left to right); rows sum to 100%. 'Name + version' denotes a sentence containing a dotted number, which may be a version string or another decimal quantity (Section 2.4).

Two categories stand out. Statistical packages have the lowest marking rate of any named category (2.2%) and the highest rate of the name-with-version style (59.5%). Machine-learning frameworks show

the reverse pattern: 18.0% marked and 14.5% with a dotted number. Programming languages fall between the two on both measures (15.8% and 33.2%). Reference managers are almost never marked (1.0%) and are given as a bare name in 78.7% of cases, although with 731 mentions, of which 7 are marked, this estimate is imprecise. The high name-with-version rate for statistical packages should be interpreted cautiously. Sentences that name statistical software often report *p*-values and other decimal statistics, which the version pattern also matches (Section 2.4), so part of this rate may reflect reported statistics rather than version numbers. The low marking rate for statistical packages does not depend on the version pattern, because the citation-marked style takes priority in the classification.

3.3 Mention-level correlates of citation marking (RQ2)

Table 2 reports the fitted model with paper-clustered standard errors, and Fig. 3 displays the odds ratios. Relative to domain-specific tools, the adjusted odds of citation marking are higher for machine-learning frameworks (OR = 1.70, 95% CI [1.38, 2.09]) and programming languages (OR = 1.47, [1.40, 1.55]), and lower for GIS and visualization tools (OR = 0.70, [0.65, 0.76]) and, more markedly, for statistical packages (OR = 0.23, [0.22, 0.25]). Reference managers yield the most extreme estimate (OR = 0.08, [0.04, 0.16]), but it rests on 7 marked mentions, and little weight should be placed on the point value.

The three structural markers do not behave as a group. Dotted numbers (OR = 1.11, [1.07, 1.15]) and URLs (OR = 1.28, [1.19, 1.38]) are associated with somewhat higher odds of marking, so a mention that already carries a URL is more, not less, likely also to carry a citation marker. Developer attribution is associated with substantially lower odds (OR = 0.29, [0.27, 0.32]). This is an association within sentences, not evidence that naming a developer leads authors to omit a citation. It is consistent with the substitution interpretation discussed in Section 4.3, but equally consistent with confounding by discipline, journal, or tool, none of which the model observes.

Table 2
Logistic regression of citation marking on mention-level covariates (N = 799,748), with standard errors clustered by paper.

Covariate	*b*	OR	SE (clustered)	95% CI for OR	*p*
Intercept	−2.0510	0.129	0.0099	[0.126, 0.131]	< 0.001
Type: GIS / visualization	−0.3550	0.701	0.0403	[0.648, 0.759]	< 0.001
Type: programming language	0.3856	1.470	0.0258	[1.398, 1.547]	< 0.001
Type: ML framework	0.5294	1.698	0.1047	[1.383, 2.085]	< 0.001
Type: reference manager	−2.5654	0.077	0.3805	[0.036, 0.162]	< 0.001
Type: statistical package	−1.4555	0.233	0.0341	[0.218, 0.249]	< 0.001
Year (centered, per year)	−0.0047	0.995	0.0021	[0.991, 0.999]	0.022
Has version string	0.1033	1.109	0.0188	[1.069, 1.150]	< 0.001
Has URL	0.2492	1.283	0.0387	[1.189, 1.384]	< 0.001
Has developer attribution	−1.2267	0.293	0.0474	[0.267, 0.322]	< 0.001

Note. Reference category: domain-specific tool. Confidence intervals are Wald intervals computed as exp(b ± 1.96 × clustered SE). Model-based standard errors and the corresponding intervals are given in Supplementary Table S3; no conclusion differs between the two. Likelihood-ratio test against the intercept-only model: G = 10,849.19, df = 9, p < .001. Nagelkerke R^2 = 0.027; McFadden R^2 = 0.020; Cox–Snell R^2 = 0.014.

Table 3 expresses the same estimates as model-implied probabilities. With the structural markers absent and year at its mean, the predicted probability of marking ranges from 1.0% for a reference manager to 17.9% for a machine-learning framework. The markers extend this range: a machine-learning framework mentioned with a dotted number and a URL reaches 23.7%, whereas a reference manager with a developer attribution falls to 0.3%. Against the reference profile (a domain-specific tool with no structural marker, 11.4%), the risk difference is − 8.5 percentage points for statistical packages, + 4.5 points for programming languages, and − 7.8 points for developer attribution. These are point predictions. Interval estimates for them require the full coefficient covariance matrix, which was not retained with the exported results, so the values in Table 3 are illustrative; the very small predicted probabilities in particular should not be over-interpreted.

Table 3
Model-implied probability of citation marking for illustrative covariate profiles (year at its mean).

Covariate profile	Predicted probability
Reference manager, developer attribution present	0.003
Statistical package, developer attribution present	0.009
Reference manager, no structural markers	0.010
Statistical package, no structural markers	0.029
Domain-specific tool, developer attribution present	0.036
GIS/visualization tool, no structural markers	0.083
Domain-specific tool, no structural markers (reference profile)	0.114
Programming language, no structural markers	0.159
Machine-learning framework, no structural markers	0.179
Machine-learning framework, version string and URL present	0.237

Note. Point predictions obtained by inverse-logit transformation of the linear predictor in Table 2. Interval estimates are not reported because the coefficient covariance matrix was not retained.

The model improves clearly on the intercept-only specification (G = 10,849.19, df = 9, $p < .001$) but explains little of the variation in individual outcomes (Nagelkerke R^2 = 0.027; McFadden R^2 = 0.020; Cox–Snell R^2 = 0.014). With N close to 800,000, the p-values in Table 2 say little about practical importance, and we do not rely on them. Part of the unexplained variation is attributable to measurement: the outcome is generated at the level of the sentence, whereas the covariates describe the mention, and the denominator includes non-software strings whose marking the model cannot be expected to explain.

Clustering by paper inflated the standard errors by factors between 1.00 and 2.62 (Supplementary Table S3) and changed no conclusion; the year term is the least secure (p = .022). Excluding SPSS, the single most frequent software string and the dominant member of the statistical category, left eight of the nine slope coefficients unchanged to within 1% but moved the statistical-package odds ratio from 0.233 to 0.303, an increase of 30% (Supplementary Table S4). The direction of the statistical-package estimate therefore survives the exclusion, but its magnitude depends substantially on one product, and we treat the estimate as indicative rather than precise. Equivalent leave-one-out checks were not run for other categories.

