

Addressing Predicate Redundancy in Research Knowledge Graphs: Duplicate Detection, Resolution, and Prevention

Lena John¹[0009–0007–2097–9761], Sushant Aggarwal², Sören Auer¹[0000–0002–0698–2864], and Oliver Karras¹[0000–0001–5336–6899]

¹ TIB - Leibniz Information Centre for Science and Technology, Hannover, Germany

² Leibniz University Hannover, Germany

{lena.john, soeren.auer, oliver.karras}@tib.eu
sushant.aggarwal@stud.uni-hannover.de

Abstract Research Knowledge Graphs (RKGs) enable the structured representation of scientific knowledge, but their weakly enforced schemas make them prone to inconsistencies, particularly in how predicates are defined and used. Duplicate predicates, i.e., distinct identifiers expressing the same or highly similar relationships, introduce semantic redundancy, hinder reuse, and reduce RKG quality. While prior work has addressed duplicate detection for downstream tasks such as query answering or schema alignment, predicate redundancy as a data quality challenge, remains underexplored, particularly in terms of resolution, prevention, and semi-automated curator support.

In this paper, we propose a framework for managing duplicate predicates in RKGs that covers detection, resolution, and prevention. The framework combines automated similarity-based methods with human validation and is designed for integration into the lifecycle of evolving, crowdsourced RKGs. We implement the framework in the context of the Open Research Knowledge Graph (ORKG) by extending its existing curation dashboard SciKGDash with embedding-based clustering, interactive inspection, and resolution actions such as merging and deleting.

We evaluate the framework on the ORKG, where clustering reveals that up to 30% of predicates are potentially redundant. The analysis also shows recurring modeling patterns that lead to predicate redundancy, user-induced duplication, inconsistent identifier usage, and a lack of standardization in predicate naming and usage.

Our findings demonstrate that duplicate predicates arise from user behavior and interface design. Addressing this, requires combining automated methods with human-centered curation and preventive mechanisms. This work positions predicate redundancy as a central data quality challenge and provides a foundation for more systematic and proactive RKG curation.

Keywords: Duplicate Predicates · Knowledge Curation · Research Knowledge Graphs · Quality Assurance

1 Introduction

Knowledge Graphs (KGs) model entities and their relationships in a semantically enriched graph structure, enabling both human and machine interpretability [11]. Research Knowledge Graphs (RKGs) have emerged as a key technology for representing scientific knowledge across various domains [31].

Their weakly enforced schemas and flexible nature support the integration of heterogeneous and evolving knowledge, but also introduce data quality challenges such as ambiguities, missing values, and duplicates [11,12,13]. Under the open world assumption (OWA) [10], (R)KGs are inherently incomplete [33]. Crowdsourced RKGs amplify these challenges by promoting decentralized and community-driven knowledge creation [22], which can lead to inconsistent modeling practices and evolving schemas [29]. Ensuring semantic consistency therefore requires continuous curation beyond initial data integration [12].

A central yet often overlooked source of inconsistency lies in predicates. While entities can often be aligned to external references, predicates define how these entities are semantically related and thus shape the meaning of (R)KGs. Divergent modeling practices frequently result in *duplicate predicates*, i.e., distinct identifiers expressing the same or highly similar relationships [17,23]. These duplicates propagate semantic redundancy across all triples using them [8], reducing RKG quality and utility [20]. Despite advances in related areas such as ontology alignment [5,34] and schema mapping [14], predicate redundancy remains largely underexplored as a data quality challenge.

This work addresses this challenge by proposing a framework for the detection, resolution, and prevention of duplicate predicates. The framework is informed by existing literature and practical curation experience in the Open Research Knowledge Graph (ORKG) [3], a crowdsourced RKG for structured and machine-actionable scientific knowledge. We implemented the framework within SciKGDash [17], the existing curation dashboard of the ORKG. We tested the detection and resolution actions of the framework in a sandbox instance of the ORKG that mirrors the live data, enabling safe experimentation and roll-back while preserving real-world characteristics, and subsequently used the live system to derive practical insights into predicate redundancy and its curation.

Our findings indicate that a substantial portion of predicates (up to 30%) are potentially redundant and stem from recurring modeling patterns, including repeated creation of predicates with identical labels and descriptions but different identifiers by the same users, confusing identifiers with labels, and the absence of canonical forms for frequently used predicates.

The contribution of this work is twofold: (1) a structured framework for managing predicate redundancy in RKGs, and (2) a practical demonstration of its applicability in a real-world setting.

The remainder of this paper is structured as follows: Section 2 introduces the background knowledge. Section 3 presents related work. Section 4 presents the framework. Section 5 outlines the implementation. Section 6 presents the preliminary evaluation. Section 7 discusses the findings. Section 8 concludes the paper, together with future work.

2 Background: What is a duplicate predicate?

With duplicate predicates we refer to distinct predicates with different identifiers that encode the same or highly similar semantic relationship. We distinguish between syntactic similarity (labels, URIs), structural similarity (graph patterns), semantic similarity (equivalence in meaning and use), and ontological similarity (explicit formal relations). In this context, predicate uniqueness refers to the degree to which an RKG avoids multiple representations of the same underlying relation [10]. When RKGs are populated, data (i.e. symbols) often transforms into information (i.e. connected data), and finally into knowledge (i.e. interpreted meaning) [27]. Duplicate predicates arise when information is not consistently abstracted into knowledge. Two predicates may differ at the data layer, but result into the same knowledge representation.

With regard to RKG quality, duplicate predicates fall under succinctness and more specifically conciseness, including both *extensional conciseness* (redundant instances) and *intensional representational conciseness* (multiple schema elements serving the same purpose) [6,11]. Low conciseness ultimately leads to unnecessary proliferation of predicates without adding expressive power, thereby reducing overall structural clarity.

