Reproducible aggregation of multi-source citation data with OpenAlex support and GitHub Actions

Dušan Nikolić [0000-0002-7379-3588] and Dragan Ivanović[0000-0002-9942-5521]

University of Novi Sad, Faculty of Technical Sciences, Trg Dositeja Obradovića 6, Novi Sad, Serbia
{nikolic.dusan,dragan.ivanovic}@uns.ac.rs

Abstract. Bibliometric analyses often rely on combining data from multiple citation databases to achieve comprehensive coverage and accurate results. In our previous study, we presented an open-source tool that enables automated merging and deduplication of records from Scopus and Web of Science (WoS). However, emerging approaches - like integrating OpenAlex, minimizing manual data handling, and enabling reproducible automation, are still underdeveloped. To address this, we enhance our toolkit with four key features: (i) integration of the OpenAlex API, (ii) execution of the entire workflow on GitHub Actions for unattended data retrieval, (iii) a fork-based configuration model that simplifies collaboration, and (iv) an aggregation module coupled with versioning via git tags. These improvements reduce researcher workload and advance the principles of openness and reproducibility in bibliometric practice.

Keywords: Bibliometric Analysis · Citation Databases · OpenAlex · Continuous Integration · GitHub Actions · Version Control

1 Introduction

Comprehensive bibliometric studies increasingly demand the use of proprietary and now explore the usage of open citation indexes. Proprietary services such as Web of Science (WoS) and Scopus provide curated bibliometric coverage, but remain costly and license restricted. However, the fully open OpenAlex [7] citation database offers transparent metadata and broader coverage of humanities and non-English research. In practice, researchers still expend substantial manual effort to fetch, merge and deduplicate records across sources. This effort produces manual workflows that are difficult to reproduce or audit.

Building on our earlier tool that merged Scopus and WoS data with high accuracy [1], we extend the system to (i) ingest OpenAlex via a dedicated sub-command, and (ii) run the entire pipeline in a continuous-integration (CI) environment. This study is driven by three research questions:
RQ1: How can bibliometric data be automatically retrieved from multiple sources, including OpenAlex?
RQ2: Which CI practices best guarantee user friendly, reproducible data retrieval and processing?

RQ3: What collaboration model lowers the barrier for non-expert researchers to adapt and reuse the toolkit?

At the same time the expectations for methodological transparency are increasing. Recent guidelines emphasize that bibliometric studies should document data preprocess and analytical steps in a form that can be independently repeated [4]. Nonetheless, in everyday practice researchers still invest considerable manual effort to download, merge, and clean records arriving in mutually incompatible formats: CSV or RIS from Scopus, tab-delimited text from WoS, and nested JSON from OpenAlex. Such ad-hoc processing not only consumes time but also makes it harder for peers to retrace decisions, verify results, or reuse intermediate datasets.

Our contribution is a workflow that can be forked on GitHub and configured with a single YAML file; each run produces a versioned or tagged dataset that can be cited or reused without additional scripting. Although conceived for bibliometrics, the design is content-agnostic: any domain in which structured records are exposed via an API: patent analytics, clinical-trial registries, policy documents, or even large-scale literature reviews, may adopt the same pattern to gain automated aggregation and transparent provenance tracking. Accordingly, the toolkit has the potential to foster open-science practices well beyond traditional citation analysis.

Section 2 surveys related work and the work that motivates multi-source aggregation; Sections 3 and 4 detail the methodology and implementation; Section 5 outlines current limitations and future work; and Section 6 concludes.

2 Related work

Early approaches focused on combining WoS and Scopus exports via DOI matching or fuzzy author/title similarity [5]. Recent Python libraries such as BibexPy provide GUI-assisted merging and metadata enrichment but remain limited to WoS and Scopus and lack open CI support [9].

Since OpenAlex was released in 2022, several language specific clients have been developed: openalexR [6], PyAlex, and the official API tutorial notebooks [10]. These tools are great at querying a single source. However, they do not yet provide any built-in mechanisms for merging proprietary datasets or for large-scale automation. Coverage comparison studies have demonstrated that OpenAlex outperforms WoS and Scopus in several Open Access (OA) categories [8, 12].

Several comparative studies underline the practical value of incorporating OpenAlex into multi-source workflows. Scheidsteger and Haunschild examine engineering conference proceedings and observe that OpenAlex records approximately 15% more entries than WoS while providing more complete affiliation and language fields [3]. In a broader assessment of open-access coverage, Huang et al. report that the share of openly

accessible publications indexed by OpenAlex is consistently higher than the corresponding shares in Scopus and WoS across most subject categories, with particularly marked differences in social-science and health-science datasets [8]. These findings indicate that an exclusive reliance on proprietary indices may omit a non-trivial volume of relevant material as well as valuable contextual metadata. For this reason, RQ1 concentrates on developing an automated mechanism that retrieves OpenAlex data alongside traditional sources and integrates the results into a unified pipeline.