We draw no conclusion about change over time, because the estimate depends on where the analysis window begins (Fig. 4). Marking rates peak in the sparsely populated early 2000s (24.95% in 2004, on 6,470 mentions) and lie between 7.65% and 12.27% from 2008 onwards, once annual volumes exceed 28,000. Over the full window the adjusted coefficient is slightly negative (OR = 0.995 per year). An unadjusted analysis of the 2008–2021 aggregates reverse the sign (OR = 1.014 per year, 95% CI [1.012,

1.016]; n = 776,400 mentions in 14 annual points), and the annual rates show no monotone trend (Spearman ρ = 0.420, p = .135). The years before 2008 contain about 3% of the mentions but exert most of the leverage on the full-window estimate. Given this disagreement, we do not treat the full-window year coefficient as evidence of a trend.

3.4 Where residual ambiguity lies (RQ3)

The curated labels produced an unexpected result. Each of the six names selected in advance as likely to be ambiguous is labelled as software in every curated instance: R (12,499 instances), Prism (1,887), Python (721), SPARK (58), Orange (21) and Comet (2). Five further candidates, including ImageJ (8,553), Excel (6,616) and MATLAB (3,686), behave in the same way, and SAS does not occur in the subcorpus at all (Supplementary Table S2). Whatever ambiguity these names carry in free text is not visible in the curated labels; it has either been resolved during extraction and curation or is not represented by labels assigned in this way (Section 5).

The registry did not cover the terms that do show label variation. None of the eight terms selected by the fallback rule appears in the registry, so coverage is 0 of 8. Approach A therefore classifies all 1,961 mentions as not software and returns zero precision, recall and F1 for every term (Table 4). This is a statement about coverage rather than a performance comparison: a registry compiled by listing known tools is unlikely to contain everyday words that also happen to be tool names, and the ΔF1 values should be read as distance from an approach with no coverage, not as the improvement available over a maintained production linker.

Approach B performs moderately in aggregate and unevenly by term. Across the 1,961 mentions, it reaches a precision of 0.548, a recall of 0.642 and an F1 of 0.591. For comparison, the best transformer-based joint NER and RE systems in the SOMD 2025 shared task reached a macro-F1 of about 0.89 on in-distribution and about 0.63 on out-of-distribution test data (Upadhyaya et al., 2025). The tasks differ (full extraction with relations versus binary disambiguation of pre-extracted strings), so the figures are not directly comparable, but the aggregate F1 reported here is clearly below what a fine-tuned model would be expected to achieve, and we present it as a transparent baseline. Per-term F1 ranges from 0.116 to 0.895. The rules perform well on cluster (0.895), star (0.829; precision 0.941) and zoom (0.733; precision 0.966), and poorly on string (0.116; precision 0.064), quest (0.250) and oncomine (0.273). Precision varies far more across terms (0.064–0.966) than recall (0.424–0.895): where the rules fail, they mostly do so by accepting non-software uses that occur in technical biomedical prose. Approximate randomization indicates that Approach B outperforms the zero-coverage registry for seven of the eight terms; the exception is quest (p = .113, 144 mentions).

Table 4
Entity-disambiguation diagnostic: registry matching (A) versus rule-based context (B).

Term	n	P (A)	R (A)	F1 (A)	P (B)	R (B)	F1 (B)	ΔF1	p
zoom	556	0.000	0.000	0.000	0.966	0.591	0.733	0.733	< 0.001
string	519	0.000	0.000	0.000	0.064	0.639	0.116	0.116	0.005
star	248	0.000	0.000	0.000	0.941	0.740	0.829	0.829	< 0.001
oncomine	161	0.000	0.000	0.000	0.200	0.429	0.273	0.273	< 0.001
quest	144	0.000	0.000	0.000	0.158	0.600	0.250	0.250	0.113
cluster	119	0.000	0.000	0.000	0.895	0.895	0.895	0.895	< 0.001
prime	113	0.000	0.000	0.000	0.333	0.742	0.460	0.460	< 0.001
prosite	101	0.000	0.000	0.000	0.610	0.424	0.500	0.500	< 0.001
Overall	**1,961**	**0.000**	**0.000**	**0.000**	**0.548**	**0.642**	**0.591**	**0.591**	**—**

Note. P = precision; R = recall. p-values from approximate randomisation of ΔF1 within term. Registry coverage was 0 of 8 terms, which fixes all Approach A metrics at zero by construction; ΔF1 therefore equals F1(B).

4. Discussion

4.1 The level of citation marking

A sentence-level citation-marking rate of 10.7% is consistent with the picture of low formal software citation established by Howison and Bullard (2016) and later large-corpus studies (Pan et al., 2016; Schindler et al., 2022a). It is, however, well below the 31–43% formal-citation share that Howison and Bullard (2016) reported from hand-coding 90 biology articles, and three differences contribute, in proportions we cannot separate. The first is the construct: their coders credited a citation wherever in the article it appeared, whereas our instrument counts only markers in the same sentence (Section 2.3). The second is the corpus: 90 selected articles from one discipline, compared with mentions from 138,019 machine-processed biomedical papers. The third is measurement noise: non-software strings in the denominator and pattern errors in both directions. Because these sources act in different directions, the 10.7% figure should not be read as an estimate of reference-list citation, nor as evidence of decline; our own temporal analysis provides no support for a trend.

The more informative result is that the rate is not uniform. Unadjusted marking rates differ by more than an order of magnitude between the least- and most-marked tool classes, and the association between

software type and mention style is of moderate strength (V = 0.219). If this pattern holds under validated measurement, under-citation of software is better described as a set of such conventions than as a uniform cultural deficit, and category-specific conventions lend themselves to targeted intervention in a way that a general norm does not.

4.2 Differences between classes of tool

Statistical packages are citation-marked in 2.2% of mentions and have about a quarter of the adjusted odds of marking compared a domain-specific tool. This finding is consistent with work on individual general-purpose analytical tools, where citation counts recover only a fraction of documented use (Li et al., 2017, 2019). One interpretation is that statistical packages are conventionally described as apparatus rather than cited as works: an author who reports that an analysis was performed in a named package is following methodological reporting conventions rather than citation conventions. If so, interventions aimed at better methodological reporting alone would not close the credit gap for statistical software. Two caveats apply. The high rate of dotted numbers in sentences naming statistical packages may partly reflect reported statistics rather than version numbers (Section 3.2), and the magnitude of the statistical-package estimate depends substantially on SPSS (Section 3.3).

