

A python pipeline for interactive visualization of systematic literature review data

Kerem Düzenli^{a,*}, Abdallah Al-Hamad^a, Attila Gilányi^b

^a Doctoral School of Informatics, University of Debrecen, Egyetem tér 1, 4032 Debrecen, Hungary

^b Faculty of Informatics, University of Debrecen, Egyetem tér 1, 4032 Debrecen, Hungary

HIGHLIGHTS

- Automates ETL processing for complex multi-value SLR datasets.
- Integrates Matplotlib, Plotly, and Graphviz via Streamlit UI.
- Generates publication-ready PRISMA flowcharts and Sankey diagrams.

ARTICLE INFO

Keywords:

Systematic literature review
Data visualization
Python
PRISMA
Bibliometrics

ABSTRACT

Systematic Literature Reviews (SLRs) generate complex, multidimensional datasets. A major bottleneck in bibliometric analysis is the prevalence of multi-value cells, which are instances where a single study utilizes multiple tools, methodologies, or platforms. These are typically stored as semicolon-delimited strings in standard spreadsheets. Generic data visualization software struggles to parse these one-to-many relationships natively, forcing researchers to rely on manual data cleaning and external graphic design tools. To address this, we introduce an open-source Python pipeline and interactive Streamlit web application specifically engineered to automate the Extract, Transform, Load (ETL) process for SLR datasets. The software provides a robust tokenization engine that dynamically splits arrays, applies exclusion logic, and maps hierarchical relationships. By abstracting the complexities of three rendering engines (Matplotlib, Plotly, and Graphviz), the tool enables users to generate publication-ready visualizations. These outputs range from automated PRISMA flow diagrams to relational Sankey diagrams and interactive hierarchical charts, all without requiring programming expertise. This methodology significantly reduces the manual overhead of SLR reporting, enhances analytical reproducibility, and aligns with FAIR data principles.

Metadata

Code metadata

Nr.	Code metadata description	Metadata
C1	Current code version	v1.0
C2	Permanent link to code/repository used for this code version	https://github.com/KeremDUZENLI/python-slr-data-visualizer
C3	Permanent link to Reproducible Capsule	
C4	Legal Code License	CC BY-NC-SA 4.0
C5	Code versioning system used	git
C6	Software code languages, tools, and services used	Python, Streamlit, Matplotlib, Plotly, Graphviz
C7	Compilation requirements, operating environments & dependencies	Python 3.8+, Windows/macOS/Linux
C8	If available Link to developer documentation/manual	https://github.com/KeremDUZENLI/python-slr-data-visualizer
C9	Support email for questions	kerem.duzenli@inf.unideb.hu

* Corresponding author.

Email address: kerem.duzenli@inf.unideb.hu (K. Düzenli).

<https://doi.org/10.1016/j.softx.2026.102773>

Received 9 March 2026; Received in revised form 16 May 2026; Accepted 2 June 2026

Available online 11 June 2026

2352-7110/© 2026 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

1. Motivation and significance

Systematic Literature Reviews (SLRs) are fundamental for synthesizing research trends [1]. SLRs can be used to identify relevant evidence in multidisciplinary fields, such as the integration of virtual reality, the metaverse, and digital assets into architectural design and art [2,3]. However, the data extracted from these SLR processes are inherently complex and multidimensional.

While recent advancements have introduced artificial intelligence to automate the initial data extraction process [4], the subsequent transformation and visualization phases remain heavily manual. A critical bottleneck in bibliometric and SLR visualization is the prevalence of multi-value cells. These occur when a single study utilizes multiple tools, platforms, or methodologies. In standard spreadsheet formats (.csv or .xlsx), these values are typically concatenated into a single string separated by delimiters (e.g., “Unity; Blender; Unreal Engine”). Beyond unnesting these arrays, researchers face additional technical challenges: mapping nested hierarchical taxonomies, calculating multi-dimensional co-occurrences, and standardizing PRISMA exclusion logic. Addressing these tasks simultaneously usually requires advanced programming expertise and the use of multiple disparate software environments.

