(Re)Use of Research Results (is Rampant)

MARIA TERESA BALDASSARRE, University of Bari, Italy
NEIL ERNST, University of Victoria, Canada
BEN HERMANN, Technische Universität Dortmund, Germany
TIM MENZIES, RAHUL YEDIDA, NC State University, USA

In any field, finding the "leading edge" of research is an on-going challenge. Researchers cannot appease reviewers and educators cannot teach to the leading edge of their field if no one agrees on what is the state-of-the-art.

Using a crowdsourced "reuse graph" approach, we propose a new method to learn the state-of-the-art. Reuse graphs are less effort to build and verify than other monitoring methods (e.g. artifact tracks or citation-based searches). Based on a study of 170 papers from software engineering (SE) conferences in 2020, we found 1,600+ instances of reuse; i.e., reuse is rampant in SE research. Prior pessimism about reuse in SE research may have been a result of using the wrong methods to measure the wrong things.

ACM Reference Format:
Maria Teresa Baldassarre, Neil Ernst, Ben Hermann, and Tim Menzies, Rahul Yedida . 2022. (Re)Use of Research Results (is Rampant). 1, 1 (November 2022), 16 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

INTRODUCTION

According to Popper [23], the ideas we can *most trust* are those that have been *most tried* and *most tested*. For that reason, many of us are involved in the process called "Science" that produces trusted knowledge by sharing one's ideas, and trying out and testing others' ideas. Science and scientists form communities where people do each other the courtesy of curating, clarifying, critiquing and improving a large pool of ideas.

R1.6 According to this definition, one measure of the health of a scientific community is how much it reuses results. By that measure, the software engineering research community might seem to be very unhealthy. Da Silva et al. reported that from 1994-2010, only 72 studies had been replicated by 96 new studies [9]. As a double check for da Silva's conclusion, in February 2022, we queried the ACM Portal for products from the "International Conference on Software Engineering" (the premier conference in that field). In the period 2011 to 2021, only 111 out of the 8774 ICSE research entries were associated with 'available', 74 with 'reusable', 24 with 'functional', and none with 'replicated' or 'reproduced' reuse (for a definition of those terms, see Table 1). To say that another way, according to the ACM Portal, only 2.4% of the ICSE publications are explicitly associated with any kind of reuse. Worse still, according to that report, there were no replicate or reproduce results from ICSE in the last decade.

Authors' addresses: Maria Teresa BaldassarreUniversity of Bari, Italy, mariateresa.baldassarre@uniba.it; Neil ErnstUniversity of Victoria, Canada, nernst@uvic.ca; Ben HermannTechnische Universität Dortmund, Germany, ben.hermann@cs.tu-dortmund.de; Tim Menzies, Rahul Yedida NC State University, USA, timm@ieee.org.ryedida@ncsu.edu.