Machine-learning frameworks show the opposite profile: the highest marking rate and a low rate of dotted numbers. This is consistent with a norm inherited from computer science, in which the canonical representation of a framework is a paper. Disciplinary variation of this kind is well documented (Hwang et al., 2017; Pan et al., 2016), and evidence from the CORD-19 corpus suggests that compliance with software-citation practice rises fastest where a citable object already exists (Du et al., 2022). A plausible reading is that when a citable paper is the accepted representation of a tool, citation tends to follow, and when the tool is treated as equipment, it tends not to. Because the corpus lacks field metadata, part of this contrast may reflect the disciplines in which each class of tool is used rather than the tool class itself.

These class-based findings rest on rule-based instruments that have not been validated against human annotation, and they would be strengthened by replication with a validated, transformer-based extractor (Upadhyaya et al., 2025; Rastogi & Tiwari, 2025; Pride et al., 2025). The contrasts hold to the extent that measurement error does not fall disproportionately on particular classes, an assumption we state rather than verify.

4.3 Developer attribution and resolvable identifiers

Parenthetical naming of a developer is associated with 71% lower odds of citation marking; the corresponding predicted probability is 3.6%, compared with 11.4% for the reference profile. One interpretation is that in-text attribution functions as an alternative way of acknowledging the tool, in place of a citation rather than as a step towards it. Developer and URL attributes are recorded as distinct relations in gold-standard corpora (Schindler et al., 2021) and have been reported at different completeness rates in earlier content analyses (Howison & Bullard, 2016; Schindler et al., 2022a), but their opposite associations with citation marking have not, to our knowledge, been quantified before.

URLs and dotted numbers point the other way: markers that make a mention machine-actionable tend to co-occur with citation marking, whereas the marker that acknowledges a vendor tends to appear without one. Because the developer pattern detects mainly corporate designators (Section 2.4), this finding concerns vendor-style attribution specifically. The interpretation is associational, but it suggests that two editorial requests often treated as interchangeable may differ in practice: asking authors to name a producer is associated with a mention form in which citation is rare, whereas asking for a resolvable identifier is associated with one in which it is more common.

4.4 Statistical significance and explanatory power

The model is highly significant yet explains only 2.7% of outcome variation (Nagelkerke R^2). This combination does not make the findings unimportant, but it does define their scope. Pseudo-R^2 measures the prediction of individual outcomes, whereas the object of inference here is the difference in marking rates between classes of tool. Whether an author places a citation in a particular sentence depends on many factors that the available mention-level covariates cannot capture, so a low ceiling on explanatory power is built into the data. Within that ceiling, the covariates still span predicted probabilities from 0.3% to 23.7% around a base rate of 10.7%. The model therefore identifies systematic differences within a phenomenon that remains largely idiosyncratic at the level of the individual mention.

4.5 Implications for extraction tools

Two findings bear on tool development. The first concerns coverage. None of the eight names that the curated data identify as ambiguous appears in a registry assembled from a conventional tool lexicon. Registries list things that people recognise as tools, whereas the names that require disambiguation in these data are everyday words, such as cluster, star, string and prime. A pipeline that resolves entities through registry membership alone is therefore poorly placed to handle its hardest cases, which suggests that a resolution step independent of registry membership is needed. This is consistent with conclusions reached for bioinformatics resource names (Duck et al., 2015; Ozyurt et al., 2016) and with the general entity-linking literature, in which candidate generation and disambiguation are treated as separate stages (Shen et al., 2015).

The second concerns contextual performance. Simple context rules achieve an aggregate F1 of 0.591, with per-term precision ranging from 0.064 to 0.966. Measured against the 2025 shared tasks (Upadhyaya et al., 2025; Rastogi & Tiwari, 2025), this is low, and we treat it as a baseline rather than as support for lexicon methods. The failures are systematic: precision collapses when the competing sense is itself technical, because generic computational vocabulary cannot separate two senses that both occur in technical prose. Lexical context may therefore serve as an inexpensive first pass whose failure cases can be anticipated, and the terms on which it fails are those for which embedding-based resolution (Beltagy et al., 2019) or document-level recognition (Lopez et al., 2021) is most likely to justify its cost. More broadly, the result suggests that disambiguation benchmarks should be built from observed label variation rather than from names that merely appear ambiguous, subject to the caveat about label granularity discussed in Section 5.

4.6 Tentative implications for policy

Given the associational design and the unvalidated instruments, the following implications are offered as hypotheses for further study rather than as recommendations. First, if the class differences are confirmed, journals could target software-citation requirements where the deficit appears largest, for example by asking that statistical and other general-purpose analytical software be cited to a persistent identifier. Citation requirements will not capture every form of software contribution: the finding that code contributors are often not listed as authors (Brown et al., 2025) points to a layer of unrecognised work beneath the mention level analysed here, so citation requirements and contribution mapping are complementary. Second, evidence on journal policies suggests that requirements are only partially effective and that their design matters (Stodden et al., 2018; Katz et al., 2021). Third, funders and evaluators should recognise that citation counts substantially undercount software use (Howison et al., 2015; Nesbitt et al., 2024). Fourth, indexers and editors could prioritise resolvable identifiers over vendor naming, in line with the FAIR principles for research software and with archival referencing of source code (Barker et al., 2022; Di Cosmo et al., 2020).

5. Limitations

Sampling. The subcorpus is a deterministic, position-based selection of the first 400,000 records of two source tables, not a probability sample. File order was not audited, and irregularities in the annual counts (Section 3.1) suggest that position may be associated with publication date or source. We therefore make no claim that 10.7% is the citation-marking rate of the CZI dataset, of PubMed Central or of the biomedical literature. The two tables also differ in name normalisation (one disambiguated, one raw), which may affect lexicon matching, and the corpus subset was not included as a covariate.

Non-software strings. In the subcorpus, 15.0% of mentions with a definite label are labelled as not software, and a third of mentions carry no label. The prevalence and regression analyses were not restricted to mentions labelled as software, so absolute rates include an unknown share of non-software strings, and the class contrasts are unbiased only if this noise is spread evenly across classes, which is not guaranteed. A refit restricted to the 431,375 mentions labelled as software is the most direct check and was not performed.

Measurement. The outcome is sentence-level citation marking, not reference-list citation, and the classification instruments have not been validated against human annotation; no inter-rater coefficient of the kind reported for hand-built corpora is available (Du et al., 2021; Schindler et al., 2021). Specific known weaknesses are: the 200-character text limit, which can truncate citation markers; the superscript and parenthetical-numeral patterns, which also match unit exponents and enumerations; the version pattern, which also matches decimal quantities such as *p*-values; and the developer pattern, which under-detects non-commercial developers. Some of these errors may differ between software classes. For example, if sentences reporting statistical analyses contain citation markers for cited methods at a

different rate from sentences reporting other tools, the statistical-package deficit could be larger or smaller than measured. We therefore regard it as a substantial deficit of uncertain magnitude.

