SemRepo: A Knowledge Graph for Research-Connected GitHub Repositories

Abdul Rafay[1], David Lamprecht[2], and Michael Färber[1]

[1] ScaDS.AI, TU Dresden, Germany
[2] metaphacts GmbH, Walldorf, Germany
abdul.rafay@mailbox.tu-dresden.de
michael.faerber@tu-dresden.de
dl@metaphacts.com

Abstract. We present *SemRepo*, an RDF knowledge graph containing over 81 million triples about nearly 200,000 GitHub repositories linked to scientific research. *SemRepo* captures fine-grained key repository metadata – contributors, issues, dependencies, and languages – and connects repository authors to profiles in *SemOpenAlex* as well as repositories to *LinkedPapersWithCode* and *MLSea-KG*. Released under CC0, the data are available as RDF dumps and as a public SPARQL endpoint. *SemRepo* enables diverse use cases, such as tracing the evolution of research ideas into software, discovering research-backed repositories for reuse, analyzing collaboration patterns between academia and development, tracking the downstream impact of publications, and identifying expertise based on code contributions. By unifying code, papers, and authors in a single, queryable graph, *SemRepo* transforms fragmented silos into a transparent resource for the Semantic Web community.

Data & Services:	https://semrepo.org https://w3id.org/semrepo
Code:	https://github.com/abdulrafay97/SemRepo
Data License:	Creative Commons Zero (CC0)
Code License:	MIT License

Keywords: Code Repositories, Scholarly Data, Open Science

1 Introduction

Software now supports nearly every phase of modern science, but the scientific record and the code that operationalizes it are still scattered across incompatible silos. Bibliographic databases meticulously log papers and authors, while developer platforms capture commits, issues and dependency graphs – but almost nothing semantically connects these two. The result is a pervasive blind spot: reviewers cannot verify that the "reference implementation" cited in a paper even exists; funding agencies cannot track whether award-winning ideas have become actively maintained libraries; and meta-researchers cannot chart how

methods spread across disciplines or identify the teams driving that spread. These fractures complicate tests of replicability, slow the transfer of innovations into production environments, and obscure macro-level trends that influence strategic decisions. What is missing is a semantic bridge that connects publications, repositories, and contributors into a single knowledge space – one capable of answering questions like "Which research-driven projects have achieved more than 1,000 GitHub stars, and in which venues were they first introduced?"

Scholarly KGs such as the Microsoft Academic Knowledge Graph (MAKG) [12] and SemOpenAlex [14] richly describe publications and authors, while software-centric graphs like SemanGit [22] and RCGraph [30] focus on repository statistics. LinkedPapersWithCode (LPWC) [13] provides *links* from papers to GitHub, but the repositories, including information about the contributors, issues, dependencies, are not semantically modeled. Consequently, cross-domain SPARQL queries that traverse the research-implementation boundary are still impossible.

Recent scholarly knowledge graphs such as the Open Research Knowledge Graph (ORKG) aim to semantically represent research contributions, modeling elements like research problems, methods, and results [20,2]. However, they do not provide integration with code repositories or contributors, which limits their utility in bridging conceptual research insights with their operational implementations. Similarly, initiatives like Wikidata and WikiCite attempt to represent scholarly metadata in a structured, community-curated form, but they are constrained by scalability issues and lack in-depth modeling of software artifacts [31].

Other open infrastructures such as OpenCitations [26] and the OpenAIRE Graph [23] focus on open bibliographic data and citation networks. While these are valuable for tracking scientific influence and fostering transparency, they do not semantically interlink code, issues, or development contributions. AceKG [32], for instance, offers an RDF graph of academic entities, but is closed in updates and does not model code. These gaps highlight the need for a unified semantic graph that supports cross-domain queries over papers, contributors, and software—enabling new forms of impact assessment and discovery at the interface of research and development.

In this paper, we introduce *SemRepo*, an RDF knowledge graph that semantically models 197,566 GitHub repositories including fine-grained metadata – such as contributors, issues, dependency graphs, and licences – yielding *82 million triples*. *SemRepo* links all 197,566 repositories to LinkedPapersWithCode Repository entities and 148,185 repositories to MLSea Software entities. Furthermore, it contains 11,867 author links (6%) to *SemOpenAlex*, forming the first coherent graph that systematically spans publications, code, and people.