While robust bibliometric mapping tools exist in the R ecosystem [5] primarily for citation network analysis, they often lack the flexibility to parse custom, researcher-defined metadata variables extracted during qualitative reviews. Standard data visualization software, such as Microsoft Excel, and generic Python plotting scripts fail to parse these one-to-many relationships natively. To overcome this limitation, researchers are often forced to manually duplicate rows or write complex data-cleaning scripts to unnest arrays. Furthermore, generating specialized bibliometric figures, such as relational Sankey diagrams, nested hierarchies, or automated PRISMA flowcharts, typically requires alternating between different programming libraries and graphic design software. It is this highly fragmented, time-consuming post-extraction workflow that necessitates a unified, specialized tool. To resolve this, we developed a robust, open-source Python pipeline and interactive web application specifically engineered to automate the Extract, Transform, Load (ETL) process for SLR datasets, enabling the seamless generation of publication-ready visualizations.

2. Software description

2.1. Software architecture

The software architecture is designed around a strictly modular ETL to Visualization pipeline, ensuring that data processing logic is completely decoupled from rendering mechanics (Fig. 1).

To maximize accessibility for both researchers without programming backgrounds and advanced data scientists, the tool is distributed with three distinct usage modalities: Standalone Executable, Interactive Web UI, and a Programmatic API allowing advanced users to execute and batch-generate charts directly from the terminal. The software is designed with minimal computational requirements, operating efficiently on standard consumer desktop hardware (e.g., a dual-core processor and 4GB of RAM) without the need for hardware acceleration. Utilizing the standalone executables allows users to deploy the local Streamlit server without installing a Python runtime or managing package dependencies.

2.2. Software functionalities

The core methodology relies on the `input/` and `operation/` modules, which act as the parsing engine of the software. When a CSV is ingested, the pipeline actively scans specified fields for semicolon delimiters (;). It executes a string tokenization process to separate these arrays, cleans the isolated strings, and dynamically computes unique aggregates for plotting. Data accuracy during this transformation is strictly deterministic; the pipeline utilizes robust combinatorial algorithms to calculate exact cross-frequencies between tokenized arrays.