Software-type lexicon. The taxonomy is coarse, and 86.2% of mentions fall into the residual domain-specific category, which also serves as the regression reference; contrasts are therefore made against a heterogeneous mixture. The static lexicon matches exact and word-boundary strings only, so misspellings, aliases, and newer tools fall into the residual category, and several entries are themselves ambiguous (for example, 'papers', 'origin', 'spark', and 'transformers'). Some assignments are debatable, such as the inclusion of image-analysis tools and plotting libraries in the GIS and visualisation category. SAS, although listed in the statistical lexicon, does not occur in the subcorpus; we could not determine whether this reflects the extractor, the source tables, or the position-based selection, but it implies that the statistical category is incompletely populated.

Disambiguation diagnostic. The curation labels of the CZI dataset may have been assigned to software strings rather than to individual mentions in context (Istrate et al., 2022). If so, label variation within a term may partly reflect differences in case or surface form (for example 'STAR' versus 'star') rather than in contextual sense, and the absence of negative labels for names such as R and Python would follow from how labels were assigned rather than from the absence of ambiguity. The eight diagnostic terms were selected conditional on label variation and are not representative of ambiguous names in general; the minimum of 30 labelled mentions is a heuristic rather than the result of a power analysis. Approach A has zero coverage of these terms by construction, Approach B's competing-sense lexicon was not tailored to them, and no majority-class baseline (classifying every mention as software) was reported, so the diagnostic cannot establish how much of Approach B's performance is due to context rather than to the base rate of software uses.

Confounding and model reporting. Journal policy, discipline, publisher, funding, and author characteristics are unobserved, and each is known to be associated with citation practice (Hwang et al., 2017; Pan et al., 2016; Stodden et al., 2018). The coefficients are therefore marginal with respect to these factors and are not causal effects; a tool-class contrast may partly reflect the disciplinary composition of the papers in which that class is used. Paper-clustered inference addresses dependence between mentions in the same paper but not confounding, and mentions of the same software string across papers were not modelled as a further source of dependence. The coefficient covariance matrix was not retained, so the predicted probabilities in Table 3 lack intervals, and information criteria and out-of-sample checks were not computed. The leave-one-out check was carried out only for SPSS.

Scope. The corpus is biomedical, ends in 2022, and may inherit systematic biases of the extraction model that produced it. No finding should be extended to other fields without replication. Several elements of the original study plan, including bibliographic linkage, field, and venue comparisons and dual-coder validation, were not carried out; these departures are documented in Supplementary Table S1. A natural next stage is deterministic linkage to an open bibliographic index by DOI or PubMed identifier (Priem et al., 2022), whose reference coverage is now comparable to that of proprietary databases

(Culbert et al., 2025) and whose topic classification would supply the field variable missing here, combined with a multilevel model with venue-level policy indicators and validation of the instruments against the SoMeSci and Softcite gold standards.

6. Conclusion

In a defined subcorpus of 799,756 software mentions from the biomedical literature, about one mention in ten carries a citation marker in the same sentence, and seven in ten consist of a bare name in running text. This sentence-level figure is lower than earlier hand-coded estimates of reference-list citation, largely because it measures something narrower. The deficit is not uniform: it is greatest for general-purpose statistical packages and smallest for machine-learning frameworks and programming languages, which more often have an established citable reference. In-text naming of a developer co-occurs with the absence of a citation marker, whereas URLs co-occur with its presence, and the subcorpus provides no consistent evidence of change over time. A conventional registry contained none of the names that the curated data identify as ambiguous, and simple context rules provided only a modest baseline. Because the instruments are unvalidated and the subcorpus is not a probability sample, these findings are exploratory. If they are confirmed with validated instruments and linked bibliographic data, they would suggest that a general appeal to cite software (Smith et al., 2016) is less likely to change practice for everyday analytical tools than category-specific requirements tied to persistent identifiers (Barker et al., 2022; Katz et al., 2021).

Declarations

Ethics approval.

This study analyzed openly licensed, previously published bibliographic metadata and text-mined software mentions. No human participants were involved, and ethics approval was not required.

Competing interests.

The author declare no competing interests.

Use of Generative AI.

Generative AI tools were used for language editing of the manuscript. The author reviewed all text and take full responsibility for the content.

Funding.

This research received no specific grant from any funding agency in the public, commercial or not-for-profit sectors.

Author Contribution

M. M. Billa. is the sole author and was responsible for the conception and design of the study, data curation, analysis, preparation of figures and tables, and writing of the manuscript.

Data Availability

The CZI Software Mentions dataset is openly available from Dryad (https://doi.org/10.5061/dryad.6wwpzgn2c). The analysis code, derived tables, figure data and full instrument specification (Supplementary Table S6) will be deposited at Harvard Dataverse.

References

1. Baker, M. (2016). 1,500 scientists lift the lid on reproducibility. Nature, 533(7604), 452–454. https://doi.org/10.1038/533452a
2. Bandrowski, A. E., & Martone, M. E. (2016). RRIDs: A simple step toward improving reproducibility through rigor and transparency of experimental methods. Neuron, 90(3), 434–436. https://doi.org/10.1016/j.neuron.2016.04.030
3. Barker, M., Chue Hong, N. P., Katz, D. S., Lamprecht, A.-L., Martinez-Ortiz, C., Psomopoulos, F., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., & Honeyman, T. (2022). Introducing the FAIR Principles for research software. Scientific Data, 9, 622. https://doi.org/10.1038/s41597-022-01710-x
4. Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (pp. 3615–3620). Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1371
5. Brown, E. M., Slaughter, I., & Weber, N. (2025). Code contribution and credit in science [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2510.16242
6. Cameron, A. C., & Miller, D. L. (2015). A practitioner’s guide to cluster-robust inference. Journal of Human Resources, 50(2), 317–372. https://doi.org/10.3368/jhr.50.2.317
7. Culbert, J. H., Hobert, A., Jahn, N., Haupka, N., Schmidt, M., Donner, P., & Mayr, P. (2025). Reference coverage analysis of OpenAlex compared to Web of Science and Scopus. Scientometrics, 130(4), 2475–2492. https://doi.org/10.1007/s11192-025-05293-3
8. Di Cosmo, R., Gruenpeter, M., & Zacchiroli, S. (2020). Referencing source code artifacts: A separate concern in software citation. Computing in Science & Engineering, 22(2), 33–43.