This issue is closely tied to a modeling trade-off between expressivity and abstraction. RKGs with fine-grained predicate sets increase expressivity and modeling flexibility, but introduce graph fragmentation and reduce predicate overlap, which negatively affects querying and inference [8]. Conversely, more unified predicate sets improve overlap and reasoning capabilities, at the cost of semantic nuance. This tension is further complicated by the OWA, under which missing information cannot be interpreted as false or complete. However, duplicate predicate detection requires a notion of completeness. Therefore, we adopt a closed world assumption (CWA) for this task, restricting the analysis to the observed predicate set and treating it as complete within the modeled scope [10].

3 Related Work

This section is based on an [ORKG comparison](#) [16] of 8 papers on approaches to duplicate detection, resolution, and prevention in KGs and Linked Data. ORKG comparisons provide a structured, tabular overview of contributions, enabling them to be compared along relevant properties. The comparison shows that prior work primarily focuses on duplicate detection (7 out of 8) to support downstream tasks such as query expansion, schema alignment, and KG completion. For instance, Abedjan et al. [1] perform schema analysis such as range-based filtering, while Kalo et al. [20,21] and Niazmand et al. [26] apply rule mining and data-driven methods to identify semantically related predicates for query answering. Huaman et al. [13] are providing an overview for further duplicate detection tools. In particular, ontology alignment approaches, e.g., within the [OAEI initiative](#), address schema similarity across ontologies and include methods for predicate matching based on lexical, structural, and embedding-based

techniques [5,8,30]. However, these approaches typically assume multiple ontologies and focus more strongly on entity alignment than on predicate redundancies within a single, evolving graph. Predicate matching remains challenging due to high lexical variability and heterogeneous representations [30].

Although these methods identify semantically related predicates, they primarily treat similarity as a means to improve consumption tasks rather than as a data quality challenge, with few exceptions, such as Salem et al. [28]. Approaches that explicitly consider duplicate resolution (2 out of 8) and prevention (2 out of 8) remain limited [13]. Notable examples include clustering-based predicate recommendation, based on paper title and abstract, by Oghli et al. [2] in the ORKG and the GDup framework by Manghi et al. [24], which supports detection, grouping, and merging with curator feedback. However, semi-automated approaches involving users or curators remain underexplored.

Overall, existing work rarely treats duplicate predicates as a first-class problem and lacks integrated support for detection, resolution, and prevention within the lifecycle of evolving, crowdsourced (R)KGs. In contrast, we address predicate redundancy as a data quality challenge targeting RKG curation and creation, and propose a reusable, human-centered framework covering all three stages.

4 Framework

To address predicate redundancy in especially crowdsourced RKGs, we propose an iterative framework consisting of Duplicate Detection (DD), Duplicate Resolution (DR), and Duplicate Prevention (DP). Based on our experience as contributors and curators of the ORKG, as well as our prior work on [SciKGDash](#) [17] with the ORKG Curation & Community Building team we encountered numerous duplicate predicates, which made it difficult to consistently reuse existing terminology. From a curation perspective, we identified it as a systemic issue: Users tend to create new predicates rather than reuse ones from the already noisy predicate set, further increasing redundancy. This motivated both the cleanup of existing predicates and the prevention of future duplicates, drawing on prior work on duplicate detection and resolution, especially by Manghi et al. [24].

The framework, as illustrated in Figure 1, follows an iterative process suited to continuously evolving RKGs with frequent data imports and contributions. DD and DR are inherently sequential, as duplicates must first be identified before they can be resolved, and are applied iteratively in practice. DD provides step-wise guidance for identifying duplicate predicates across four levels (cf. subsection 4.1), progressively incorporating richer contextual information. Ontological similarity may be absent in early iterations and can emerge through resolution decisions. DR suggests context-dependent resolution actions (cf. subsection 4.2), while final decisions remain under human control. DP operates at a broader level and can be initiated at any time, but is most effective after DD and DR insights are available. These insights, such as root causes of duplication, usage patterns, and modeling behavior, inform targeted interventions in RKG design, user guidance, and data import processes. Resolving duplicates improves consistency,

reducing the likelihood of future duplicate creation and prevention effort over time. The framework combines automated support where possible with human oversight where necessary. It involves multiple roles [17,23], that may overlap. Individuals may act as contributors, curators, and builders depending on the task at hand and level of automatization. Contributors create content and potentially duplicates, curators analyze predicate usage and perform graph cleaning, builders refine interface design and modeling guidance.

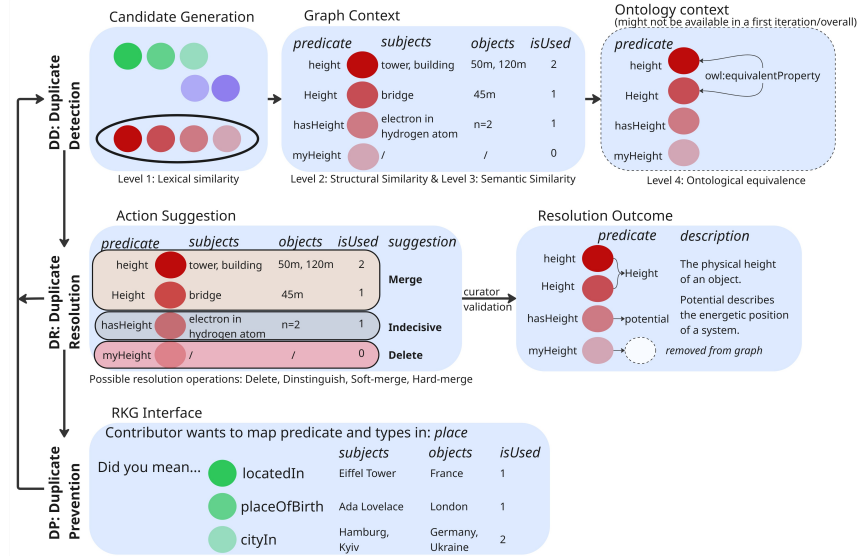


Figure 1: Framework for Duplicate Predicate Detection, Resolution, and Prevention. Detection uses stepwise similarity levels, resolution is guided by extensional usage and human validation, prevention is integrated into RKG import pipelines.