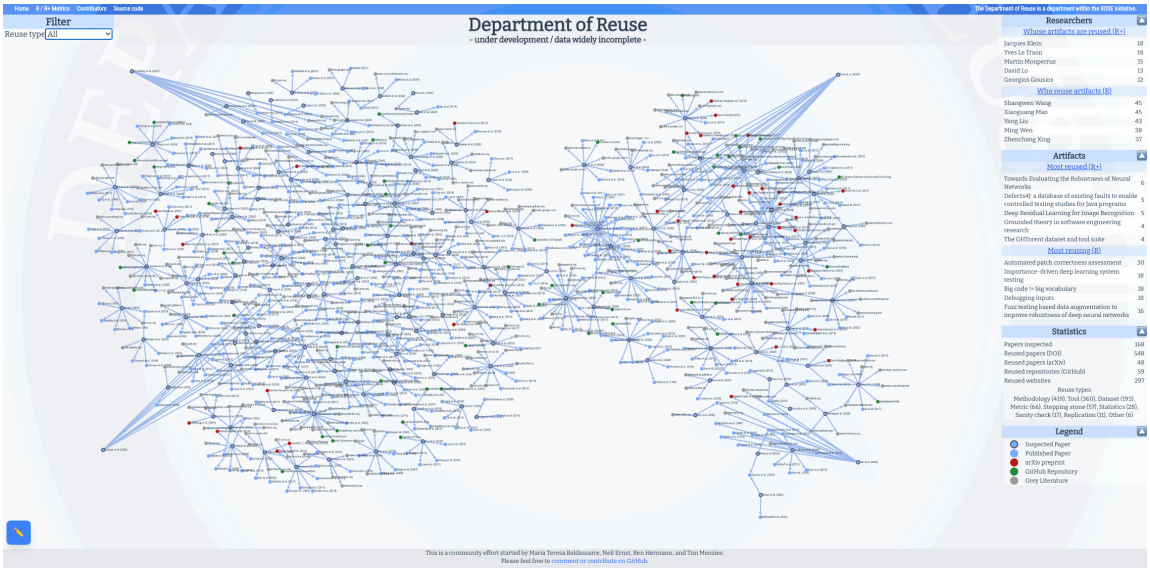


Fig. 1. From the web-site https://reuse-dept.org. The 1,635 arrows in this diagram connects reuser to reused. Blue dots denote the 714 sources found with with a digital object identifier (a DOI); e.g. any paper from a peer-reviewed source. Red dots denote the 48 papers we found without DOIs (e.g. those from arxiv.org). Gray and green denote the 297 works in grey literature and 57 Github repositories (respectively) reused in this sample. The black squares pull out four examples where a paper has reused material from dozens of other sources. Data collection for this graph is on-going and at the time of this writing, that data comes from 40% of the papers published in the 2020 technical program of ICSE, ASE, FSE, ICMSE, MSR, and ESEM.

Table 1. Badges currently awarded conferences [2]. This table is for the ACM. Analogous tables are used at other conferences.

Available	Functional	Reusable	Reproduced	Replicated
				
In a public repository with a long-term retention policy. A DOI needs to be provided.	Artifacts are documented, consistent, complete, exercisable, and include evidence of verification and validation.	Functional, significantly exceed minimal functionality.	Results of this paper have been reproduced by a different team using the original artifact.	Results of this paper have been replicated by a different team without the original artifact.

We argue that, at least in the area of software engineering, this "reuse problem" is more apparent, than real. We describe a R1.7 successful approach to recording research reuse where teams of researchers from around the world read 170 recent (2020) conference papers from software engineering. This work generated the "reuse graph" of Figure 1. In that figure, each edge connects papers to the prior work that they are (re-)using. As discussed below, when compared to other community monitoring methods (e.g. artifact tracks or bibliometric searches [3, 11, 19]), these reuse graphs are less effort to build and verify. For example, it took around 12 minutes per paper for our team from Hong Kong, Canada, the United States, Italy, Sweden, Finland, and Australia to apply this reuse graph methodology to software engineering[1].

[1]That team included the authors of this paper plus Jacky Keung from City University (Hong Kong); Greg Gay from Chalmers University (Sweden); Burak Turhan from Oulu University (Finland); and Aldeida Aleti from Monash University (Australia). We gratefully acknowledge their work, and that of their graduate students. In particular, we especially call out the work of Afonso Fontes from Chalmers University (Sweden).

The rest of this paper discusses generation and application and value of our reuse graphs. Before beginning, we offer the follow introductory remark. This paper is written as a protest, of sorts, against how we currently assess science and scientific output. Between us, the authors of this paper have worked as researchers for decades, supervising graduate students and organizing prominent conferences and journals. Based on that experience, we assert that researchers do more than write papers. Rather, we are all engaged in long-term stewardship of the ideas. As part of that stewardship, we generate more than just papers. Yet of our products, it is only our papers that are used, mostly in some annual bibliometric analysis of our worth. We view this as an inadequate way to measure what researchers do.

The problem, we think, is in the very term "bibliometric". This term is heavily skewed towards publications and monographs and the kinds of things we can easily store in the repositories of our professionals societies (e.g., IEEE Xplore and ACM Portal). In fact, the term *bibliométrie* was first used by Paul Otlet in 1934 [25], was defined as "the measurement of all aspects related to the publication and reading of books and documents." Subsequent definitions tried to broaden that definition e.g. the anglicised version *bibliometrics* was first used by Alan Pritchard in his 1969 paper *Statistical Bibliography or Bibliometrics?* [24] where he defined the term as "the application of mathematics and statistical methods to books and other media of communication". R2.0 But what we are observing in 2022 is that "other media of communication" in software engineering (and other fields) is far broader than just the products stored in the repositories of our professional societies. RE.2 For example, researchers might "use" the results of papers, or "follow" guidance from one paper in their own work, or "download" data or code used on another paper (then use locally). We argue that all such "downloads" or "guidance following" as examples of "reuse", since all are examples where the members of our research community reused products from other research into their own work (for a more exact categorization of the types of reuse we are studying, please see our section *Studying Reuse*).