This integration unlocks several high-impact applications: (i) *Open-source collaboration analytics*—identify key contributors, form project teams, and map cross-lab cooperation; (ii) *Issue-resolution recommenders*—match open issues to domain experts and surface engaged community members; (iii) *Research-industry bridges*—trace company uptake of cutting-edge papers and quantify technology transfer; (iv) *Expertise and trend dashboards*—rank developers by

scholarly influence, monitor language/package adoption, and forecast emerging topics.

Overall, our main contributions are as follows:

1. We provide a public knowledge graph under CC0 of 82M triples covering research-connected GitHub repositories.[3]
2. We propose an OWL ontology with 18 classes and 46 relations, plus VoID metadata for LOD compliance.
3. We develop and evaluate an open-source pipeline for entity and author linking.[4]
4. We outline case studies on reproducibility auditing, collaboration analytics, and trend mining.

The rest of the paper is structured as follows: We outline related works in 2 and describe the construction of *SemRepo* in Sec. 3, including the created ontology in Section 3.1 and an evaluation in Sec. 3.5. We discuss use cases in Sec. 4 and conclude in Sec. 5.

2 Related Work

Existing efforts to model scholarly knowledge and software artifacts offer valuable partial views but fall short of bridging the research-implementation gap. Scholarly graphs such as the Microsoft Academic Knowledge Graph (MAKG) [12] and SemOpenAlex [14] provide rich descriptions of publications, venues, and authors, enabling large-scale scientometric analyses. However, they operate almost entirely in the bibliographic domain. Software-centric graphs like SemanGit [22] and RCGraph [30] focus on repository-level features (e.g., commits, forks, stars) but lack structured links to the papers that motivated the code, or to the contributors beyond platform usernames.

Some initiatives do attempt cross-linking. LinkedPapersWithCode (LPWC) [13] connects machine learning papers to their GitHub repositories, providing valuable bridges between research claims and implementations. Yet, the underlying repositories remain opaque: contributors, dependency trees, licences, and issue discussions are not semantically represented, which precludes expressive queries across research and development.

Recent projects like the Open Research Knowledge Graph (ORKG) [20,2] focus on fine-grained modeling of research contributions – such as problems, methods, and results – using structured templates and community curation. While ORKG helps formalize scientific discourse, it does not integrate software artifacts or developer identities. Similarly, Wikidata and WikiCite [31] offer community-driven, cross-domain structured data but suffer from scalability limitations and lack dedicated models for software repositories.

[3] See URLs on page 1 and `https://doi.org/10.5281/zenodo.15399468` for the dump files.

[4] `https://github.com/abdulrafay97/SemRepo`

Open infrastructures like OpenCitations [26] and the OpenAIRE Graph [23] center on bibliographic transparency and citation networks. Though essential for tracking scholarly impact, they do not incorporate repositories or development workflows. AceKG [32] includes an RDF representation of academic entities but is not openly maintained and excludes code-related metadata.

The software engineering community has contributed large-scale datasets such as GHTorrent [18,19] and GitHub Archive, which track repository activity, collaboration events, and commit histories. These support empirical software analysis but are not aligned with scholarly entities, and they lack semantic schemas for research-method links or academic provenance.

Efforts like Papers with Code [29] go a step further by curating links between machine learning papers and corresponding implementations. However, these links are shallow and often manually curated, with no semantic model to unify code metadata, contributor roles, or project evolution. Similarly, institutional KGs like VIVO [21] model academic profiles and affiliations but are disconnected from open-source development.

In summary, current solutions either model scholarly communication or software ecosystems – but rarely both in a unified, semantic way. The absence of cross-domain expressiveness limits our ability to assess reproducibility, trace knowledge transfer, or quantify the scholarly impact of code. Our work addresses this gap by introducing a semantic graph that coherently interlinks papers, repositories, and contributors, enabling richer analyses and smarter research infrastructure.

3 SemRepo

For creating and evaluating the *SemRepo* knowledge graph, we followed a structured, multi-step approach:

1. **GitHub URL Extraction:** We began by extracting all GitHub repository links from the knowledge graph *LinkedPapersWithCode* (LPWC)[13] – regardless of whether they were linked to a research paper or dataset. After validating and preprocessing the URLs, we identified a total of 197,566 unique valid repositories.
2. **Metadata Scraping:** Over a period of 21 days, we scraped metadata from the identified repositories and stored the results in JSON format. Each JSON file includes detailed information such as the repository name, URL, description, topics, programming languages, and packages used in the code. It also captures interaction metrics (stars, forks, issues, contributors), author profiles and affiliations, and the full *readme* content to preserve usage context.[5]
3. **Knowledge Graph Modeling:** In correspondence with the collected metadata, we defined the ontology for *SemRepo* in OWL.