4.1 Duplicate Detection (DD)

Duplicate detection can be formulated as a filtering problem with a precision–recall trade-off, supported by human curator validation. We define four levels ranging from lexical similarity to ontological equivalence (see Table 1), where each level provides increasing confidence in predicate equivalence. Candidate generation starts with Level 1 (lexical similarity), which is computationally efficient and maximizes recall by identifying predicates with similar labels or URIs. This step produces clusters of potential duplicates, but is prone to ambiguity and false positives due to polysemy. To improve precision, Levels 2 and 3 incorporate structural and semantic context, including subject/object types, graph usage patterns, and provenance. This filters out predicates that are similar at the surface level but differ in meaning, shifting from intensional similarity to extensional, usage-based similarity. Since this stage often requires contextual interpretation, human validation is used to resolve ambiguous cases. Level 4 leverages ontological information to identify explicit equivalences. However, such information is often unavailable a priori and may instead emerge as a result of resolution

decisions in DR. Based on these results, refined candidate groups are passed to DR for final curator validation.

Table 1: Levels of evidence for duplicate predicate detection. Higher levels indicate stronger justification for equivalence, lower levels may detect false positives.

Level & Evidence	Likely Duplicates	False Duplicates
1. Lexical: label/URI similarity	<code>method/Method</code> (naming variant)	<code>location/location</code> (place vs. URL)
2. Structural: same graph usage	<code>Product hasPrice value.</code> <code>Product cost value.</code> (identical types)	<code>Website head Title.</code> <code>Person headOf Company.</code> (different domains and ranges)
3. Semantic: contextual meaning	<code>Event period xsd:integer.</code> <code>Event length xsd:integer.</code> (used interchangeably)	<code>Sensor tempC xsd:decimal.</code> <code>Sensor tempF xsd:decimal.</code> (different units)
4. Ontological: formal equivalence	<code>foaf:name</code> <code>owl:equivalentProperty</code> <code>schema:name.</code>	<code>capitalOf</code> <code>rdfs:subPropertyOf</code> <code>locatedIn.</code>

4.2 Duplicate Resolution (DR)

The system proposes resolution actions for detected duplicate predicates. Curators select candidate predicates and apply one of four actions shown in Table 2: Delete, distinguish, soft-merge, or hard-merge. As these actions modify the RKG, they are restricted to authorized curators. The *delete* action removes unused predicates from the RKG. The *distinguish* action refines labels or metadata to disambiguate semantically different predicates. For schemas, W3C standards do not provide explicit mechanisms to declare predicate inequality. OWL 2 [punning](#) allows predicates to be related via `owl:differentFrom`, but this is limited to individual instances and does not support reasoning across predicates. The merge action is applied either as soft-merge or hard-merge. *Soft-merge* introduces ontological equivalence (e.g., `owl:equivalentProperty` or `owl:sameAs` using punning) while preserving existing statements when confidence for hard-merge is limited. *Hard-merge* replaces all occurrences of duplicate predicates with a canonical root predicate and removes redundant predicates from the RKG.

4.3 Duplicate Prevention (DP)

DP begins already during the DD and DR phases by considering patterns. Because predicates form the semantic layer of an RKG, their extensional usage provides key insights into intended meaning. By analyzing creation and usage patterns, supported by workflow backtracking, usage analytics, and curation dashboards [17], we can identify common causes of duplicate predicates. Preventing duplicates therefore requires addressing these causes within the RKG interface and ingestion workflow, reducing the need for post-hoc DD and DR.

Based on our prior [17] and present experiences, as well as general insights from literature [13], typical sources of duplicate predicates include:

- **Limited visibility:** Existing predicates are hard to discover or browse.
- **Weak documentation:** Missing or unclear labels and usage guidance.
- **Low reuse incentives:** Creation is easier than reusing existing predicates.
- **Insufficient safeguards:** Creation is not checked against similar predicates.
- **Import noise:** External data sources introduce duplicates during ingestion.

To mitigate these issues, RKG systems should provide real-time predicate suggestions, similarity-based warnings, usage analytics, and guided mapping during data import, especially during semantic enrichment phases [18].

Table 2: Overview of resolution actions and their modifications to the RKG.

Action	Example Predicate(s)	Resolution action	Outcome
Delete	<code>tempLabel</code> (unused)	Remove predicate	Predicate is removed from the RKG (only if unused).
Distinguish	<code>degree</code> (temperature) vs. <code>degree</code> (qualification)	Refine labels or add meta-data	Predicates remain, but are semantically disambiguated.
Soft-merge	<code>period/length</code>	Add <code>period</code> <code>owl:equivalentProperty</code> <code>length</code> .	Establishes ontological equivalence; candidates for later hard-merge.
Hard-merge	<code>method/Method</code>	Replace with root predicate <code>Method</code> → <code>method</code> , <code>Method</code> removed	All instances mapped to one predicate; duplicates removed.

5 Implementation

We implemented the proposed framework in the environment of the crowd-sourced [ORKG](#) to demonstrate its practical applicability. To support DD and DR, we extend the curation dashboard [SciKGDash](#) [17], whose [source code](#) is publicly available. A dedicated [predicate curation interface](#) integrates the DD and DR workflows. DP is currently under development and planned for integration into the ORKG frontend and backend.