It is all too easy to propose a broader definition for how scholars reuse and communicate their products. Such a new definition is practically useless *unless* we can propose some method to collect data on that new definition. Here we suggest that our new definitions can be operationalized via crowdsourced methods.

CAPTURING REUSE

There are many methods to map the structure of SE research like (a) manual or automatic citations searchers or (b) "artifact evaluation committees" that foster the generation and sharing of research products. Such studies can lag significantly behind current work. For example, in our own prior citation analysis of SE [19], we only studied up to to 2016. The study itself was conducted in 2017, but not fully published till 2018. Given the enormous effort required for that work, we have vowed never to do it again.

Reuse graphs, on the other hand, are faster to keep up to date since the work of any one individual working on these graphs is minimal. Other reasons for favoring reuse graphs are that they are community *comprehensible*, community *verifiable*, and community *correctable*. All the data used for our reuse graphs is community-collected. All the data can be audited at https://reuse-dept.org and if errors are detected, issue reports can be raised in our GitHub repository (and the error corrected). The same may *not* be true for studies based on citation servers run by professional bodies and for-profit organizations (e.g., see Table 2). New data can be contributed by anyone either directly supplying data in our format or using a user interface directly on our website for easier access. The resulting issue report is then reviewed and (when necessary) corrected. After a third person successfully inspects the data it will be added to the reuse graph.

What is the value of a verified, continually updated, snapshot of some current research area? Once our reuse graph covers several years (and not just 2020 conference publications), we foresee several applications:

EXAMPLE #1: One of the authors has an entry in Google Scholar metrics software systems saying that their paper "How to" has 80 citations in the last five years at IEEE Transactions on Software Engineering. That link connects to some other paper, not written by any of us. There is no "help" button at Google Scholar where this error can be reported. This is disappointing since it is suspected that this mysterious "How to" paper references work that might be the most cited from IEEE TSE in the last five years. That would be a significant achievement, if we could document it (but using Google Scholar, we cannot).
EXAMPLE #2: In our recent large scale text-mining studies of 30,000+ SE papers [19], it was found that papers can appear in our citation server, but not in others. Also, examples were found where some papers had twenty times the citations in one server than in another. Here again, we were unable to contact anyone working on those citation servers to fix those errors.
EXAMPLE #3: Even if owners of these citations sites can be contacted, their errors may remain unfixed. E.g. there is no accepted convention for how to typeset a hyphen. Different venues add in zero to one space before or after while others typeset the hyphen as two dashes. Zhou et al. found that (a) papers with hyphens in the title get reported as different papers in different venues; which means that (b) those papers get fewer citations [29]. Zhou et al. report that when they contacted the owners of these citation servers, rather than fix the errors, those owners started lobbying for the Zhou et al. paper not to be published.

Table 2. Examples of errors in citation servers.

(1) Academics can check that their contributions to science are being properly recorded;
(2) When applying for promotion or hiring, research faculty or industrial workers could document the impact of their work beyond papers, including tools, datasets, and innovative methods;
(3) Graduate students could direct their attention to research areas that are both very new (nodes from recent years) and very productive (nodes with an unusually large number of edges attached);
(4) Organizers of conferences could select their keynote speakers from that space of new and productive artifacts.
(5) Growth patterns might guide federal government funding priorities or departmental hiring plans.
(6) Venture capitalists could use these graphs to detect emergent technologies, perhaps even funding some of those.
(7) Conference organizers could check if their program committees have enough members from currently hot topics.
(8) Further, those same organizers could create new conference tracks and journals sections in order to service active research communities that are under-represented in current publication venues.
(9) Journal editors could find reviewers with relevant experience.
(10) Educators can use the graphs to guide their teaching plan.