[5] A sample is available at `https://github.com/abdulrafay97/SemRepo/blob/main/assets/khuangaf_awesome-chart-understanding.json`.

4. **RDF Knowledge Graph Construction:** The metadata was transformed into RDF triples to form a structured knowledge graph. To ensure a tight integration into the existing semantic scholarly landscape, *SemRepo* is linked to both Linked Papers With Code, SemOpenAlex and MLSea-KG. Our Knowledge Graph, released alongside this paper, is available at Zenodo [1].
5. **Evaluation:** We evaluated the resulting knowledge graph in terms of quality, completeness, and accuracy, verifying the correctness of metadata mapping and semantic consistency.
6. **Analysis and Validation:** We conducted exploratory queries and analyses to validate the graph's usability and to extract meaningful insights about the software landscape in scientific research.

In the following, we describe the ontology and the interlinking to other knowledge graphs in more detail. To ensure long-term usability, we provide the full update pipeline for *SemRepo.org* as open-source scripts, enabling periodic regeneration of the knowledge graph. Updates are currently scheduled every three months, following the same proven procedures used for maintaining our previous graphs such as SemOpenAlex.

3.1 SemRepo Ontology

The SemRepo ontology represents a comprehensive semantic framework designed to model software repositories and their associated metadata, encompassing a structured vocabulary of 19 entity types and 47 object- and datatype-properties as illustrated in Figure 1. The ontology's core architecture centers around primary entity types including repositories, issues, packages, language references, persons, and organizations, establishing intricate relationships through properties such as sr:hasIssue, sr:hasContributor, and sr:hasTopic, while incorporating datatype properties for descriptive metadata including repository titles, abstracts, and statistical measures. This comprehensive modeling approach enables detailed representation of software development ecosystems, capturing both the technical artifacts and the social dynamics inherent in collaborative software development environments.

The ontology demonstrates strategic alignment with established semantic web standards through systematic integration with widely-adopted vocabularies, including Dublin Core Terms (dcterms) for bibliographic metadata, FOAF (Friend of a Friend) for person and organization descriptions, FABIO for bibliographic resources, and OWL for ontological constructs. Furthermore, the schema indicates sophisticated integration capabilities with domain-specific ontologies such as LinkedPapersWithCode (lpwc) and SemOpenAlex (soa), suggesting robust potential for cross-domain knowledge linking and interoperability. The complete specification is formally published as an OWL file, accompanied by a VoID (Vocabulary of Interlinked Datasets) description that documents dataset statistics and interlinks in adherence to Linked Open Data best practices. This architectural approach enables the SemRepo ontology to leverage existing semantic infrastructure while providing specialized modeling capabilities for software