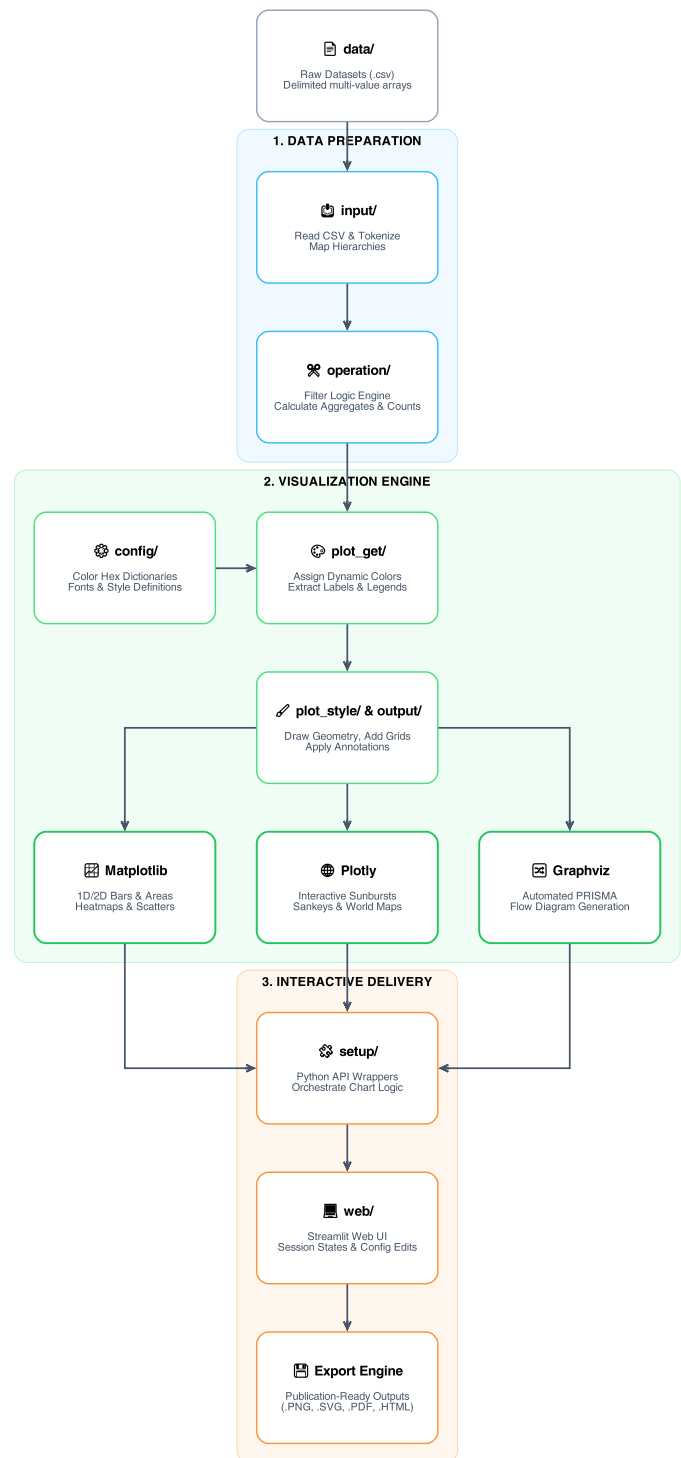


Fig. 1. The modular architecture of the Python pipeline, illustrating the flow from raw CSV extraction, through the ETL processing engine, into the multi-engine abstraction layer, and finally to the interactive delivery outputs.

This guarantees that no relational data is dropped or misrepresented when scaling from flat strings to complex multi-dimensional visualization matrices. Furthermore, the tokenization engine includes built-in fault tolerance for missing or incomplete data; empty cells are safely parsed as explicit empty arrays ([""]), ensuring that all downstream operations execute without computational exceptions.

Furthermore, the web application allows for real-time dataset grouping and hierarchical mapping (Fig. 2). Researchers can

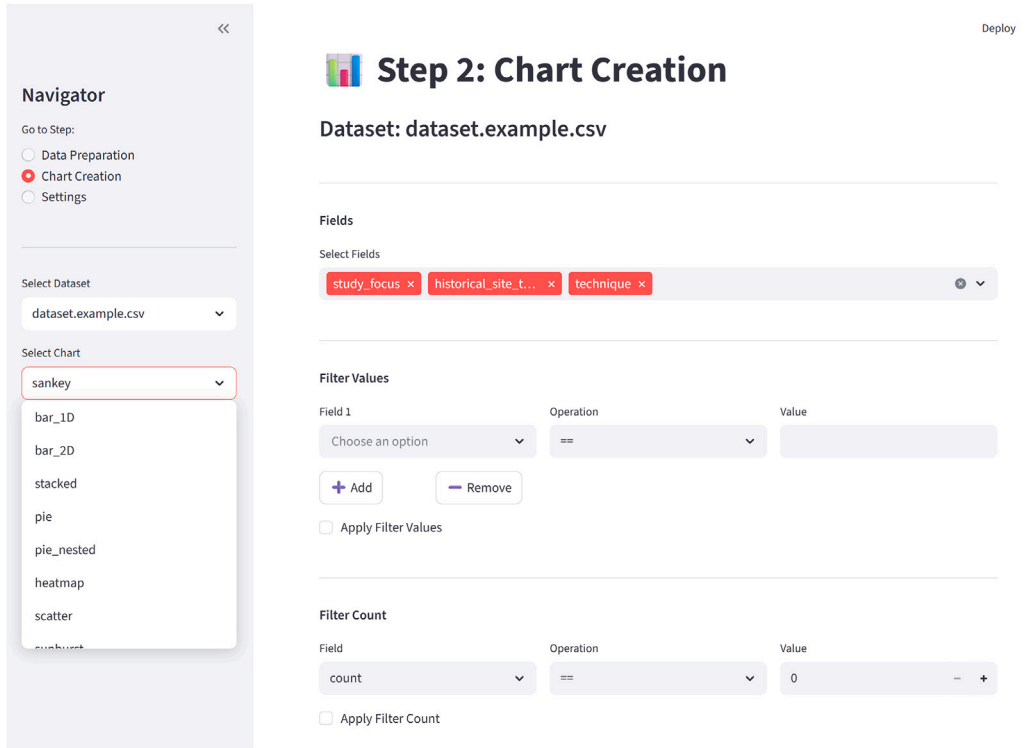


Fig. 2. The Data Preparation interface within the Streamlit application, demonstrating the dynamic grouping and mapping of dataset columns.