STUDYING REUSE

In our reuse study, we targeted papers from the 2020 technical programs of six major international SE conferences: Software Engineering (ICSE), Automated Software Engineering (ASE), Joint European Software Engineering Conference / Foundations of Software Engineering (ESEC/FSE), Software Maintenance and Engineering (ICSME), Mining Software Repositories (MSR), and Empirical Software Engineering and Measurement (ESEM). These conferences were selected using advice from [19], but our vision is to expand; for example, by looking at all top-ranked SE conferences. GitHub issues were used to divide up the hundreds of papers from those conferences into "work packets" of ten papers each. Reading teams were set up from software engineering research teams from around the globe in Hong Kong, Istanbul (Turkey), Victoria (Canada), Gothenburg (Sweden), Oulu (Finland), Melbourne (Australia), and Raleigh (USA). Team members would assign themselves work packets and then read the papers looking for the kinds of reuse enumerated below. Once completed, a second person (from any of our teams) would do the same and check for consistency. Fleiss Kappa statistics are then computed to track the level of reader disagreement. GitHub issues[2] were used to manage this

[2] e.g. https://github.com/bhermann/DoR/issues/20

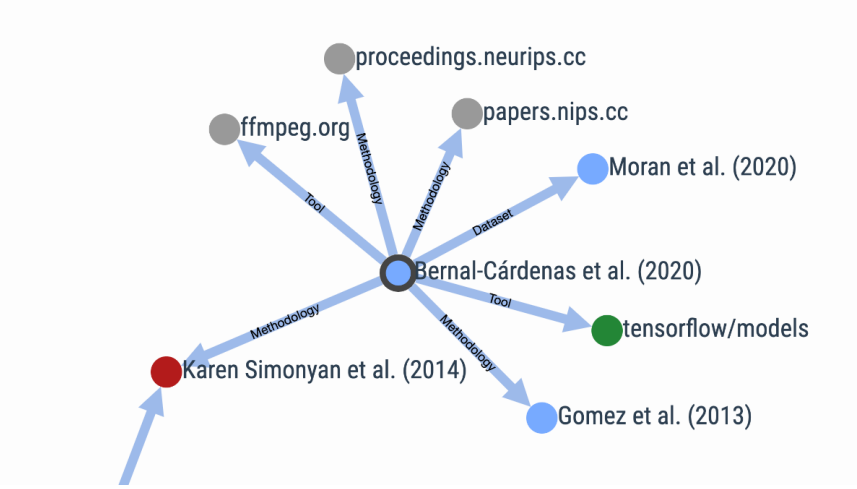


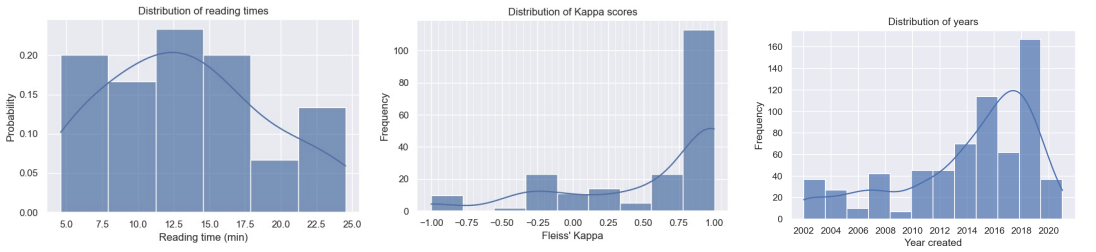
Fig. 2. Detailed view of a part of the reuse graph (from Fig. 1), showing reuse from Bernal-Cárdenas et al. [5], DOI:10.1145/3377811.3380328. Edges reflect *tool*, *dataset* and *methodology* reuse. Red nodes indicate arXiv preprint; green GitHub repository; blue published paper, and grey other websites or grey literature locations.

in the open but raters were asked not to examine the previous results. A member of the author team of this paper then did a final check on disagreements before including the data into the graph. R3.2

Teams were asked to record six kinds of reuse:

(1) Most papers have to benchmark their new ideas against some prior recent state-of-the-art paper. That is, they reuse old papers as **stepping stones** towards new results.
(2) Another thing that is often reused are **statistical** methods. Here we do not mean "we use a two-tailed t-test" or some other decades-old widely-used statistical method. Rather, we refer instead to statistical methods for recent papers that propose statistical guidance for the kinds of analysis seen in SE. Perhaps because this kind of analysis is very rare, this work is exceedingly highly cited; e.g.
   - A 2008 paper *Benchmarking Classification Models for Software Defect Prediction* [17] has 1,178 citations;
   - A 2011 paper *A practical guide for using statistical tests to assess randomized algorithms* [1] has 778 citations.
(3) **Metrics and methodology** descriptions specific to the research area. These include software metrics such as CK metrics or flow metrics, and research methods such as grounded theory or sampling criteria R2.2 ;
(4) **Data sets**;
(5) **Sanity checks**, which are justification for why a particular approach works or is reasonable to avoid bad data, e.g., why to avoid using GitHub stars to select repositories [15] R2.3 ;
(6) and, indeed, the software packages of the kind currently being reviewed by SE conference artifact evaluation committees (**tools and replications**).