[SciKGDash](#) consists of a Python backend, a PostgreSQL database, and a Next.js frontend. The DD pipeline retrieves predicates from ORKG, concatenates their labels and available descriptions into a single text sequence, embeds them using Sentence Transformers ([all-MiniLM-L6-v2](#)), and stores the results in the database. Predicate similarity is computed via cosine similarity and a k -nearest neighbor graph ($k = 50$, threshold 0.65) is constructed to capture semantically related predicates. These parameters were chosen based on exploratory experiments to balance connectivity among highly similar predicates with meaningful clustering. Clustering is performed using [Louvain community detection](#), which ensures full assignment of predicates to clusters and scales efficiently to large graphs [7], in contrast to density-based methods that may leave points unassigned. To improve coverage of lexical duplicates beyond the k -neighborhood

($k = 50$), predicates with identical normalized labels are explicitly linked prior to clustering. An approximate global coherence score based on cluster sizes relative to the total number of predicates is displayed to motivate incremental curation progress. For each cluster, lightweight heuristic rules provide low-cost initial curation suggestions. Predicates with zero usage and outdated creation timestamps are flagged for deletion, while identical labels indicate merge candidates. Remaining clusters are marked for manual review, enabling an initial screening before applying more expensive context-based similarity analysis.

Clusters are visualized as graph and list views. The graph view (see Figure 2) uses the [react-force-graph library](#), which creates a two-dimensional network layout that produces dense visual structures with closely positioned nodes. Curators can inspect cluster-based predicates and metadata, including its impact on the global coherence score, labels, descriptions, usage statistics, and linked subjects and objects. A key consideration specific to ORKG is, that predicates can occur in any position within a triple, which is reflected in the breakdown provided in the number of uses column (see Figure 3).

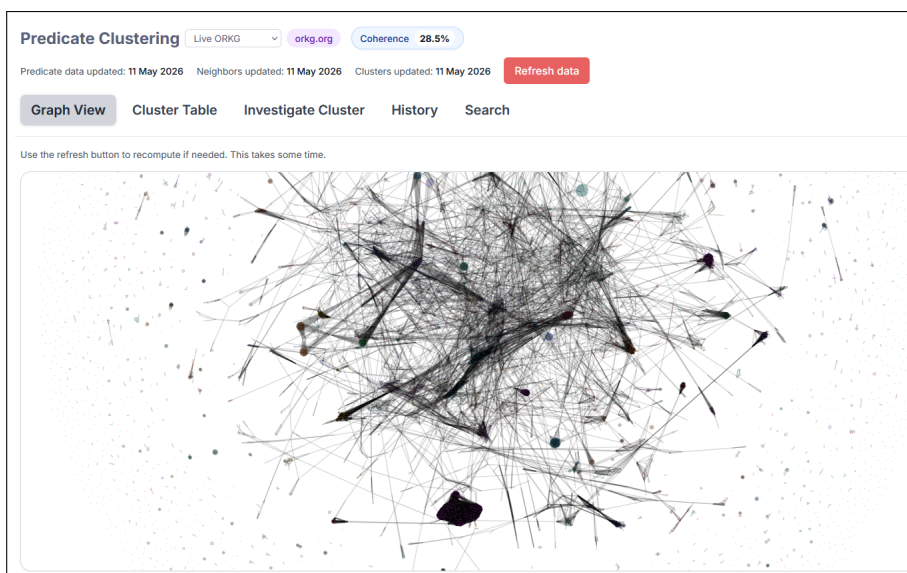


Figure 2: Initial graph view of the k -nearest neighbor ($k = 50$) similarity graph (cosine similarity of 0.65) and its coherence score. Nodes represent predicates, edges indicate high semantic similarity, and clusters group potential duplicates.

An advanced suggestion mode refines a selected cluster by re-embedding predicates with the now available, extended contextual information of subjects and objects. Pairwise cosine similarity is then computed using a higher threshold (0.8), focusing on high-confidence semantic similarity, whereas the lower threshold (0.65) used in graph construction preserves broader connectivity. The resulting similarity graph in one cluster is partitioned into connected components via depth-first search, which are interpreted as fine-grained suggestion groups and

labeled using heuristics based on usage, label similarity, and shared authorship, yielding candidate decisions such as merge, delete, or reviews (see Figure 3).

Group 38 (2)

delete

avg similarity: 90.0%

ID	Label	Description	Number of Uses	Created At	Last Used	Created By	Subjects	Objects	Suggestion
wikidata:P170	creator	The creator/author of this CreativeWork. This is the same as the Author property for CreativeWork.	0 (no uses)	2 April 2024		Farzaneh			delete
P49019	creator	The creator/author of this CreativeWork. This is the same as the Author property for CreativeWork.	0 (no uses)	14 April 2022		Allard Oelen			delete

Group 39 (2)

merge

avg similarity: 90.0%

ID	Label	Description	Number of Uses	Created At	Last Used	Created By	Subjects	Objects	Suggestion
P37334	Venue		12 (no uses)	27 May 2021		Qamar Abbas			merge
HAS_VENUE	venue		25111 (no uses)	3 March 2020		unknown			merge

Figure 3: Advanced suggestions with finer-grained groups based on contextual embeddings and higher similarity threshold (0.8) for a merge and a delete group.

The DR workflow allows authenticated curators to select predicates and apply actions on the production or sandbox ORKG instance. For merging, a root predicate is selected as the canonical target. As ORKG does not employ a formal ontology and already uses the **sameAs** predicate, it is reused for soft-merges to declare equivalence. Since **sameAs** does not enforce structural properties such as symmetry or transitivity [4], it is stored as a statement from the source predicate to the root. Hard-merges additionally replace all occurrences of merged predicates in statements with the root predicate and remove the unused predicates. Level 4 of DD is implemented prototypically. Ontological connections via **sameAs** are not yet visualized in the UI, but are nonetheless considered during hard-merges to prevent invalid self-references (e.g., **A sameAs A**). Distinguish actions are currently supported via manual label and description editing in ORKG, while enhanced UI and (LLM-supported) metadata editing are planned.

Due to pipeline latency, there is a temporary discrepancy between backend and UI. Curator actions are recorded in a history tab and, until manual pipeline re-execution, modified predicates are filtered out to avoid inconsistent states. Automatically re-running the pipeline after each change was deemed inefficient.