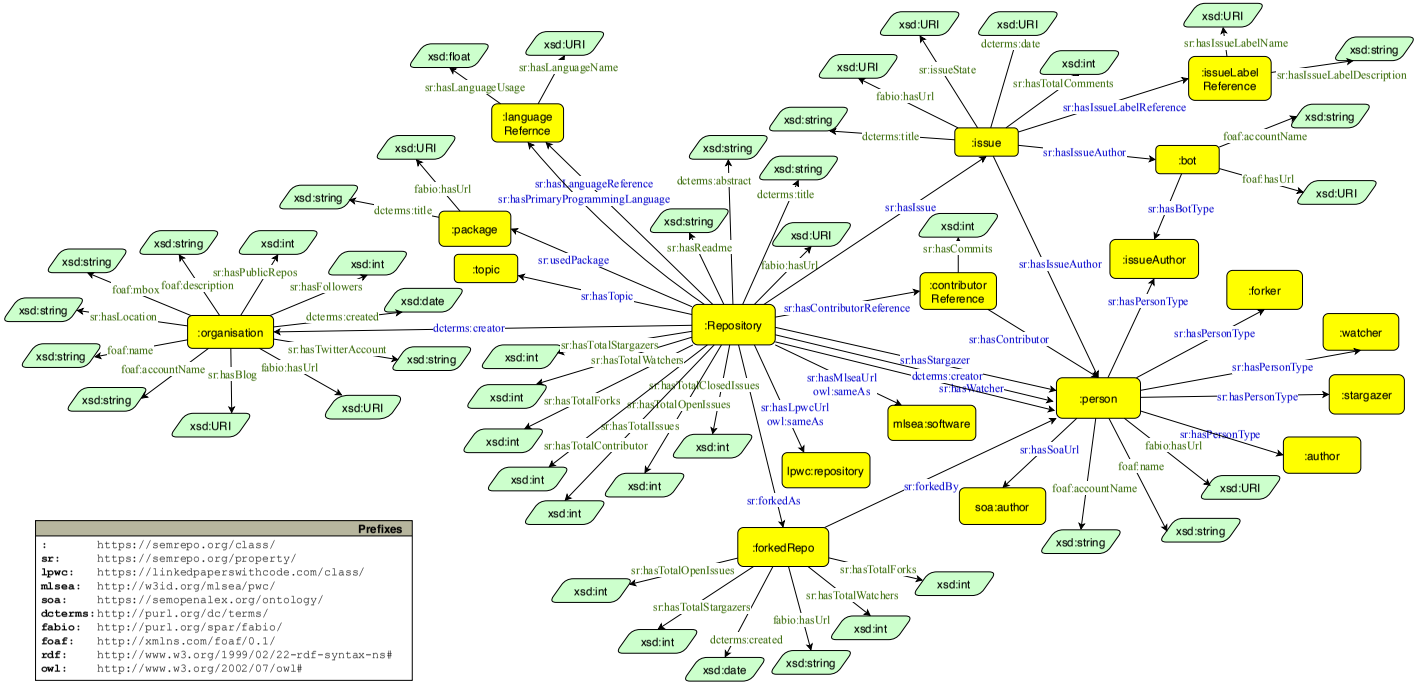


Fig. 1: Schema of SemRepo

repository ecosystems, thereby facilitating enhanced discoverability, analysis, and integration of software development artifacts within the broader semantic web landscape.

Two modeling challenges required the use of n-ary relation patterns [16,33] to adequately capture the semantics of the data. N-ary relations are a well-established modeling approach used when simple binary relationships are insufficient, for instance, when we need to model the strength, certainty, or relevance of the relationships, as well.

Contributor activity: In modeling contributor activity, the relation between a GitHub repository and a user can initially be expressed as a simple binary relation (e.g., `sr:hasContributor`). However, such modeling fails to reflect the extent of an individual's contribution (e.g., number of commits). To capture this, we introduce the auxiliary class `sr:ContributorReference`, allowing for the representation of contribution magnitude. This class acts as an intermediate node that links a repository and a contributor, while additionally specifying the number of commits. This design preserves quantitative details of the contribution and supports fine-grained analyses, such as identifying top contributors.

Language usage: Projects frequently employ several languages in their repositories, each with a distinct share of the code base (e.g. `Python 70 %`, `C++ 25 %`). We therefore introduce the class `sr:LanguageReference` that links a repository and a language and annotates this relationship with its usage percentage. These auxiliary nodes preserve the quantitative context enabling precise queries such as identifying "repositories where Rust accounts for more than 50% of the code base."

3.2 Linking to *Linked Papers With Code*

Linked Papers With Code (LPWC)[6] [13] offers a large knowledge base of research papers – primarily in the research field of machine learning and computer science – together with their associated GitHub repository URLs, thereby supporting reproducibility and rapid re-use of new methods. Because LPWC served as the primary source for the repositories in *SemRepo*, we establish an explicit linkage between all 197,566 repositories of the two knowledge graphs.

3.3 Linking to *MLSea-KG*

The MLSea-KG[7] [10,11] is a large-scale, declaratively constructed knowledge graph that integrates metadata from major machine learning repositories, such as OpenML, Kaggle, and Papers with Code, capturing over 1.44 billion RDF triples related to ML datasets, experiments, pipelines, software, and scientific works. In total 148,185 repositories in SemRepo (75%) are linked to MLSea software entities, tightly interconnecting these two semantic knowledge bases.

[6] `https://linkedpaperswithcode.com`
[7] `https://w3id.org/mlsea`

3.4 Linking to *SemOpenAlex*

Linking *SemRepo* to *SemOpenAlex* (SOA)[8] [14] is performed through a fully automated, six-stage pipeline. First, for each LPWC-matched repository, the corresponding GitHub URL is retrieved from the LPWC repository index. Second, this GitHub URL is used to query the LPWC paper index in order to obtain the URI of the associated scholarly work. Third, the list of authors of the identified paper is extracted. Fourth, each author name is submitted to *SemOpenAlex* to retrieve the corresponding `soa:Author` URI, if available. Fifth, the returned author names are matched against GitHub usernames recorded in *SemRepo*. Sixth, in cases of exact matches, we add an RDF triple of the form `<semrepo:Person> sr:hasSoaUrl <soa:Author>` to establish a semantic link between the *SemRepo* person and the authoritative SOA profile.