Figure 2 shows an example. Starting with a paper by Bernal-Cárdenas et al. [5], we find among others a reused dataset from Moran et al. [22], tool reuse of FFmpeg and Tensorflow object detection, and several reused methods, including the ConvNet approach described by Simonyan. Readers can follow the DOI linked in the figure caption to see more detailed information.



(a) Reading times (b) Fleiss Kappa scores (c) Years that the works were created.

Fig. 3. Reading time results (left). Agreement scores (center). Yearly prevalence of reused papers (right). In the middle chat, the Fleiss Kappa score [12] measures the inter-rater reliability among multiple raters when there are categorical ratings to items. We prefer it to Cohen's kappa measure, which only works for 2 raters. In our case, we had multiple raters assigning the categories "Yes" (signifying that a paper was reused), or "No" (signifying that it was not) to a pool of papers. This pool was obtained by taking the union of all the papers marked as being reused by a specific paper.

We can report that it is not difficult to read papers in order to detect these kinds of reuse:

- It is fast to find the above six kinds of reuse. Our graduate students report that reading their first paper might take up to an hour. But after two or three papers, the median reading time drops to around 12 minutes (see Figure 3a).
- When we compare the reuse reported by different readers, we get Figure 3b. In our current results, the median Fleiss Kappa score (for reviewer agreement) is one (i.e. very good).
- The one caveat we would add is that any graduate student involved in this activity needs at least two years of active research experience in their area of study. We say this since when we tried data collection from a large intro-to-SE graduate subject, the resulting Kappa agreement scores were poor **R2.9** .

RE.3 The result of this data collection is a directed multi-graph of publications and other forms of dissemination of research artifacts. The edges of this graph are annotated with the type of reuse according to the list above. Reuse metrics for a specific publication (or other form, e.g. a GitHub repository) are the in-degree and out-degree measures of the node that represents this publication. When accumulated for the originating authors, individual reuse metrics can be collected. Zooming into the graph on our website reuse types are visibly annotated at the graph edges (see Figure 2). A filter allows to extract a graph for a single reuse type out of the multi-graph.

Of course, there any many other items being reused than the six listed above[3]. It is an open question, worthy of future work, to check if those other items can be collected in this way, and indeed, to refine these categories as understanding changes **R2.4** .

RELATED WORK

Apart from software engineering [21], many other disciplines are actively engaged in artifact creation, sharing, and re-use [8, 18]. Artifacts are useful for building a culture of replication and reproducibility[7, 16], already acknowledged as important in SE [6, 9, 14, 27]. Fields such as psychology have had many early results thrown into doubt because of a failure to replicate the original findings [28]. Sharing research protocols and data through replication packages and artifacts allows for other research teams to conduct *severe* tests of the original studies [20], strengthening (or rejecting)

[3]See e.g. see the 22 types of potentially reusable items at https://github.com/researchart/patterns#patterns-and-artifacts

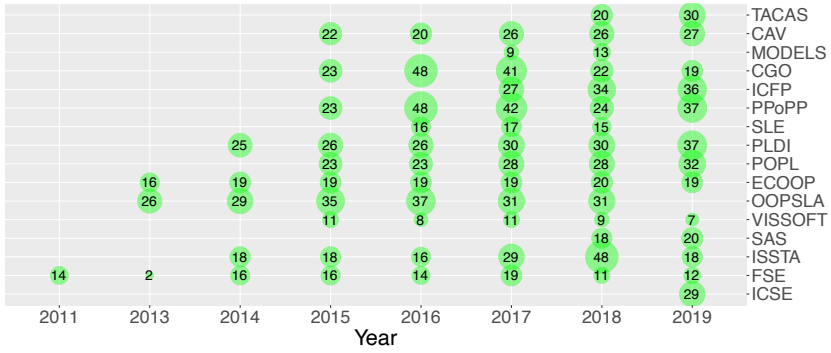


Fig. 4. Artifact evaluation committee sizes 2011-2019. From Hermann et al. [13]

these initial findings. In medicine, drug companies are mandated to share the research protocols and outcomes of their drug trials, something that has become of vital recent importance (albeit not without challenges [10]). In physics and astronomy, artifact sharing is so commonplace that large community infrastructures exist solely to ensure data sharing, not least because governments which fund these costly experiments insist on it.