For DP, we aim to enhance the autocomplete for predicate selection, currently limited to string matching (see Figure 4a), directly into the ORKG interface. We have [started extending](#) the ORKG backend with [Neo4j’s vector index](#) and plan to integrate [Sentence Transformers for Spring](#), enabling similarity-based predicate suggestions. As ORKG is a mature system, these changes require coordination with the team due to the system’s complexity. To explore the approach, a simplified version (see Figure 4b) is implemented in the curation dashboard in the [search tab](#), where existing predicate embeddings are used.

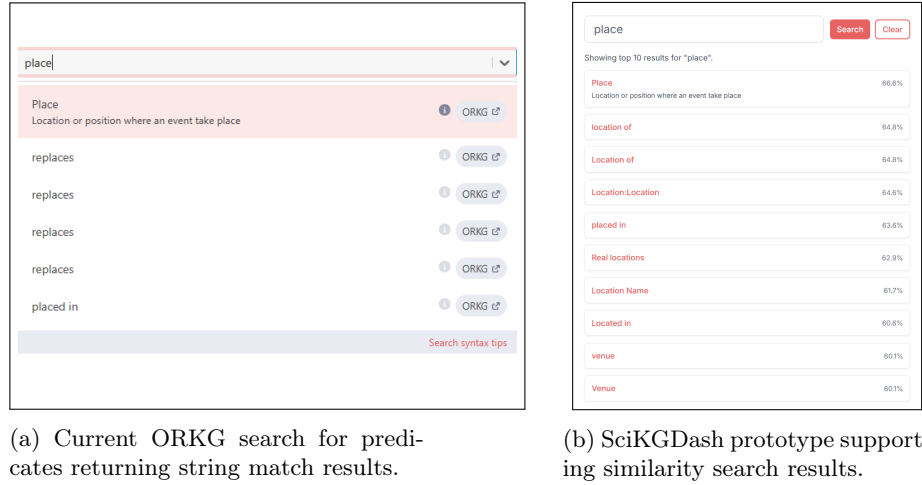


Figure 4: Concept of embedding-based similarity search for duplicate prevention.

6 Evaluation

Analysis. We conduct a feasibility study to evaluate the current state of predicate data in the ORKG and assess the potential of our framework for DD and DR. The study outlines the technical capabilities of our implementation and focuses on identifying common modeling practices that lead to duplicate predicates, demonstrating how our approach can improve RKG quality. Since the framework operates on live graph data, a careful technical assessment is required before exposing it to general curators. Establishing a solid technical foundation is therefore a prerequisite for broader adoption by the ORKG curators. To support the evaluation, we developed a [script](#) that analyzes clustering results stored in the database. It computes cluster statistics, including size distributions and estimates of predicates eligible for merging or deletion. The corresponding raw results are [online available](#) [15]. We assume that an ideal similarity graph consists mostly of isolated nodes representing semantically distinct predicates, with only a small number of dense clusters indicating potential duplicates.

Out of 13,457 predicates, 3,837 clusters were detected, including 2,769 singleton clusters. A total of 2,801 predicates appear as nodes without neighbors in the similarity graph but were later assigned to clusters based on lexical label similarity. This indicates that lexical signals can partially compensate for embedding limitations, particularly when descriptions distort the embedding space. Table 3 shows the distribution of cluster-level recommendations derived from the advanced suggestions. Each cluster is treated as a single group and classified into *merge*, *delete*, *review*, or *keep* based on heuristic rules. Singleton clusters correspond to the *keep* category, when they contain actively used predicates. However, not all suggested actions are directly executable. The cleanup plan refines these results by applying additional constraints, such as excluding unused predicates

and requiring at least two used predicates per merge candidate group. This reduces the set of actionable clusters.

Table 3: Distribution of cluster-level recommendations and affected predicates.

Action	Clusters (Total)	Clusters (Relative)	Predicates (Total)	Predicates (Relative)
Merge	852	$\approx 22.2\%$	3,185	$\approx 23.7\%$
Delete	338	$\approx 8.8\%$	426	$\approx 3.2\%$
Review	162	$\approx 4.2\%$	7,361	$\approx 54.7\%$
Keep	2,485	$\approx 64.8\%$	2,485	$\approx 18.4\%$
Total	3,837	$\approx 100\%$	13,457	$\approx 100\%$

Overall, 1,989 predicates (14.78%) are unused and marked for deletion, also noting 54 completely unused clusters. Furthermore, 2,044 predicates (15.19%) across 786 clusters are identified as redundant merge candidates, meaning they can be replaced by a single representative predicate (already substracted) per cluster. In total, 4,033 (30%) predicates are considered potentially removable. However, these candidates require human validation, as computed similarity does not always imply true duplication. Additional cluster-based signals provide further insight: 593 clusters contain predicates created by the same user, and 79 clusters contain identical labels. These patterns indicate recurring modeling inconsistencies that are further discussed below. The cluster size distribution is highly skewed, with many singleton clusters and a small number of large clusters (up to size 477), as shown in Figure 5.

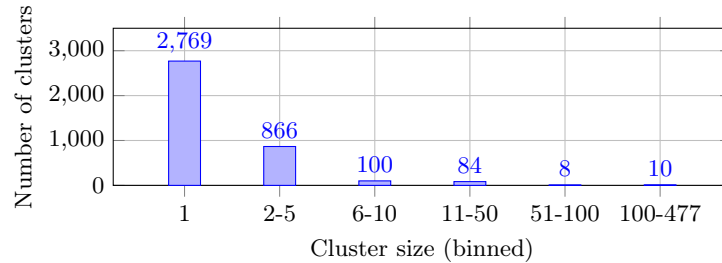


Figure 5: Cluster size distribution grouped into bins. The distribution is highly skewed, with many singleton clusters and few large clusters.

Modeling patterns. To complement the quantitative results, we manually inspected clusters for the live instance of ORKG using [SciKGDash](#) and identified recurring modeling patterns that lead to duplicate or inconsistent predicates.