programmatically merge discrete columns into a single stackable category without altering the original source file.

Once data is prepared, the Chart Creation interface abstracts the complexities of Matplotlib, Plotly, and Graphviz. It allows users to apply specific conditional filters (e.g., count thresholds) and instantly render complex geometric representations.

3. Illustrative examples

To demonstrate the efficiency and versatility of the pipeline, we processed an example SLR dataset detailing the use of immersive technologies in historical site preservation [6,7]. The virtual reconstruction of cultural heritage sites has become a highly active research area. It utilizes advanced 3D modeling and cognitive infocommunications to preserve historical architecture and present it in collaborative virtual environments [2,8]. The following outputs highlight the software's ability to map these diverse, multi-dimensional data structures into publication-ready formats. Furthermore, all figures presented in this section are direct, unedited exports from the software's rendering engine. By utilizing the tool's integrated export functionality, researchers can seamlessly generate customized, high-fidelity graphics (e.g., natively as vector PDFs or SVGs) to guarantee visual crispness and satisfy stringent publication standards.

Tracking inclusion and exclusion criteria across hundreds of studies manually is prone to arithmetic errors and formatting complexities. The Graphviz integration [9] natively calculates nodes based on sequential filter queries, such as applying exclusion logic constraints, and renders standard PRISMA reporting [10] without the need for manual vector drafting.

When analyzing methodologies, individual papers often employ multiple overlapping techniques. The tokenization engine seamlessly splits these one-to-many arrays and maps them across multiple dimensions (e.g., Study Focus → Site Type → Technique). This process preserves the correct weight of interconnected workflows and renders them in a Plotly-based Sankey diagram (Fig. 3).

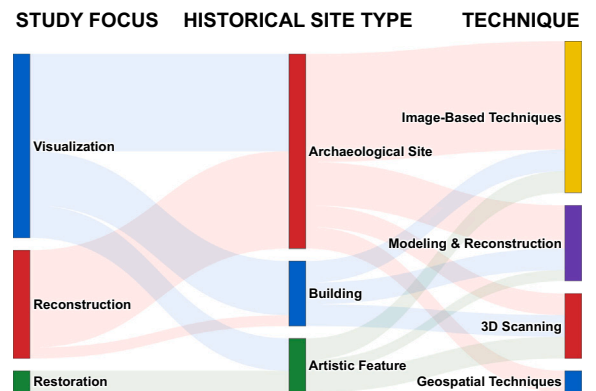


Fig. 3. A Sankey diagram demonstrating the software's ability to tokenize multi-value cells and map three-dimensional data relationships.

By utilizing the data preparation logic, the pipeline effortlessly aggregates disparate sub-categories (e.g., Software Data vs. Software Modeling). It computes the total occurrences and renders them alongside pre-configured categorical colors using the Matplotlib abstraction layer [11] (Fig. 4).

To visualize parent-child taxonomies extracted from bibliometric metadata, the software constructs interactive, nested hierarchical rings. For example, broad site classifications are programmatically parsed into distinct architectural sub-types based on dataset mapping rules (Fig. 5).

Identifying tool and technique synergy is a core requirement of modern SLRs. The pipeline computes occurrence intersections from fragmented strings to reveal methodological clusters. These intersections are then mapped instantly to intensity gradients for rapid visual analysis (Fig. 6).