In more theoretical areas of CS, pioneering use of preprint servers has enabled 'reuse' of proofs, essential to progress. In machine learning, replication is focused on stepping-stones, enabled by highly successful benchmarks such as ImageNet [26]. However, recent advances with extremely costly training regimens have called replicability into question[4].

In the specific case of software engineering research, prior to this paper, there was little recorded and verified evidence of reuse. Many researchers have conducted *citation studies* that find links to highly cited papers (e.g. [19]). As stated in our introduction, such studies can lag behind the latest results. Also, recalling Table 2, we have cause to doubt the conclusions from such citation studies.

From a practical perspective, many conferences have recently introduced *artifact evaluation committees* to entice reuse and replication. Moreover, authors of accepted conference papers submit software packages that, in theory, let others re-execute that work.[6, 7] These evaluation committees award "badges" as shown in Table 1.

Artifact evaluation is something of a "growth industry" in SE (and programming languages, or PL, community) as can be seen in Figure 4 which shows the increasing number of people evaluating artifacts 2011 to 2019. One may conclude that such practices make the community more aware of what is "available" and can be "reused", and therefore, become a potential node of a "reuse graph". As such, the source is explicitly made available to any other researcher willing to (re)use it [6, 13]. R1.3

Now, the question to be asked is: are all the people of Figure 4 making the best use of their time? Perhaps not. We note that most artifacts are assigned the badges requested by the authors. Given that, it might be safe to ask some of the personnel from Figure 4 to (e.g.) spend less time on evaluating conference artifacts and spend more time working on Figure 1.

But most importantly, it is not clear whether the artifact evaluation process is actually creating *reused* artifacts and therefore, indirectly contributing to the "reuse graph" concept. Indeed, if we query ACM Portal for "software

[4]https://www.technologyreview.com/2020/11/12/1011944/artificial-intelligence-replication-crisis-science-big-tech-google-deepmind-facebook-openai/

Table 3. The ROSE initiative: Recognizing and Rewarding Open Science in Software Engineering: an international multi-conference workshop that will continually report updates to the SE reuse graphs.

Researchers that reuse the most from other papers will be applauded and awarded a "R-index" (reuse index).
Researchers that build the artifacts that are most reused will be applauded (even louder) and be awarded an "R+-index" indicating that they are the people producing the artifacts that are most used by the rest of the community.
In between each conference, the ROSE initiative will co-ordinate an international team of volunteers incrementally updating the SE reuse graph. R1.12 The authors of this paper have volunteered to serve as the initial nucleus of that group– and we will hold open elections to grow that steering committee using other people from our community.
This reuse graph will be displayed at a publicly available web site (reuse-dept.org). Individual researchers can browse that site, check their entries, and propose corrections and extensions.
All reports of reuse will be double checked and disputed claims will be then be tripled-checked.
All the tools used to create that web site will be freely available for download. Hence, if the SE community does not like how we are running these reuse graphs, they can take all our code and data and do something else.
Also, researchers from other disciplines can take our tools and apply them to their own community.

engineering" and "artifacts" in the range 2015 to 2020, we find that most of the recorded artifacts are not reused in replications or reproductions[5]. Specifically, only $\frac{1}{20}$th are reproduced and only $\frac{1}{50}$th are replicated.

Perhaps it might be useful to reflect more on what is actually being reused (as we have done, above). This is what has motivated our research and led us to creating the "reuse graph". R2.13

NEXT STEPS FOR REUSE GRAPHS

RE.4 When discussing this work with colleagues, we are often asked if we have assessed this work. In reply we comment that, at this stage, this is like asking the inventors of kd-trees [4] in 1975 how much that method has sped up commercial databases. Right now, we are engaged in community building and have shown that with very little effort, and not much coding, we can create the infrastructure needed to collect our data. While Figure 1 is a promising start, to scale up from here, we need to organize a larger reading population. Our goal is to analyze 200, 2000, 5000 papers in 2022, 2023, 2024 (respectively) by which time we would have covered most of the major SE venues in the last five years. After that, our maintenance goal would be to read 500 (ish) papers per year to keep up to date with the conferences (and after that, we would move on to journals). Based on Figure 3a, then assuming each paper is read by two people, then maintenance goal would be achievable by a team of twenty people working two hours per month on this task. To organize this work, we have created the "ROSE initiative", see Table 3.