https://doi.org/10.1109/MCSE.2019.2963148

9. Dror, R., Baumer, G., Shlomov, S., & Reichart, R. (2018). The hitchhiker's guide to testing statistical significance in natural language processing. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Vol. 1, pp. 1383–1392). Association for Computational Linguistics. https://doi.org/10.18653/v1/P18-1128
10. Druskat, S., Chue Hong, N. P., Buzzard, S., Konovalov, O., & Kornek, P. (2024). Don't mention it: An approach to assess challenges to using software mentions for citation and discoverability research [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2402.14602
11. Du, C., Cohoon, J., Lopez, P., & Howison, J. (2021). Softcite dataset: A dataset of software mentions in biomedical and economic research publications. Journal of the Association for Information Science and Technology, 72(7), 870–884. https://doi.org/10.1002/asi.24454
12. Du, C., Cohoon, J., Lopez, P., & Howison, J. (2022). Understanding progress in software citation: A study of software citation in the CORD-19 corpus. PeerJ Computer Science, 8, e1022. https://doi.org/10.7717/peerj-cs.1022
13. Duck, G., Kovacevic, A., Robertson, D. L., Stevens, R., & Nenadic, G. (2015). Ambiguity and variability of database and software names in bioinformatics. Journal of Biomedical Semantics, 6, 29. https://doi.org/10.1186/s13326-015-0026-0
14. Duck, G., Nenadic, G., Brass, A., Robertson, D. L., & Stevens, R. (2014). Extracting patterns of database and software usage from the bioinformatics literature. Bioinformatics, 30(17), i601–i608. https://doi.org/10.1093/bioinformatics/btu471
15. Duck, G., Nenadic, G., Filannino, M., Brass, A., Robertson, D. L., & Stevens, R. (2016). A survey of bioinformatics database and software usage through mining the literature. PLOS ONE, 11(6), e0157989. https://doi.org/10.1371/journal.pone.0157989
16. Hannay, J. E., MacLeod, C., Singer, J., Langtangen, H. P., Pfahl, D., & Wilson, G. (2009). How do scientists develop and use scientific software? In 2009 ICSE Workshop on Software Engineering for Computational Science and Engineering (pp. 1–8). IEEE. https://doi.org/10.1109/SECSE.2009.5069155
17. Howison, J., & Bullard, J. (2016). Software in the scientific literature: Problems with seeing, finding, and using software mentioned in the biology literature. Journal of the Association for Information Science and Technology, 67(9), 2137–2155. https://doi.org/10.1002/asi.23538
18. Howison, J., Deelman, E., McLennan, M. J., Ferreira da Silva, R., & Herbsleb, J. D. (2015). Understanding the scientific software ecosystem and its impact: Current and future measures. Research Evaluation, 24(4), 454–470. https://doi.org/10.1093/reseval/rvv014
19. Howison, J., & Herbsleb, J. D. (2011). Scientific software production: Incentives and collaboration. In Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work (pp. 513–522). Association for Computing Machinery. https://doi.org/10.1145/1958824.1958904
20. Hwang, L., Fish, A., Soito, L., Smith, M., & Kellogg, L. H. (2017). Software and the scientist: Coding and citation practices in geodynamics. Earth and Space Science, 4(11), 670–680.

https://doi.org/10.1002/2016EA000225

21. Istrate, A.-M., Li, D., Taraborelli, D., Torkar, M., Veytsman, B., & Williams, I. (2022). A large dataset of software mentions in the biomedical literature [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2209.00693
22. Jurek, A. M., Greenland, S., Maldonado, G., & Church, T. R. (2005). Proper interpretation of non-differential misclassification effects: Expectations vs observations. International Journal of Epidemiology, 34(3), 680–687. https://doi.org/10.1093/ije/dyi060
23. Katz, D. S., Chue Hong, N. P., Clark, T., Muench, A., Stall, S., Bouquin, D., Cannon, M., Edmunds, S., Faez, T., Feeney, P., Fenner, M., Friedman, M., Grenier, G., Harrison, M., Heber, J., Leary, A., MacCallum, C., Murray, H., Pastrana, E., … Yeston, J. (2021). Recognizing the value of software: A software citation guide [version 2]. F1000Research, 9, 1257. https://doi.org/10.12688/f1000research.26932.2
24. Krüger, F., & Schindler, D. (2020). A literature review on methods for the extraction of usage statements of software and data. Computing in Science & Engineering, 22(1), 26–38. https://doi.org/10.1109/MCSE.2019.2943847
25. Li, K., Chen, P.-Y., & Yan, E. (2019). Challenges of measuring software impact through citations: An examination of the lme4 R package. Journal of Informetrics, 13(1), 449–461. https://doi.org/10.1016/j.joi.2019.02.007
26. Li, K., Lin, X., & Greenberg, J. (2016). Software citation, reuse and metadata considerations: An exploratory study examining LAMMPS. Proceedings of the Association for Information Science and Technology, 53(1), 1–10. https://doi.org/10.1002/pra2.2016.14505301072
27. Li, K., Yan, E., & Feng, Y. (2017). How is R cited in research outputs? Structure, impacts, and citation standard. Journal of Informetrics, 11(4), 989–1002. https://doi.org/10.1016/j.joi.2017.08.003
28. Lopez, P., Du, C., Cohoon, J., Ram, K., & Howison, J. (2021). Mining software entities in scientific literature: Document-level NER for an extremely imbalanced and large-scale task. In Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21). Association for Computing Machinery. https://doi.org/10.1145/3459637.3481936
29. Nagelkerke, N. J. D. (1991). A note on a general definition of the coefficient of determination. Biometrika, 78(3), 691–692. https://doi.org/10.1093/biomet/78.3.691
30. Nesbitt, A., Veytsman, B., Mietchen, D., Brown, E. M., Howison, J., Pimentel, J. F., Hébert-Dufresne, L., & Druskat, S. (2024). Biomedical open source software: Crucial packages and hidden heroes [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2404.06672
31. Ozyurt, I. B., Grethe, J. S., Martone, M. E., & Bandrowski, A. E. (2016). Resource disambiguator for the web: Extracting biomedical resources and their citations from the scientific literature. PLOS ONE, 11(1), e0146300. https://doi.org/10.1371/journal.pone.0146300
32. Pan, X., Yan, E., Cui, M., & Hua, W. (2018). Examining the usage, citation, and diffusion patterns of bibliometric mapping software: A comparative study of three tools. Journal of Informetrics, 12(2), 481–493. https://doi.org/10.1016/j.joi.2018.03.005