To identify gaps in global academic literature, the pipeline incorporates multi-dimensional scatter logic. By plotting intersecting

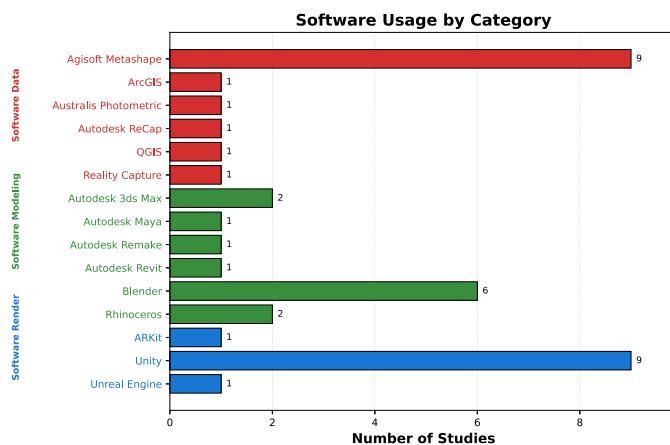


Fig. 4. A horizontal bar chart illustrating grouped software usage.

Historical Site Type & Sub-Type Distribution

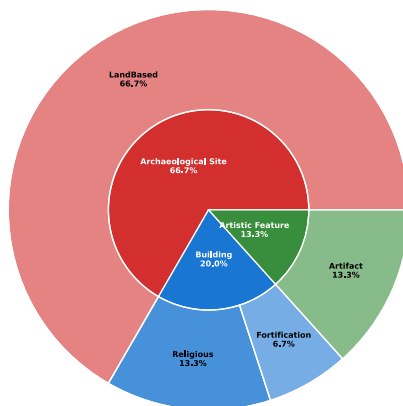


Fig. 5. A nested pie chart mapping parent-child categorical hierarchies.

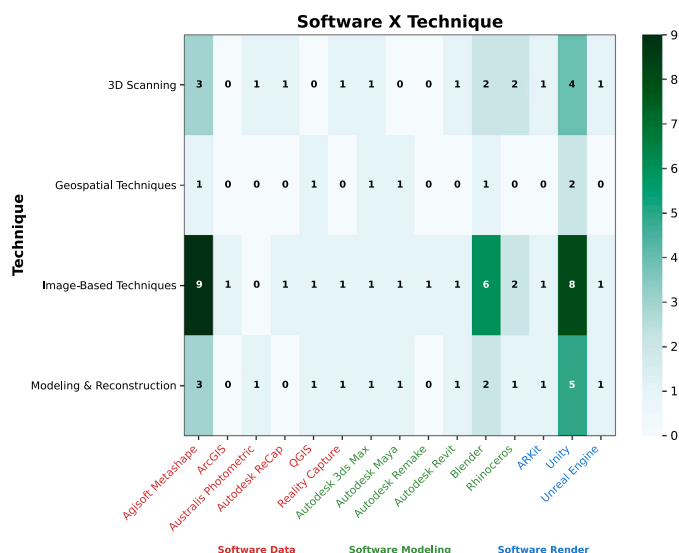


Fig. 6. A heatmap visualizing the co-occurrence frequency of various software combinations against extracted techniques.

categorical fields (Country vs. Site Type) and scaling the node markers by tokenized frequency, the software highlights geographical research biases.

4. Impact

The visualization of Systematic Literature Review data has traditionally required extensive manual data parsing and the use of disparate graphic design tools. Recent methodological advancements in data science heavily favor automated, reproducible pipelines using robust frameworks like Pandas [12]. By engineering a custom Python pipeline that natively parses multi-value delimited cells, we have bridged the gap between raw bibliometric extraction and publication-ready visualization.

This software creates a significant impact by eliminating the manual overhead associated with formatting complex arrays, tracking exclusion logic, and drawing PRISMA flows. Furthermore, the methodology can be easily adapted to other complex bibliometric domains, ranging from historical architecture to broader science mapping workflows [5]. The integration of a Streamlit web interface makes this advanced ETL architecture highly accessible to researchers across all scientific disciplines without requiring programming expertise.