If that work interests you, then there are many ways you can get involved:

- If you are a researcher and wish to check that we have accurately recorded your contribution, please visit https://reuse-dept.org.
- If you want to apply reuse graphs to your community, please use our tools at https://github.com/bhermann/DoR/.
- If you are interested in joining this initiative and contributing to an up-to-minute snapshot of SE research, then please (a) take our how-to-read-for-reuse tutorial[6]; (b) then visit the dashboard at the GitHub site (bhermann/DoR), find an issue with no one's face on it, and assign yourself a task.

We see this effort as one part of the broader open science effort, in addition to helping the community identify the state of the art (e.g., patterns of growth in the reuse graph). Among the goals of open science are to increase confidence

[5] As of December 10 2020, that search returns 2,535 SE papers with artifact badge. Of these, 43%, 30%, 20%, 5%, 2% are available, functional, reusable, reproduced and replicated artifacts (respectively).
[6] https://github.com/bhermann/DoR/blob/main/workflow/training.md

in published results, and to acknowledge that science produces more types of artifacts than just publications: researchers also produce method innovations, new datasets, and improved tools. If we take an agile view of SE science, then as researchers we should focus on generating these artifacts and rapidly securing critique, curation, and clarification from our peers and the public.

REFERENCES

[1] Andrea Arcuri and Lionel Briand. 2011. A Practical Guide for Using Statistical Tests to Assess Randomized Algorithms in Software Engineering. In *Proceedings of the 33rd International Conference on Software Engineering* (Waikiki, Honolulu, HI, USA) *(ICSE '11)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/1985793.1985795

[2] Association for Computing Machinery. 2020. Artifact Review and Badging. https://www.acm.org/publications/policies/artifact-review-and-badging-current. Accessed: 2020-12-08.

[3] Maria Teresa Baldassarre, Danilo Caivano, Simone Romano, and Giuseppe Scanniello. 2019. Software models for source code maintainability: A systematic literature review. In *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 252–259.

[4] Jon Louis Bentley. 1975. Multidimensional Binary Search Trees Used for Associative Searching. *Commun. ACM* 18, 9 (sep 1975), 509–517. https://doi.org/10.1145/361002.361007

[5] Carlos Bernal-Cárdenas, Nathan Cooper, Kevin Moran, Oscar Chaparro, Andrian Marcus, and Denys Poshyvanyk. 2020. Translating video recordings of mobile app usages into replayable scenarios. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ACM. https://doi.org/10.1145/3377811.3380328

[6] Bruce R. Childers and Panos K. Chrysanthis. 2017. Artifact Evaluation: Is It a Real Incentive?. In *2017 IEEE 13th International Conference on e-Science (e-Science)*. 488–489. https://doi.org/10.1109/eScience.2017.79

[7] Christian Collberg and Todd A. Proebsting. 2016. Repeatability in Computer Systems Research. *Commun. ACM* 59, 3 (feb 2016), 62–69. https://doi.org/10.1145/2812803

[8] Badampudi D., Wohlin C., and Gorschek T. 2019. Contextualizing research evidence through knowledge translation in software engineering. *ACM International Conference Proceeding Series* (2019), 306 – 311. https://doi.org/10.1145/3319008.3319358 Cited by: 9.

[9] Fabio Q. B. da Silva, Marcos Suassuna, A. César C. França, Alicia M. Grubb, Tatiana B. Gouveia, Cleviton V. F. Monteiro, and Igor Ebrahim dos Santos. 2012. Replication of empirical studies in software engineering research: a systematic mapping study. *Empirical Software Engineering* (Sept. 2012). https://doi.org/10.1007/s10664-012-9227-7

[10] Nicholas J DeVito, Seb Bacon, and Ben Goldacre. 2020. Compliance with legal requirement to report clinical trial results on ClinicalTrials.gov: a cohort study. *The Lancet* 395, 10221 (feb 2020), 361–369. https://doi.org/10.1016/s0140-6736(19)33220-9

[11] Katia Romero Felizardo, Érica Ferreira de Souza, Bianca Minetto Napoleão, Nandamudi Lankalapalli Vijaykumar, and Maria Teresa Baldassarre. 2020. Secondary studies in the academic context: A systematic mapping and survey. *Journal of Systems and Software* 170 (2020), 110734.