33. Pan, X., Yan, E., & Hua, W. (2016). Disciplinary differences of software use and impact in scientific literature. Scientometrics, 109(3), 1593–1610. https://doi.org/10.1007/s11192-016-2138-4
34. Pan, X., Yan, E., Wang, Q., & Hua, W. (2015). Assessing the impact of software on science: A bootstrapped learning of software entities in full-text papers. Journal of Informetrics, 9(4), 860–871. https://doi.org/10.1016/j.joi.2015.07.012
35. Park, H., & Wolfram, D. (2019). Research software citation in the Data Citation Index: Current practices and implications for research software sharing and reuse. Journal of Informetrics, 13(2), 574–582. https://doi.org/10.1016/j.joi.2019.03.005
36. Prabhu, P., Jablin, T. B., Raman, A., Zhang, Y., Huang, J., Kim, H., Johnson, N. P., Liu, F., Ghosh, S., Beard, S., Oh, T., Zoufaly, M., Walker, D., & August, D. I. (2011). A survey of the practice of computational science. In State of the Practice Reports (SC '11). Association for Computing Machinery. https://doi.org/10.1145/2063348.2063374
37. Pride, D., Guenci, M., Docekal, M., Peroni, S., & Knoth, P. (2025). Identifying and classifying software mentions in full text scholarly documents. In Proceedings of the 25th ACM/IEEE Joint Conference on Digital Libraries (JCDL 2025). IEEE. https://doi.org/10.1109/JCDL67857.2025.00041
38. Priem, J., Piwowar, H., & Orr, R. (2022). OpenAlex: A fully-open index of scholarly works, authors, venues, institutions, and concepts [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2205.01833
39. Rastogi, P., & Tiwari, R. (2025). Extracting software mentions and relations using transformers and LLM-generated synthetic data at SOMD 2025. In Proceedings of the Fifth Workshop on Scholarly Document Processing (SDP 2025) (pp. 173–181). Association for Computational Linguistics. https://doi.org/10.18653/v1/2025.sdp-1.17
40. Schindler, D., Bensmann, F., Dietze, S., & Krüger, F. (2021). SoMeSci—A 5 star open data gold standard knowledge graph of software mentions in scientific articles. In Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21) (pp. 4574–4583). Association for Computing Machinery. https://doi.org/10.1145/3459637.3482017
41. Schindler, D., Bensmann, F., Dietze, S., & Krüger, F. (2022a). The role of software in science: A knowledge graph-based analysis of software mentions in PubMed Central. PeerJ Computer Science, 8, e835. https://doi.org/10.7717/peerj-cs.835
42. Schindler, D., Zapilko, B., & Krüger, F. (2022b). SoftwareKG—A knowledge graph of software mentions from biomedical literature. Semantic Web, 14(1), 73–95. https://doi.org/10.3233/SW-223172
43. Shen, W., Wang, J., & Han, J. (2015). Entity linking with a knowledge base: Issues, techniques, and solutions. IEEE Transactions on Knowledge and Data Engineering, 27(2), 443–460. https://doi.org/10.1109/TKDE.2014.2327028
44. Singh Chawla, D. (2016). The unsung heroes of scientific software. Nature, 529(7584), 115–116. https://doi.org/10.1038/529115a
45. Smith, A. M., Katz, D. S., & Niemeyer, K. E. (2016). Software citation principles. PeerJ Computer Science, 2, e86. https://doi.org/10.7717/peerj-cs.86

46. Stodden, V., Seiler, J., & Ma, Z. (2018). An empirical analysis of journal policy effectiveness for computational reproducibility. Proceedings of the National Academy of Sciences, 115(11), 2584–2589. https://doi.org/10.1073/pnas.1708290115
47. Upadhyaya, S., Otto, W., Krüger, F., & Dietze, S. (2025). SOMD2025: A challenging shared task for software related information extraction. In Proceedings of the Fifth Workshop on Scholarly Document Processing (SDP 2025) (pp. 137–145). Association for Computational Linguistics. https://doi.org/10.18653/v1/2025.sdp-1.13
48. Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., … Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. Scientific Data, 3, 160018. https://doi.org/10.1038/sdata.2016.18