5. Conclusions

Ultimately, this software provides a robust, reproducible, and standardized methodology for charting complex academic data. It significantly accelerates the systematic review reporting process while strictly adhering to open science and FAIR data principles [13]. Specifically, by replacing undocumented manual data manipulation with a transparent ETL pipeline, the software ensures analytical reusability. Researchers can publish their raw datasets alongside the exported chart configurations, allowing the broader scientific community to perfectly reproduce and verify complex visualizations without relying on proprietary tools.

CRedit authorship contribution statement

Kerem Düzenli: Writing – original draft, Visualization, Software.
Abdallah Al-Hamad: Project administration.
Attila Gilányi: Supervision, Project administration.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the authors used ChatGPT-4 to improve readability and language. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research has been supported by project no. 152101, implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the ADVANCED_25 funding scheme.

Supported by the University of Debrecen Program for Scientific Publication. This research has been supported by project no. 2024-1.2.3-HU-RIZONT-2024-00030, implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the 2024-1.2.3-HU-RIZONT funding scheme.

References

- [1] Kitchenham B. Procedures for performing systematic reviews. Keele University and NICTA, Tech Rep. 2004.
- [2] Bekele MK, Pierdicca R, Frontoni E, Malinverni ES, Gain J. A survey of augmented, virtual, and mixed reality for cultural heritage. *J Comput Cult Herit* 2018;11(2):1–36.

- [3] Düzenli K, Perdahçı NZ. The impact of metaverse and NFTs on architecture: how will the buildings of the future take shape? *EKSEN Dokuz Eylül Üniversitesi Mimarlık Fakültesi Dergisi*. 2023. <https://doi.org/10.58317/eksen.1286569>.
- [4] Felizardo KR, Steinmacher I, Lima MS, Deizepe A, Conte TU, Barcellos MP. Data extraction for systematic mapping study using a large language model - a proof-of-concept study in software engineering. In: *Proceedings of the 18th ACM/IEEE international symposium on empirical software engineering and measurement, ESEM '24*, Association for Computing Machinery, New York, NY, USA; 2024. pp. 407–13. <https://doi.org/10.1145/3674805.3690743>.
- [5] Aria M, Cuccurullo C. bibliometrix: an R-tool for comprehensive science mapping analysis. *J Informetr* 2017;11(4):959–75.
- [6] Düzenli K, Gilányi A. Stochastic VR reconstruction advisor: a probabilistic workflow recommendation engine. In: *2025 IEEE 16th international conference on cognitive infocommunications (CogInfoCom)*; 2025. pp. 000249–54. <https://doi.org/10.1109/CogInfoCom66819.2025.11200596>.
- [7] Düzenli K, Albazooni AM, Al-Hamad A, Gilányi A. A systematic review of utilizing extended reality technologies for historical site preservation and visualization. *Results Eng* 2026;30, 110592 <https://doi.org/10.1016/j.rineng.2026.110592>.
- [8] Gilányi A, Bujdoso G, Balint M. Virtual reconstruction of a medieval church. In: *2017 8th IEEE international conference on cognitive infocommunications (CogInfoCom)*; 2017. 283–8. <https://doi.org/10.1109/CogInfoCom.2017.8268257>.
- [9] Ellson J, Gansner E, Koutsofios L, North SC, Woodhull G. Graphviz and dynagraph – static and dynamic graph drawing tools. In: *Graph drawing: 9th international symposium*; 2001. 15–26. Springer.
- [10] Page MJ, McKenzie JE, Bossuyt PM, Boutron I, Hoffmann TC, Mulrow CD, Shamseer L, Tetzlaff JM, Akl EA, Brennan SE. The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. *BMJ* 2021;372.
- [11] Hunter JD. Matplotlib: a 2D graphics environment. *Comput Sci Eng* 2007;9(3):90–5.
- [12] McKinney W. Data structures for statistical computing in Python. *Proceedings of the 9th Python in science conference*, Austin, TX; 2010. 56–61, Vol. 445.
- [13] Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, Blomberg N, Boiten J-W, da Silva Santos LB, Bourne PE. The FAIR guiding principles for scientific data management and stewardship. *Sci Data* 2016;3(1):1–9.