Repeated creation of identical predicates by a single user. We observe cases (see Figure 6) where the same user creates multiple (near-)identical predicates, often within short time spans. These form isolated, tightly connected clusters and represent clear duplication. Our framework successfully identifies these components and enables consolidation into a single canonical predicate.

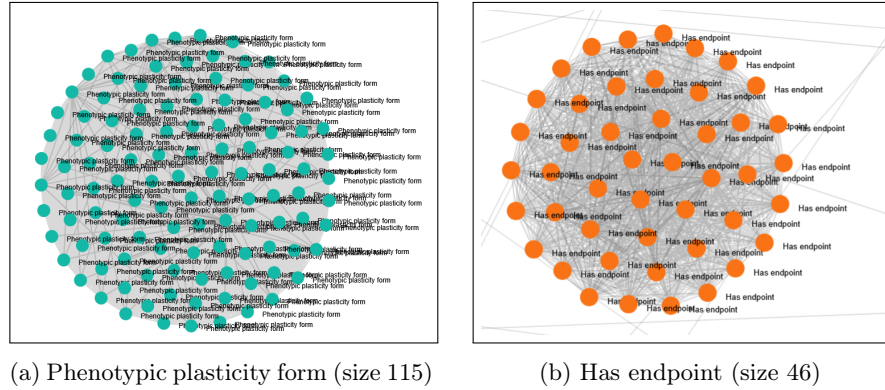


Figure 6: Clusters arising from repeated predicate creation by the same user.

Use of identifiers instead of labels. In some cases, users (accidentally) refer to predicates via IDs rather than their labels (see Figure 7). This introduces artificial lexical similarity, causing unrelated predicates to be grouped into a large cluster. These are false positives caused by label-based similarity, highlighting a core problem in the RKG interface, making prevention mechanisms urgent.

Lack of standardization for frequently used predicates. We also observe clusters containing semantically similar predicates (see Figure 8) created by different users for common modeling concepts (e.g., *method*, *data*, *material*). These are not always true duplicates. Resolving them requires human judgment and contextual inspection of predicate usage.

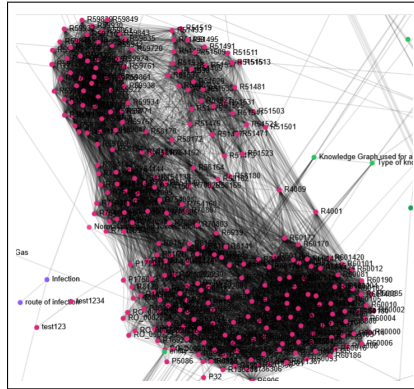


Figure 7: Cluster containing IDs instead of labels (size 312)

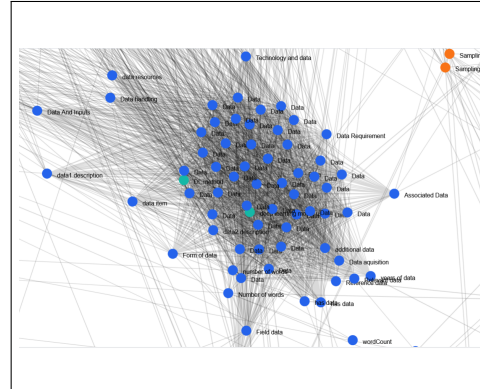


Figure 8: Example cluster describing data predicates (size 122)

Takeaway: The analysis reveals 30% predicate redundancy in the ORKG, indicating significant resolution potential. Cluster sizes are highly skewed, with many singletons and few large clusters. Recurring modeling patterns expose sources of redundancy and require different prevention mechanisms.

7 Discussion

This section will reflect on the implementation of the framework, present limitations and implications. The proposed framework is implemented in the ORKG context, with DD, DR, and prototypical DP functionalities integrated into [SciKGDash](#). Supported resolution actions include soft-merging, hard-merging, and deletion, while distinction is implicitly handled via ORKG redirects.

Reflection. The similarity graph view enables exploration of predicate clusters. Inspecting individual clusters in detail includes usage context and automatically generated suggestions for resolution actions. We identified three recurring modeling practices (cf. section 6) that contribute to duplicate predicates: Repeated creation of identical predicates by the same user, the use of identifiers instead of labels, and the lack of standardization for frequently used predicates. These observations provide concrete starting points for prevention mechanisms. The confusion between IDs and labels is partly rooted in the ORKG interface, where searching by ID requires a preceding #-symbol, a non-transparent feature that increases the likelihood of unintended duplicate creation. Likewise, the repeated creation of identical predicates by the same user points to missing feedback during predicate creation [9]. Currently, predicates can be created even if identical labels already exist, indicating the absence of basic guardrails such as duplicate warnings or recommendations. More restrictive approaches for controlled vocabularies, such as limiting predicate creation to curators or introducing a formal submission workflow (as in [Wikidata](#)), could mitigate these issues but must be balanced against ORKG’s openness as a crowdsourced platform [22,25]. Predicates can also be introduced through multiple entry points (e.g., frontend, REST API, Python package), complicating DP since consistent validation must be enforced across all pathways. In this context, mapping strategies become essential, particularly when importing data from external tools [19,25]. Approaches from ontology alignment can be leveraged to match incoming predicates to existing ones. Prior work on semi-automated predicate mapping for CSV import [18] can be extended using embedding-based similarity as part of a preventive pipeline (cf. section 5). The evaluation suggests that up to 30% of predicates may be redundant and potentially resolvable. The aim is to reach 0% redundancy. Although based on automated suggestions requiring human validation, they highlight substantial potential for improving RKG quality. To motivate continuous cleanup, [SciKGDash](#) computes a lightweight coherence score based on cluster sizes that increases as clusters are reduced through deletion and merge actions. The aim is to reach 100% coherence. At the same time, the cluster size distribution shows that the majority of clusters are singletons. These reflect either semantically distinct predicates or similarities below the threshold (0.65). This is further reinforced by the [prior observation in SciKGDash](#) that 76.62% of predicates lack descriptions, which reduces the effectiveness of embedding-based similarity and may lead to missing connections between semantically related predicates.