[12] Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76, 5 (1971), 378.

[13] Ben Hermann, Stefan Winter, and Janet Siegmund. 2020. Community Expectations for Research Artifacts and Evaluation Processes. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Virtual Event, USA) *(ESEC/FSE 2020)*. Association for Computing Machinery, New York, NY, USA, 469–480. https://doi.org/10.1145/3368089.3409767

[14] Robert Heumüller, Sebastian Nielebock, Jacob Krüger, and Frank Ortmeier. 2020. Publish or Perish, but do not Forget your Software Artifacts. *Empirical Software Engineering* (2020). https://doi.org/10.1007/s10664-020-09851-6 Preprint.

[15] Eirini Kalliamvakou, Georgios Gousios, Kelly Blincoe, Leif Singer, Daniel M. German, and Daniela Damian. 2014. The promises and perils of mining GitHub. In *Proceedings of the 11th Working Conference on Mining Software Repositories - MSR 2014*. ACM Press. https://doi.org/10.1145/2597073.2597074

[16] Shriram Krishnamurthi and Jan Vitek. 2015. The Real Software Crisis: Repeatability as a Core Value. *Commun. ACM* 58, 3 (feb 2015), 34–36. https://doi.org/10.1145/2658987

[17] Stefan Lessmann, Bart Baesens, Christophe Mues, and Swantje Pietsch. 2008. Benchmarking Classification Models for Software Defect Prediction: A Proposed Framework and Novel Findings. *IEEE Transactions on Software Engineering* 34, 4 (2008), 485–496. https://doi.org/10.1109/TSE.2008.35

[18] Baker M. and Penny D. 2016. Is there a reproducibility crisis? *Nature* 533, 7604 (2016), 452 – 454. https://doi.org/10.1038/533452A Cited by: 1507; All Open Access, Bronze Open Access.

[19] George Mathew, Amritanshu Agrawal, and Tim Menzies. 2018. Finding Trends in Software Research. *IEEE Transactions on Software Engineering* (2018), 1–1. https://doi.org/10.1109/TSE.2018.2870388

[20] D. G. Mayo. 2018. *Statistical inference as severe testing: How to get beyond the statistics wars*. Cambridge University Press.

[21] Tim Menzies. 2013. Guest editorial for the Special Section on BEST PAPERS from the 2011 Conference on Predictive Models in Software Engineering (PROMISE). *Information and Software Technology* 55, 8 (2013), 1477–1478.

[22] Kevin Moran, Carlos Bernal-Cardenas, Michael Curcio, Richard Bonett, and Denys Poshyvanyk. 2020. Machine Learning-Based Prototyping of Graphical User Interfaces for Mobile Apps. *IEEE Transactions on Software Engineering* 46, 2 (Feb. 2020), 196–221. https://doi.org/10.1109/tse.2018.2844788

[23] Karl Popper. 2014. *Conjectures and refutations: The growth of scientific knowledge*. Routledge.

[24] Alan Pritchard. 1969. Statistical Bibliography or Bibliometrics? *Journal of Documentation* 25 (1969), 348–349. Issue 4.
[25] Ronald Rousseau. 2014. Forgotten founder of bibliometrics. *Nature* 510 (2014).
[26] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* 115, 3 (2015), 211–252. https://doi.org/10.1007/s11263-015-0816-y
[27] Adrian Santos, Sira Vegas, Markku Oivo, and Natalia Juristo. 2021. Comparing the results of replications in software engineering. *Empirical Software Engineering* 26, 2 (Feb. 2021). https://doi.org/10.1007/s10664-020-09907-7
[28] Ulrich Schimmack. 2020. A meta-psychological perspective on the decade of replication failures in social psychology. *Canadian Psychology/Psychologie canadienne* 61, 4 (nov 2020), 364–376. https://doi.org/10.1037/cap0000246
[29] Zhi Quan Zhou, T. H. Tse, and Matt Witheridge. 2021. Metamorphic Robustness Testing: Exposing Hidden Defects in Citation Statistics and Journal Impact Factors. *IEEE Transactions on Software Engineering* 47, 6 (2021), 1164–1183. https://doi.org/10.1109/TSE.2019.2915065