Figures

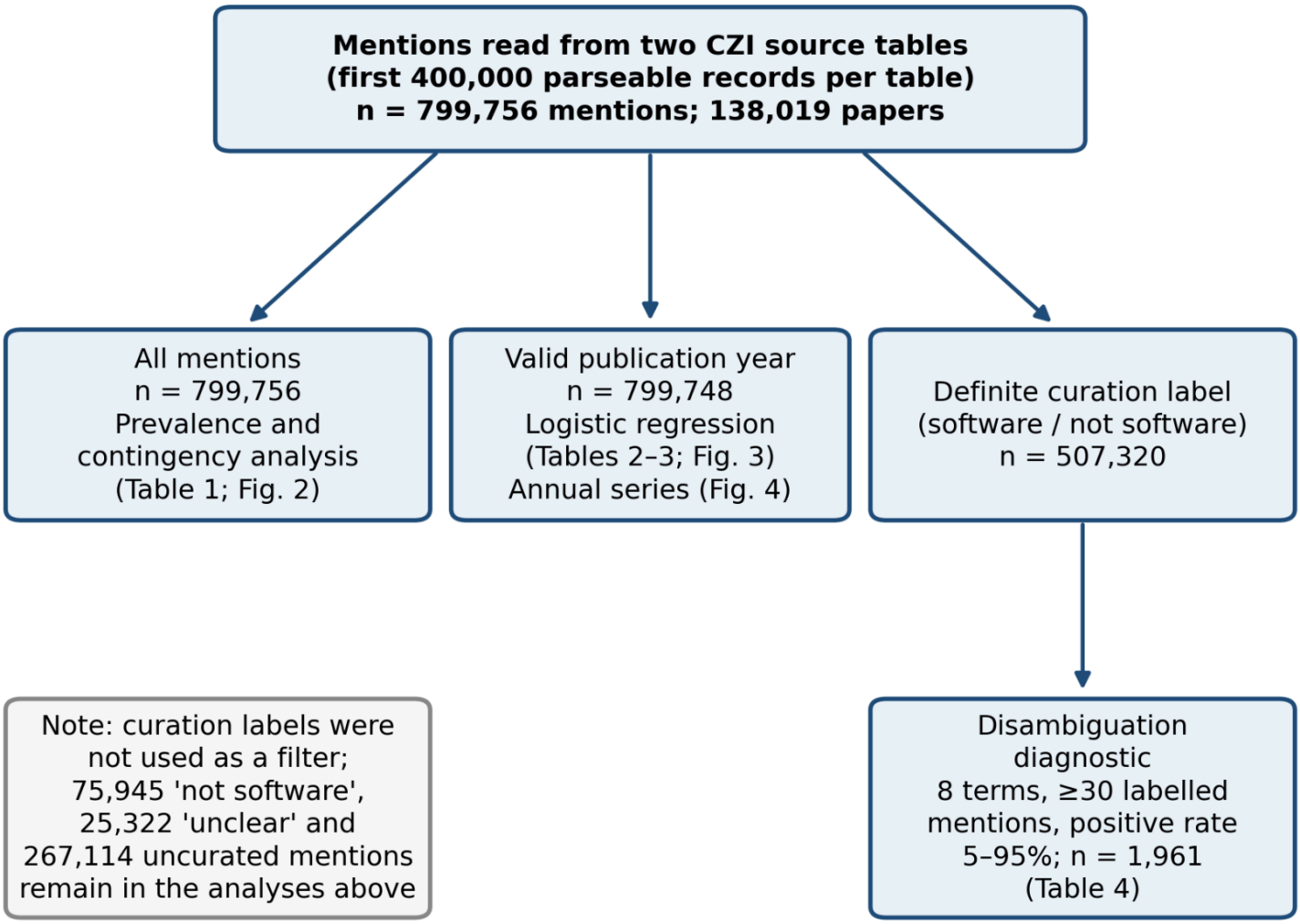


Figure 1

Flow of mentions through the analyses. All three analysis sets are drawn from the same 799,756 mentions; curation labels define the disambiguation set but were not used to filter the prevalence and regression analyses.

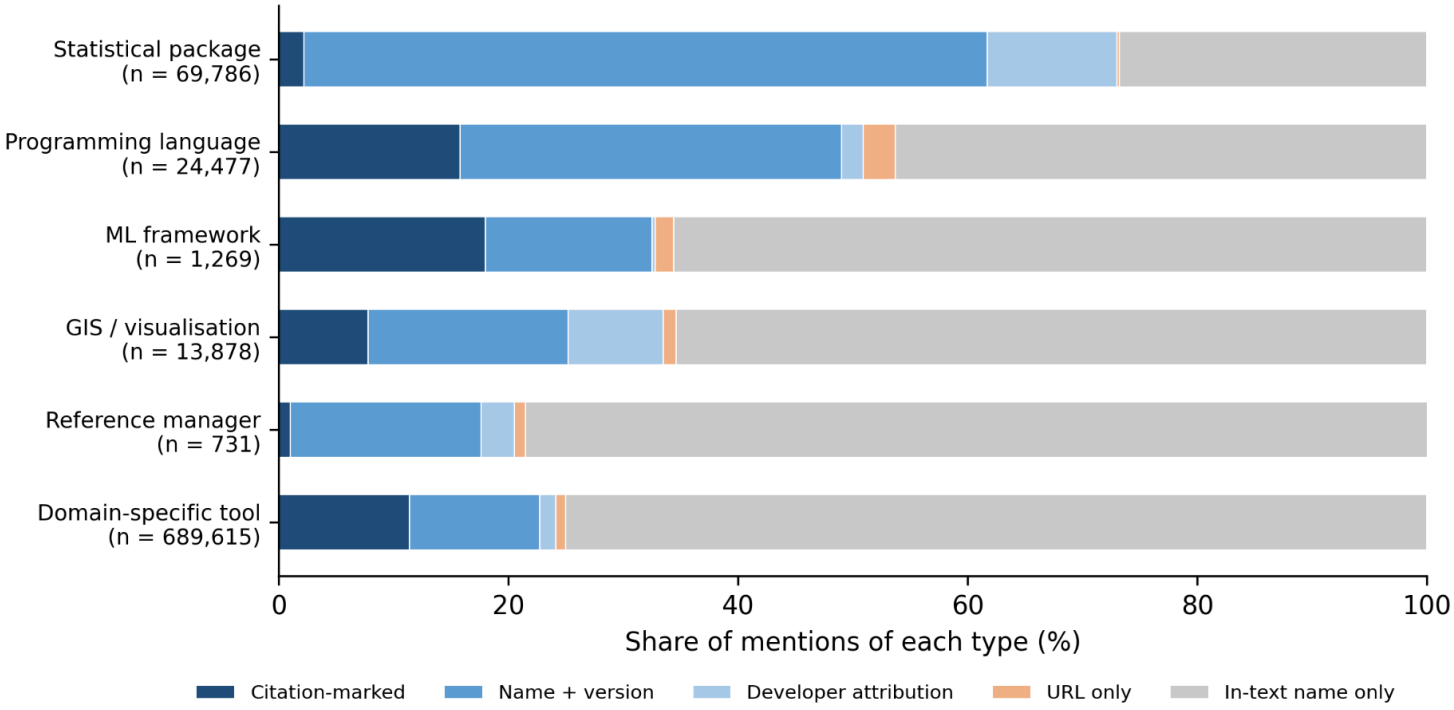


Figure 2

Mention style by software type, as a share of all mentions of each type. Data as in Table 1.