Limitations. Several limitations affect both the framework and the evaluation. *First*, the quality of DD depends on the chosen embedding and clustering strat-

egy, making both, the generated candidate groups and the reported redundancy estimates sensitive to missing or noisy textual metadata, similarity thresholds, and parameter choices. *Second*, clustering is performed as a preprocessing step, constraining subsequent DR actions and the evaluation to predicates within the same cluster. Consequently, semantically related predicates assigned to different clusters cannot be merged or counted as duplicate predicate candidates, potentially underestimating redundancy in ORKG. This reflects a design trade-off favoring high recall during clustering, followed by refinement within clusters. *Third*, the proposed resolution actions rely on heuristic rules and do not incorporate domain-specific semantics or contextual reasoning. Consequently, automated suggestions may include false positives, affecting the reliability of the reported redundancy estimates, and require human validation. *Finally*, the evaluation itself is limited to a snapshot of the ORKG and does not assess the long-term impact of applying DR strategies or the effectiveness of DP mechanisms in practice. In addition, the framework is shaped by ORKG-specific modeling patterns, limiting its generalizability to other RKGs.

Implications. Despite these limitations, the results have several important implications. *First*, predicate deduplication can significantly improve RKG quality. Consolidating predicates leads to more consistent relation usage, which facilitates entity alignment and entity DD, reduces semantic fragmentation, and improves downstream applications such as querying and analysis. *Second*, these findings highlight curation as a core component of RKG design. Because predicates encode semantics rather than real-world entities, their consolidation cannot be fully automated and requires human judgment. This positions curators as key users and underscores the need for dedicated interfaces, such as curation dashboards, to support inspection, interpretation, and decision-making. The proposed framework (cf. section 4) enables the integration of deduplication workflows into RKG infrastructures and elevates predicate redundancy to a core data quality challenge rather than a downstream issue. *Third*, duplicate creation emerges not only as a technical issue, but as a consequence of user interaction and interface design. The absence of validation during predicate creation represents a critical gap in ORKG’s crowdsourced modeling approach. Prevention mechanisms should combine backend validation with user-facing guidance, including improved autocomplete, similarity-based recommendations during creation, and clearer modeling guidelines. Integrating DD directly into the creation workflow would allow redundancy to be addressed at its source, reducing the need for costly post-hoc cleanup. In this sense, DD becomes an integral component of DP within the iterative framework (cf. Figure 1). *Finally*, the combination of DD, DR, and DP suggests a shift from reactive to proactive RKG quality management.

8 Conclusion

This section will summarize the paper and outlines future work opportunities.

Summary. We presented a framework for duplicate detection (DD), resolution (DR), and prevention (DP), demonstrated for the Open Research Knowl-

edge Graph (ORKG). By combining embedding-based similarity, clustering, and heuristic-based action suggestions, the framework identifies and consolidates duplicate predicates. Its integration into the deployed curation dashboard [SciKG-Dash](#) shows how these methods can be operationalized through interactive interfaces that support curators in inspecting and resolving data quality challenges.

Our evaluation shows that up to 30% of predicates may be redundant, highlighting substantial potential for improving Research Knowledge Graph (RKG) quality and consistency. At the same time, the predominance of singleton clusters suggests that many predicates are already well-formed or fall below current similarity thresholds. Qualitative analysis revealed recurring modeling patterns, including repeated creation of identical predicates by the same user, confusing identifiers with labels, and lack of standardization for frequently used predicates. These findings show that duplicate predicates arise from both user behavior and system design. Addressing them requires a holistic approach combining automated detection, human-centered curation, and preventive mechanisms. The proposed framework links these aspects into an iterative data quality management process for RKGs.

Future Work. Several directions can extend this work. *First*, improving scalability of DD is essential [13], e.g., by replacing pairwise similarity with approximate nearest neighbor search [8]. Future work should also benchmark runtime, analyze pipeline steps efficiency, and evaluate the Leiden algorithm [32] as an alternative to Louvain for improved efficiency and clustering quality. *Second*, incorporating richer semantic signals, such as domain and range analysis of predicates, may improve both DD and DP. While ORKG does not enforce such constraints, usage patterns may implicitly reveal them. User studies could evaluate whether these signals improve modeling decisions. Analyzing co-occurring predicates could suggest existing predicates for a given domain or modeling scope [25]. *Third*, prevention mechanisms should be more tightly integrated into the predicate creation process. Completing the similarity-based autocomplete in the frontend can guide users toward existing predicates and reduce duplicate creation. *Fourth*, large language models (LLMs) offer opportunities to support curation tasks. They can support the distinguish action by suggesting labels, generating descriptions from usage context, explaining predicate differences for human decision-making, and automating trivial duplicate cases based on curation history. *Fifth*, supporting cross-cluster actions and more flexible merging strategies would improve the DR process. Current cluster-level constraints may prevent consolidating semantically related predicates, that were not grouped initially. *Finally*, integrating the framework into real-world curation workflows remains a key step. Collaborating with the ORKG Curation & Community Building team will enable a human gold standard for practical validation, including clustering accuracy, precision/recall and parameter sensitivity analyses, and clustering parameters tuning.

Overall, this work lays the foundation for treating duplicate predicate management as a core aspect of RKG curation, moving from reactive cleanup toward proactive and continuous RKG quality management.