Overall, this process links 11 867 contributors – representing approximately 6% of all persons in *SemRepo* – to enriched *SemOpenAlex* entities. This integration enables complex queries that combine software development activity with traditional bibliometric metadata.

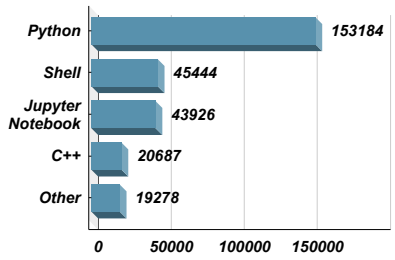
3.5 Analysis

Table 1 reports instance counts for key classes across SemRepo (`sr:`), LPWC (`lpwc:`), and SemOpenAlex (`soa:`). In total, SemRepo comprises *82,078,636* RDF triples. We can see that `sr:Repository` accounts for nearly 200,000 entries, while over 2.9 million `sr:Person` instances reflect the rich contributor landscape. The high number of `sr:Issue` (2.6M) and `sr:Forked Repository` (2.4M) entities underscores active project maintenance and reuse, respectively. Notably, 197,566 repositories are successfully linked to LPWC, and 11,867 contributors are associated with SemOpenAlex author profiles, evidencing strong integration across scholarly and developer graphs.

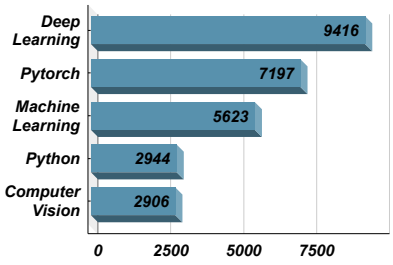
Table 2 contrasts open and closed issues – an immediate proxy for repository maintenance health. Roughly 72% of issues are closed, suggesting that most projects maintain at least a basic level of responsiveness to bug reports and feature requests.

Figure 2 reveals four macro-level patterns that emerge from simple SPARQL queries. The language panel (Fig. 2a) shows that Python remains the undisputed lingua franca of research-driven repositories, while `Rust` is the only language exhibiting a marked upward trend, signaling growing interest in safe, high-performance systems for machine learning workflows. The topic distribution (Fig. 2b) indicates that diffusion-based generative models now dominate the research conversation, having vaulted from a niche idea to the most prevalent theme in only a few release cycles. Finally, the organisational chart (Fig. 2c) highlights a pronounced concentration of activity: industry-backed actors such as `Meta Research` command the largest footprints in both repository count and community engagement, highlighting their influence in stewarding open-source

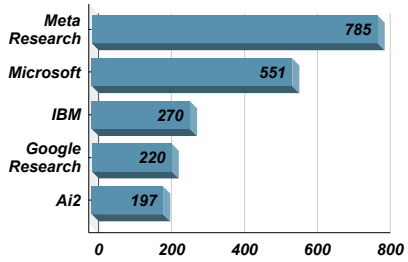
[8] `https://semopenalex.org`



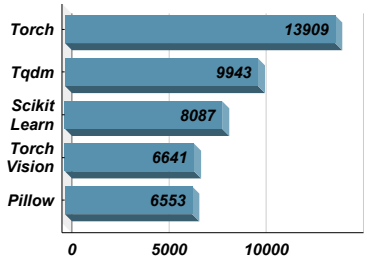
(a) Top 5 used programming languages



(b) Top 5 topics of GitHub repositories



(c) Top 5 organizations with most GitHub repositories



(d) Top 5 used packages in the domain of Machine learning

Fig. 2: Analysis of the SemRepo Knowledge Graph

ML tooling. Finally, in the package view (Fig. 2d), `PyTorch` and its surrounding ecosystem lead by a wide margin, confirming its role as the de-facto experimentation framework for deep learning. Collectively, these observations demonstrate how the unified semantics of *SemRepo* enable rapid, evidence-based assessments of technology trends, dominant tooling, and the key institutions driving open-source innovation.

Table 3 summarizes the most frequent issue labels in SemRepo, reflecting common development tasks and challenges. Tables 4 and 5 report contributor activity based on commit counts, with Table 5 restricted to contributors linked to SemOpenAlex profiles. While a substantial number of contributors are matched, not all GitHub users could be aligned with scholarly author records, highlighting limitations in cross-graph identity resolution.