At present, we haven't found any toolkit that combines multi-source data retrieval with reproducible CI. Community examples do exist, e.g. public GitHub workflows for Scopus downloads [14]. However, they remain ad hoc and undocumented, and no one integrates OpenAlex or supports fork-based collaboration. Therefore, our work aims to fill this methodological gap that we describe below.

3 Methodology

This extension builds on the methodology introduced in our previous paper [1], where deduplication was achieved through a similarity-based approach on user-configurable columns. The new components are as follows:

- OpenAlex support. We integrate the OpenAlex API, which allows retrieving bibliographic data programmatically [2]. This step increases the possible database coverage, as well as addresses potential challenges found during our prior deduplication and merging pipeline.
- CI automation where we define YAML-based workflows and periodically pull data from Scopus, WoS, and OpenAlex. Finally, the actions execute the merging, and aggregation.
- Fork-based collaboration models where researchers can fork the public repository, add their datasets, configure the pipeline and get reproducible and reliable results.
- Versioning. Because each run can be tagged with git, we can create reproducible "snapshots" of the data and results. The integrated chart-generation scripts provide visual outputs as part of the same workflow.

The toolkit follows a four-stage workflow: i) collect; ii) normalize; iii) aggregate; iv) publish. During collect, `tesci download` command retrieves raw records from each configured source and stores them on disk. A normalize step converts source-specific keys to a unified schema (DOI, title, authors, event type, year, citations, OA flag). In the current release, aggregate performs lightweight joins that concatenate normalized files based on user-configurable rules. The final publish stage is executed by GitHub Actions, committing the aggregated CSV or Excel to remote GitHub repository or multiple repositories, with both public and private visibility support.

All workflow parameters live in a single YAML file. An example of configurable parameters for API retrieval is shown in Listing 1.

```
download:
  provider: openalex
  url: "https://api.openalex.org/works?filter=authorships.institu-
tions.lineage:i170726198"
dest: "openalex-fac-tech-sci-2023-2025.csv"
```

Listing 1. .tesci/config.yml YAML intended for API dataset retrieval.

The only obligatory change for a new project is the url under download. Optional keys (schedule, branch, token_env) fall back to defaults so that non-expert users can get started within minutes.

A minimal GitHub Action (`.github/workflows/tesci.yml`) checks out the repository, spins up the CLI and then runs three commands: `tesci download`, `tesci apply`, and `tesci release`. The workflow triggers (i) on every push to main and (ii) once per day. Because each run commits the resulting dataset back to the repository (or a designated results repository), every dataset version is automatically documented and can be referenced with the commit hash or a git tag.

At present, aggregation is a separate row-wise union that preserves duplicates for transparency. Researchers can quickly inspect coverage gaps and obvious overlaps. An upcoming merge mode (see Section 5) will perform DOI-level duplication and fuzzy title matching; until then, the aggregation stage intentionally keeps records untouched so that downstream tools (e.g. VOSviewer, Bibliometrix [11, 13]) can decide how to collapse them.

Figure 1 presents the end-to-end pipeline. Each diagram component is encapsulated in an executable CLI command and orchestrated by GitHub Actions.