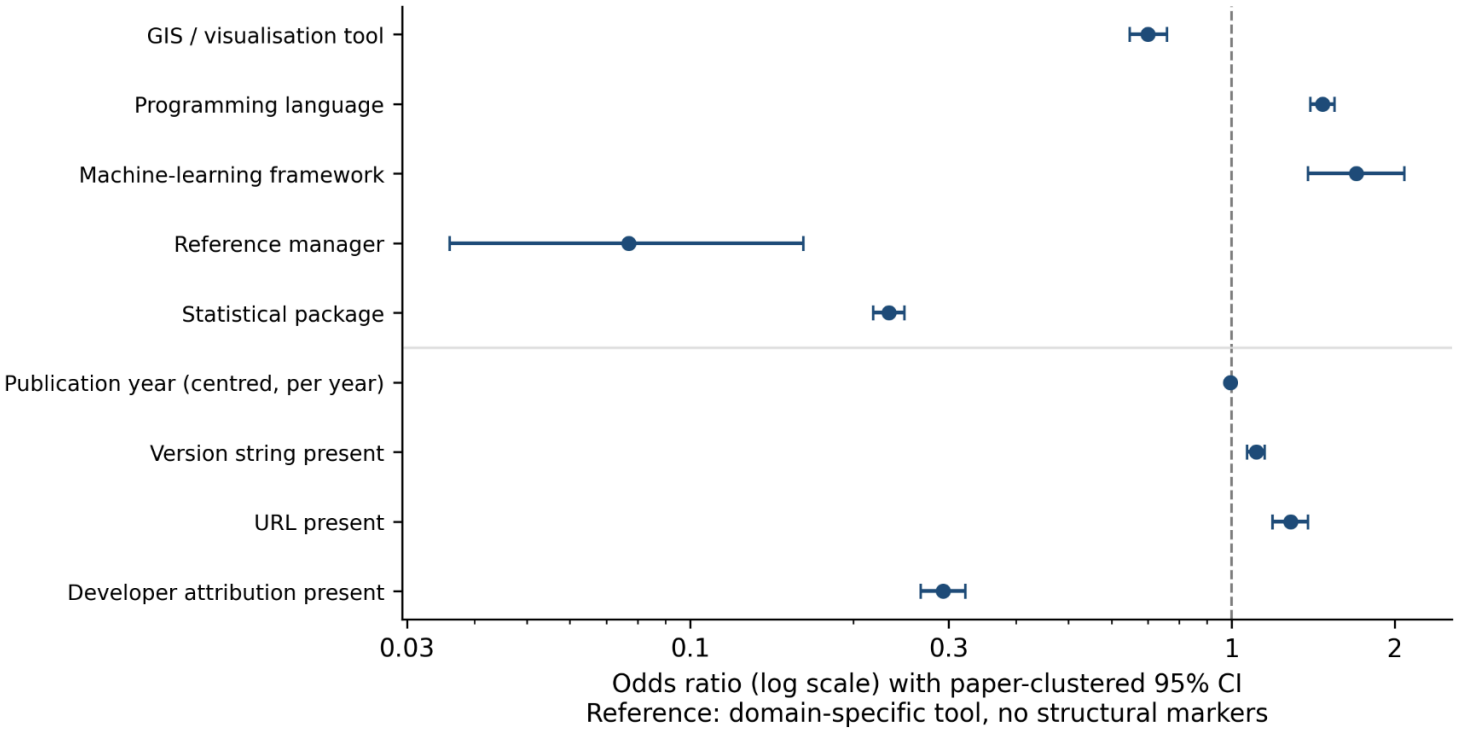


Figure 3

Odds ratios for citation marking with paper-clustered 95% confidence intervals, logarithmic scale. The dashed line marks OR = 1.

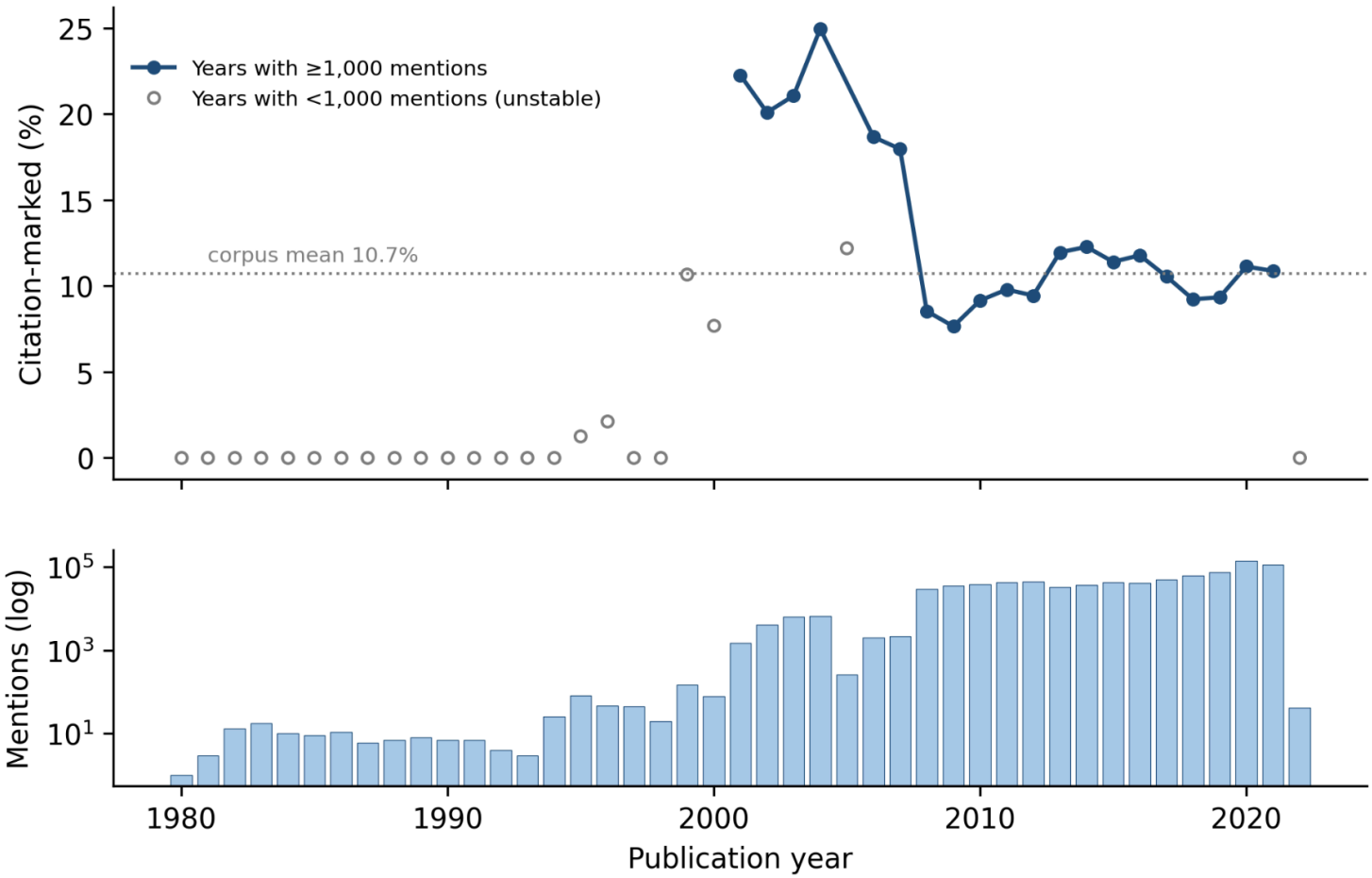


Figure 4

Share of citation-marked mentions by publication year (upper panel) and number of mentions per year on a logarithmic scale (lower panel). Hollow markers denote years with fewer than 1,000 mentions, including 2022 (41 mentions), whose rates are unstable. Data in Supplementary Table S5.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- SupplementaryMaterial.docx