Table 6 reports collaboration counts between contributors, based on co-authorship or shared repository contributions extracted via SPARQL queries. For instance, Kai Chen's
`https://semrepo.org/person/hellock` top five collaborations are shown, with 154 joint contributions alongside Lizz. This analysis captures the collaborative

Table 1: SemRepo entity types and number of instances

Entity Type	# Instances
sr:Repository	197,566
sr:Issue	2,609,510
sr:Organisation	12,879
sr:Package	95,505
sr:Forked Repository	2,468,660
sr:Person	2,916,508
sr:Topic	272,378
sr:Language	387,284
sr:Bot	36

Table 2: Distribution of open/closed issues in the knowledge graph

Issue State	# Instances
Open	729,917
Closed	1,879,593

Table 3: Top 5 Issue labels by Number of Counts

Issue Label Name	# Count
Bug	178,490
Enhancement	129,114
Question	98,861
Stale	47,187
Help wanted	27,612

structure of SemRepo and demonstrates the graph's suitability for querying social dynamics. Sample SPARQL queries are provided online in our repository.[9]

4 Use Cases of SemRepo

SemRepo fills a critical gap in the ecosystem of scholarly knowledge graphs and code repository platforms. Existing efforts such as SemOpenAlex richly model publications and authors [14], while software-centric graphs like SemanGit [22]

[9] https://github.com/abdulrafay97/SemRepo/blob/main/sparql-queries/queries.pdf

Table 4: Top 5 Contributors by Number of Commits overall

Person	# Commits
https://github.com/tensorflower-gardener	223,456
https://github.com/AlexeyAB	122,898
https://github.com/glenn-jocher	74,498
https://github.com/shelhamer	65,424
https://github.com/lattner	63,346

Table 5: Top 5 Contributors by Number of Commits with SemOpenAlex profile

SemOpenAlex Profile	GitHub username	# Commits
https://semopenalex.org/author/A5023786468	shelhamer	65,424
https://semopenalex.org/author/A5002742256	rusty1s	21,388
https://semopenalex.org/author/A5040282669	hirofumi0810	18,841
https://semopenalex.org/author/A5045561693	lemire	15,348
https://semopenalex.org/author/A5062139851	mponce0	13,038

Table 6: Top 5 collaborators of "Kai Chen"

Person	# Collaborations
https://semrepo.org/person/innerlee	154
https://semrepo.org/person/xvjiarui	144
https://semrepo.org/person/ychfan	132
https://semrepo.org/person/thangvubk	132
https://semrepo.org/person/lzhbrian	128

and RCGraph [30] focus on isolated metadata about repositories. Services like GHTorrent [18] stream raw event data via bespoke APIs, but none of these offer a semantically linked, queryable bridge between research outputs and their corresponding software implementations. By publishing 81.5M RDF triples and interlinking with LinkedPapersWithCode [13] and MLSea [10] in the Linked Open Data cloud [6,4], SemRepo enables seamless integration and federation, providing a robust framework for the following high-impact use cases:

Reproducibility Auditing One of the most pressing challenges in contemporary research is the lack of reproducibility. SemRepo directly addresses this by enabling automated SPARQL queries that link a paper's GitHub URL (via `fabio:hasURL` in LPWC) to its corresponding repository and metadata such as forks, stars, and contributors. Through its RDF structure, it becomes possible to detect abandoned or inactive forks within months of publication—tasks that would be extremely labor-intensive using siloed or API-based solutions. Such capabilities enable reproducibility audits at scale [17].

Collaboration Orchestration SemRepo models contributor–repository interactions as a bipartite graph, enabling community detection and collaboration analytics [24]. By computing early-phase activity ratios (e.g., commits vs. issue comments), teams can be categorized into productivity archetypes such as "toilers," "communicators," and "collaborators" [28]. Federated queries with GHTorrent data [18] further allow real-time tracking of project dynamics such as star surges or contributor drop-offs, supporting outreach and intervention strategies.

Research–Industry Synergy SemRepo's integration with LPWC and SemOpenAlex enables seamless linkage between research papers and their implemen-

tation in code. This facilitates a tighter feedback loop between academia and industry [25], as companies can discover research-backed tools, and researchers can track industrial adoption. Empirical studies show that such linking efforts can boost co-citation rates and foster more impactful collaborations [9].

Expertise Mapping and Skill Discovery Thanks to its fine-grained contributor modeling (e.g., `sr:contributesTo`, `sr:usesPackage`), SemRepo supports advanced profiling of developers and teams. Approaches such as DevRank [15] or skill vector models [3] can be executed directly on the graph. Recruiters and research institutions can search for developers matching specific patterns (e.g., "GraphQL contributors with Rust experience and >10 PRs in 90 days"), a task which graph-based filtering performs significantly better than keyword search [5].