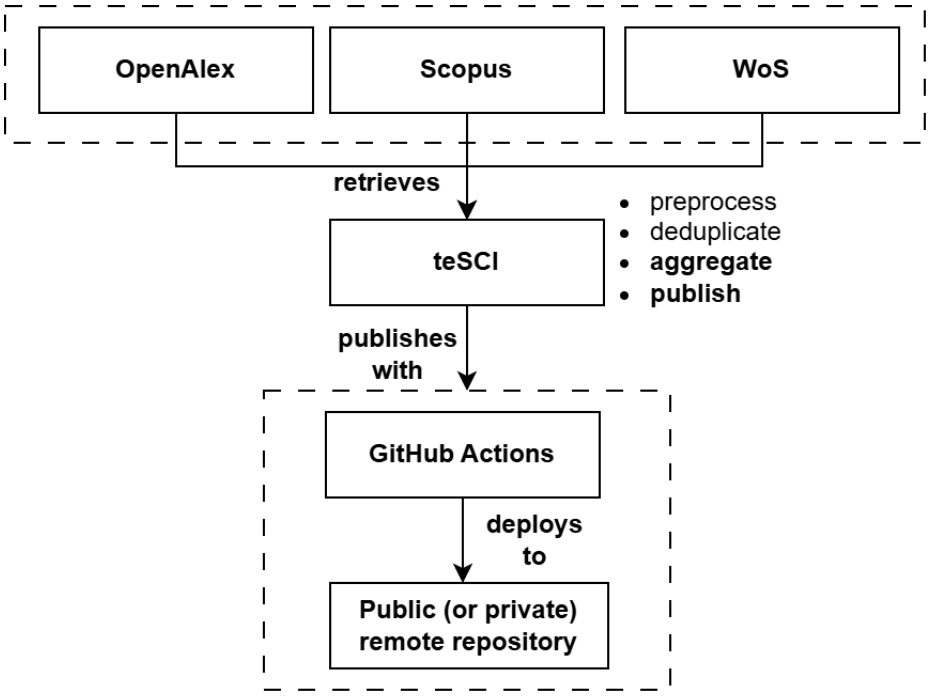

Fig. 1. Workflow of the toolkit. teSCI retrieves records from the three citation databases, performs the preprocessing and aggregation steps, and uses a scheduled GitHub Actions workflow to deploy the resulting dataset to a public or private repository. Bold items are functionalities relevant to the publishing process; non-bold items are functionalities that are planned to be automated.

4 Implementation

The TeslaSCItoolkit codebase[1] contains the Python package `tesci`, whose design maps directly onto the three research questions set out in the introduction. Addressing RQ1, `tesci download` retrieves records from every source declared in `.tesci/config.yml`; OpenAlex is fully supported alongside our existing Scopus and WoS connectors, thereby expanding coverage to the open metadata ecosystem highlighted in [3]. The harvested data is stored on disk and is normalized to a shared schema and passed to `tesci apply`, which performs lightweight aggregations, alongside any other steps outlined in `config.yml` (counts, averages, field groupings). The unified aggregated data is then written to a configurable CSV or Excel output file. A similarity-based merge routine, already robust for Scopus–WoS pairs, will be extended to OpenAlex records in a upcoming release, thereby completing the multi-source deduplication cycle and further strengthening data quality in line with the curation standards [4, 5].

[1] https://github.com/sci2zero/TeslaSCItoolkit

The same sequence satisfies RQ2 and RQ3 by executing standalone runs on GitHub Actions. The default workflow checks out both the research repository and the toolkit, provisions a Python 3.12 environment on the standard runner and invokes the teSCI workflow process. The final artefact is immediately ready for downstream visualization tools such as VOSviewer or bibliometrix. Worked examples in the examples/ directory demonstrate multiple use-case declarative configurations. Looking ahead, we will package the workflow as an official Marketplace Action (`sci2zero/tesci-action`) and add optional chart templates that slot into the same pipeline without additional code, thus lowering the adoption barrier for non-expert teams.

The aggregated file produced by `tesci apply` can be loaded directly into VOSviewer without additional transformation. The current column scheme (Title, Authors, Source, Year, DOI, Citations) follows the recommendations for bibliometric data exchange [4], and is therefore recognized by the import wizard described by VOSviewer. Preliminary tests performed on outputs from our earlier Scopus and WoS study confirmed that co-authorship and keyword maps are generated without manual relabelling. For bibliometrix, which expects a harmonized BibTeX or CSV structure, an initial validation is underway; the most recent development version already ingests the toolkit's CSV after a straightforward type declaration. This suggests that full compatibility can be achieved with minor adjustments. This level of interoperability addresses RQ3 because it allows researchers who are familiar with established visualization environments to adopt the new pipeline while preserving their existing analytical workflows.

The workflow concludes with the `tesci release` command, which collects the aggregated output and publishes them in version control. The command automatically adds a commit or, when the run is triggered from a tagged revision, a corresponding tag. This way every dataset is transparently linked to the exact code and configuration that produced it. When a personal-access token is available, the same step can also publish the artefacts to a separate repository, enabling teams to keep data and code on distinct remotes or to collect results from multiple forks into a shared "results" space. This lightweight mechanism secures a citable snapshot while supporting collaborative contributions and review.

5 Limitations and future work

The current version already fulfils its intended objective: automated retrieval from multiple citation sources, routine aggregation, and a reproducible publishing process with continuous integration. Nevertheless, several elements need additional refinement.

First, while the similarity-based merge and deduplication functions reliably for Scopus and Web of Science, its extension to OpenAlex is still under validation; completing that adaptation is the most immediate technical priority. Second, the toolkit presently provides a clean, consolidated output but leaves the visualization and analysis to external downstream software. Adding lightweight chart templates that integrate directly

into the same GitHub workflow are being prototyped and will be on the roadmap once performance is confirmed across disciplines.