Trend and Impact Forecasting SemRepo captures temporal metadata such as `sr:starredAt` and `sr:forkedAt`, which allows for longitudinal analysis of project lifecycles [7]. Combined with citation velocities from LPWC, these signals can help predict which projects are likely to gain traction. For instance, repositories linked to papers that exceed 20 citations within six months are significantly more likely to enter the top decile in GitHub popularity [8]. Language trends and library transitions (e.g., TensorFlow to PyTorch or Torch 2.0) can also be monitored in near-real time [27].

5 Conclusion

In this paper, we presented *SemRepo*, a large-scale semantic knowledge graph that captures fine-grained metadata from nearly 200,000 GitHub repositories linked to scientific research. By systematically connecting software artifacts with scholarly entities such as publications and authors, SemRepo transforms fragmented information into a coherent, machine-queryable space—bridging the longstanding gap between research and practical implementation. SemRepo enables a wide range of high-impact applications, including reproducibility auditing, expertise and talent discovery, trend analysis across machine learning domains, and tracking the downstream influence of academic research on open-source software development.

As future work, we aim to extend SemRepo beyond GitHub by integrating additional software platforms such as GitLab or Bitbucket, broadening the coverage of open-source activities. Furthermore, incorporating more granular metadata—such as commit histories, pull requests, and code review interactions—would enable even deeper insights into the dynamics of software development and collaboration.

References

1. Abdul Rafay, D.L., Färber, M.: Semrepo.org (2025). `https://doi.org/10.5281/zenodo.15399468`, `https://doi.org/10.5281/zenodo.15399468`, dataset released alongside the paper

2. Auer, S., Oelen, A., Haris, M., Stocker, M., D'Souza, J., Farfar, K.E., Vogt, L., Prinz, M., Wiens, V., Jaradeh, M.Y.: Improving access to scientific literature with knowledge graphs. Bibliothek Forschung und Praxis **44**(3), 516–529 (2020)
3. Baltes, S., Diehl, S.: Towards a theory of developer expertise. In: 2018 IEEE International Conference on Software Maintenance and Evolution (ICSME). pp. 74–84 (2018). https://doi.org/10.1109/ICSME.2018.00018
4. Berners-Lee, T.: Linked data—design issues. https://www.w3.org/DesignIssues/LinkedData.html (2006)
5. Bird, C., Nagappan, N., Gall, H.: Don't touch my code! examining the effects of ownership on software quality. In: Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering. pp. 4–14 (2011). https://doi.org/10.1145/2025113.2025119
6. Bizer, C., Heath, T., Berners-Lee, T.: Linked data—the story so far. International journal on semantic web and information systems **5**(3), 1–22 (2009). https://doi.org/10.4018/jswis.2009081901
7. Borges, H., Hora, A., Valente, M.T.O.: Understanding the factors that impact the popularity of github repositories. In: 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME). pp. 334–344 (2016). https://doi.org/10.1109/ICSME.2016.31
8. Coelho, J., Valente, M.T.O., Aniche, M.: The evolution of repository popularity in github. Empirical Software Engineering **25**(5), 4305–4332 (2020). https://doi.org/10.1007/s10664-020-09844-8
9. Crowston, K., Howison, J.: Leveraging open source communities to support evidence-based software engineering. In: 2015 IEEE/ACM 12th Working Conference on Mining Software Repositories. pp. 483–486 (2015). https://doi.org/10.1109/MSR.2015.70
10. Dasoulas, I., Yang, D., Dimou, A.: Mlsea: a semantic layer for discoverable machine learning. In: European Semantic Web Conference. pp. 178–198. Springer (2024)
11. Dasoulas, I., Yang, D., Dimou, A.: Mlseascape: Search over machine learning metadata empowered by knowledge graphs. In: European Semantic Web Conference. pp. 193–196. Springer (2024)
12. Färber, M.: The microsoft academic knowledge graph: A linked data source with 8 billion triples of scholarly data. In: The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part II 18. pp. 113–129. Springer (2019)
13. Färber, M., Lamprecht, D.: Linked papers with code: the latest in machine learning as an rdf knowledge graph. arXiv preprint arXiv:2310.20475 (2023)
14. Färber, M., Lamprecht, D., Krause, J., Aung, L., Haase, P.: Semopenalex: the scientific landscape in 26 billion rdf triples. In: International Semantic Web Conference. pp. 94–112. Springer (2023)
15. Fu, B., Zhang, M., Shang, L., Ma, J.: Devrank: Mining influential developers in github. In: 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE). pp. 464–474. IEEE (2017). https://doi.org/10.1109/ASE.2017.8115655
16. Giunti, M., Sergioli, G., Vivanet, G., Pinna, S.: Representing n-ary relations in the Semantic Web. Logic Jouornal of the IGPL **29**(4), 697–717 (2021)
17. González-Barahona, J.M., Robles, G.: On the reproducibility of empirical software engineering studies based on data retrieved from development repositories. Empirical Software Engineering **20**(2), 429–465 (2015). https://doi.org/10.1007/s10664-014-9329-5

18. Gousios, G., Spinellis, D.: Ghtorrent: Github's data from a firehose. In: Proceedings of the 9th Working Conference on Mining Software Repositories. pp. 12–21 (2012). https://doi.org/10.1109/MSR.2012.6224294
19. Gousios, G., Vasilescu, B., Serebrenik, A., Zaidman, A.: Lean ghtorrent: Github data on demand. In: Proceedings of the 11th working conference on mining software repositories. pp. 384–387 (2014)
20. Jaradeh, M.Y., Oelen, A., Farfar, K., Prinz, M., D'Souza, J., Stocker, M., Auer, S.: Open research knowledge graph: Next generation infrastructure for semantic scholarly knowledge. In: Proceedings of the 10th International Conference on Knowledge Capture (K-CAP). pp. 243–246. ACM (2019). https://doi.org/10.1145/3360901.3364435
21. Krafft, D.B., Cappadona, N.A., Caruso, B., Corson-Rikert, J., Devare, M., Lowe, B.J., Collaboration, V., et al.: Vivo: Enabling national networking of scientists (2010)
22. Kubitza, D.O., Böckmann, M., Graux, D.: Semangit: A linked dataset from git. In: The Semantic Web–ISWC 2019. pp. 215–228. Springer (2019)
23. Manghi, P., Mannocci, A., La Bruzzo, S., Atzori, C., Bardi, A., Artini, M., Principe, P., Schirrwagen, J.: The openaire research graph (2021)
24. Moradi-Jamei, B., Kramer, B.L., Calderón, J.B.S., Korkmaz, G.: Community formation and detection on github collaboration networks. arXiv preprint arXiv:2109.11587 (2021)
25. Pautasso, C., Alonso, G., Nussbaumer, B.: Software engineering research for the world wide web: methods, tools, and opportunities. IEEE Software **34**(3), 28–35 (2017). https://doi.org/10.1109/MS.2017.80
26. Peroni, S., Shotton, D.: Opencitations, an infrastructure organization for open scholarship. Quantitative Science Studies **1**(1), 428–444 (2020)
27. Ray, B., Posnett, D., Filkov, V., Devanbu, P.: A large-scale study of programming languages and code quality in github. In: Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. pp. 155–165 (2014). https://doi.org/10.1145/2635868.2635922
28. Saadat, S., Newton, O.B., Sukthankar, G., Fiore, S.M.: Analyzing the productivity of github teams based on formation phase activity. arXiv preprint arXiv:2011.03423 (2020)
29. Taylor, R., Stojnic, R., et al.: Papers with code. https://paperswithcode.com (2019)
30. Venigalla, A.S.M., Ali, M.S., Manjunath, N., Chimalakonda, S.: Rcgraph - a tool to integrate readme and commits through temporal knowledge graphs. In: 2023 IEEE/ACM 31st International Conference on Program Comprehension (ICPC). pp. 30–34 (2023). https://doi.org/10.1109/ICPC58990.2023.00014
31. Waagmeester, A., Stupp, G., Burgstaller-Muehlbacher, S., Good, B.M., Griffith, M., Griffith, O.L., Hanspers, K., Hermjakob, H., Hudson, T.S., Hybiske, K., et al.: Wikidata as a knowledge graph for the life sciences. Elife **9**, e52614 (2020)
32. Wang, R., Yan, Y., Wang, J., Jia, Y., Zhang, Y., Zhang, W., Wang, X.: Acekg: A large-scale knowledge graph for academic data mining. In: Proceedings of the 27th ACM international conference on information and knowledge management. pp. 1487–1490 (2018)
33. World Wide Web Consortium: Defining N-ary Relations on the Semantic Web (2006), https://www.w3.org/TR/swbp-n-aryRelations/