A more systematic evaluation is also on the roadmap. On the roadmap are benchmarks that compare recall, precision, and runtime against representative alternatives. This thereby quantifies the practical implications of the aggregation strategy. In parallel, a small user study could be designed to test the potential configuration models with researchers who have limited programming experience. Feedback from that exercise could inform and help documentation and default settings. While important, this item is further down our roadmap.

On the infrastructural side, preliminary tests indicate that the workflow operates as intended on GitHub with no expected runtime limits for medium-sized data. For substantially larger datasets, we will be experimenting with optional artefact storage solutions and will publish guidance should these become necessary.

Finally, the maintainers intend to distribute the pipeline as an official GitHub Marketplace Action, with the aim of reducing installation to a single configuration line and to lower the barrier to adoption.

6 Conclusion

This work presents an incremental yet practical advance towards reproducible, multi-source bibliometrics. By integrating OpenAlex into an already proven Scopus and WoS pipeline and automating execution through GitHub Actions, the toolkit cuts manual effort to a few YAML lines while keeping every intermediate artefact version-controlled.

The proposed workflow is compatible with current European open-science policies, including the Horizon Europe requirement that research outputs be deposited in trusted repositories together with the software needed to reproduce them. Because each release of the toolkit is identified by a git commit or tag, the corresponding dataset and code snapshot can be exported automatically to Zenodo, where a persistent DOI exists and becomes available for formal citation. In addition, the automated aggregation strategy conforms to the FAIR data principles set out for curated scholarly resources: the resulting files are findable through version metadata, accessible via the public repository or Zenodo record, interoperable in standard tabular formats, and reusable owing to the transparent trustworthy chain advocated by Baas et al. [5]. Future work will therefore be necessary to include community benchmarks that evaluate coverage, accuracy, and runtime across a wider range of disciplines, and external groups are invited to contribute their own use-cases, test cases and performance reports.

Although deduplication and visual analytics are scheduled for the next milestone, the current release already enables rapid, transparent aggregation. Moreover, this provides researchers with a starting point for wider and simpler scientometrics studies. We invite the community to fork, adapt and contribute patches so that future versions can offer one-click merging across all major citation indexes.

References

1. Nikolić D., Ivanović D., Ivanović L. “An open-source tool for merging data from multiple citation databases”. Scientometrics 129, 4573–4595 (2024)
2. Harder R. “Using Scopus and OpenAlex APIs to retrieve bibliographic data for evidence synthesis”. MethodsX (2024).
3. Scheidsteger T., Haunschild R. “Which of the metadata with relevance for bibliometrics are the same and which are different when switching from Microsoft Academic Graph to OpenAlex?” Profesional de la información 32(2) (2023).
4. Donthu N., Kumar S., Mukherjee D., Pandey N., Lim W.M. “How to conduct a bibliometric analysis: An overview and guidelines.” J. Business Research 133, 285-296 (2021).
5. Baas J. et al. “Scopus as a curated, high-quality bibliometric data source for academic research”. Quantitative Science Studies 1(1), 377-386 (2020).
6. openalexR. rOpenSci. Getting bibliographic records from OpenAlex. https://docs.ropensci.org/openalexR/ (2024).
7. OpenAlex. The open catalogue to the global research system. https://openalex.org (accessed May 2025).
8. Huang Y. et al. The open-access coverage of OpenAlex, Scopus and Web of Science. arXiv:2404.01985 (2024).
9. BibexPy. “Harmonising the bibliometric symphony of Scopus and Web of Science”. https://bibexpy.com (accessed May 2025).
10. OpenAlex API Tutorials. GitHub repository ourresearch/openalex-api-tutorials (2024). https://github.com/ourresearch/openalex-api-tutorials (accessed May 2025).
11. Van Eck, Nees, and Ludo Waltman. "Software survey: VOSviewer, a computer program for bibliometric mapping." scientometrics 84.2 (2009): 523-538.
12. Simard, Marc-Andre, et al. "The open access coverage of OpenAlex, Scopus and Web of Science." arXiv preprint arXiv:2404.01985 (2024).
13. Aria, M., and Cuccurullo, C. (2017). bibliometrix: An R-tool for comprehensive science mapping analysis. *Journal of Informetrics, 11*(4), 959–975.
14. ReCiter-Scopus-Retrieval-Tool, https://github.com/wcmc-its/ReCiter-Scopus-Retrieval-Tool (accessed May 2025).
15. PyAlex, https://github.com/J535D165/pyalex (accessed May 2025).