Acknowledgments. The authors would like to thank the Federal Government and the Heads of Government of the Länder, as well as the Joint Science Conference (GWK), for their funding and support. This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) SFB/TRR-298-SIIRI (No. 426335750), NFDI4Ing (No. 442146713), and NFDI4DataScience (No. 460234259).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Abedjan, Z., Naumann, F.: Synonym analysis for predicate expansion. In: The Semantic Web: Semantics and Big Data (2013). https://doi.org/10.1007/978-3-642-38288-8_10
2. Arab Oghli, O., D’Souza, J., Auer, S.: Clustering Semantic Predicates in the Open Research Knowledge Graph. In: From Born-Physical to Born-Virtual: Augmenting Intelligence in Digital Libraries (2022). https://doi.org/10.1007/978-3-031-21756-2_39
3. Auer, S., et al.: Open Research Knowledge Graph: A Large-Scale Neuro-Symbolic Knowledge Organization System. In: Handbook on Neurosymbolic AI and Knowledge Graphs. IOS Press (2025), <https://doi.org/10.3233/FAIA250216>
4. Baas, J., Dastani, M.M., Feelders, A.J.: Exploiting Transitivity for Entity Matching. The Semantic Web: ESWC (2021). https://doi.org/10.1007/978-3-030-80418-3_20
5. Babaei Giglou, H., D’Souza, J., Karras, O., Auer, S.: OntoAligner: A Comprehensive Modular and Robust Python Toolkit for Ontology Alignment. In: The Semantic Web (2025). https://doi.org/10.1007/978-3-031-94578-6_10
6. Batini, C., Scannapieco, M.: Data and Information Quality. Springer (2016)
7. Blondel, V., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment (2008). <https://doi.org/10.1088/1742-5468/2008/10/P10008>
8. Chen, Y., Li, D., Yan, L., Ma, Z.: Two-Stage Detection of Semantic Redundancies in RDF Data. Journal of Web Engineering (2023). <https://doi.org/10.13052/jwe1540-9589.2184>
9. D’Souza, J., Hussein, H., Evans, J., Vogt, L., Karras, O., Ilangoan, V., Auer, S.: Quality Assessment of Research Comparisons in the Open Research Knowledge Graph: A Case Study. In: JLIIS.It (2024). <https://doi.org/10.36253/jlis.it-547>
10. Fürber, C., Hepp, M.: SWIQA – A Semantic Web Information Quality Assessment Framework. ECIS (2011), <https://aisel.aisnet.org/ecis2011/76>
11. Hogan, A., Gutierrez, C., Cochez, M., de Melo, G., et al.: Knowledge Graphs. Springer (2022). <https://doi.org/10.1007/978-3-031-01918-0>
12. Huaman, E., Fensel, D.: Knowledge Graph Curation: A Practical Framework. In: Proceedings of the 10th International Joint Conference on Knowledge Graphs (2022). <https://doi.org/10.1145/3502223.3502247>
13. Huaman, E., Kärle, E., Fensel, D.A.: Duplication Detection in Knowledge Graphs: Literature and Tools. ArXiv (2020), <https://api.semanticscholar.org/CorpusID:215814131>

14. Jeon, M., Suh, J., Cho, S.: Schema Matching on Graph: Iterative Graph Exploration for Efficient and Explainable Data Integration. ArXiv (2025). https://doi.org/https://arxiv.org/abs/2511.20285v2?utm_source=copilot.com
15. John, L., Aggarwal, S.: Supplementary material: SciKGDash duplicate predicate analysis in ORKG (2026). <https://doi.org/10.5281/zenodo.19948623>
16. John, L.: Duplicate Predicate Detection, Resolution, and Prevention (2026). <https://doi.org/10.48366/R1637915>
17. John, L., Auer, S., Karras, O.: SciKGDash: The Scientific Knowledge Graph Dashboard for Supporting Knowledge Curation. In: ACM/IEEE Joint Conference on Digital Libraries. IEEE (2025). <https://doi.org/JCDL67857.2025.00030>
18. John, L., Farfar, K.E., Auer, S., Karras, O.: SciMantify - A Hybrid Approach for the Evolving Semantification of Scientific Knowledge. In: 25th International Conference on Web Engineering (ICWE) 2025 (2025). <https://doi.org/10.15488/19006>
19. John, L., Ghanmi, A.M., Wittenborg, T., Auer, S., Karras, O.: ExtracTable: Human-in-the-Loop Transformation of Scientific Corpora into Structured Knowledge. In: The 29th International Conference on Theory and Practice of Digital Libraries (2026). https://doi.org/10.1007/978-3-032-05409-8_27
20. Kalo, J.C., Mennicke, S., Ehler, P., Balke, W.T.: Detecting Synonymous Properties by Shared Data-Driven Definitions. The Semantic Web: 17th International Conference (2020). https://doi.org/10.1007/978-3-030-49461-2_21
21. Kalo, J., Ehler, P., Balke, W.: Knowledge Graph Consolidation by Unifying Synonymous Relationships. The Semantic Web – ISWC (2019). https://doi.org/10.1007/978-3-030-30793-6_16
22. Karras, O., Groen, E.C., Khan, J.A., Auer, S.: Researcher or Crowd Member? Why not both! The Open Research Knowledge Graph for Applying and Communicating CrowdRE Research. In: 2021 IEEE 29th International Requirements Engineering Conference Workshops (REW) (2021). <https://doi.org/10.1109/REW53955.2021.00056>
23. Li, H., Appleby, G., Brumar, C.D., Chang, R., Suh, A.: Knowledge Graphs in Practice: Characterizing their Users, Challenges, and Visualization Opportunities. IEEE Transactions on Visualization and Computer Graphics (2024). <https://doi.org/10.1109/TVCG.2023.3326904>
24. Manghi, P., Atzori, C., Bonis, M.D., Bardi, A.: Entity deduplication in big data graphs for scholarly communication. Data Technologies and Applications (2020). <https://doi.org/10.1108/DTA-09-2019-0163>
25. Nechakhin, V., D'Souza, J., Eger, S.: Evaluating Large Language Models for Structured Science Summarization in the Open Research Knowledge Graph. Information (2024). <https://doi.org/10.3390/info15060328>
26. Niazmand, E., Vidal, M.: Completing Predicates Based on Alignment Rules from Knowledge Graphs. In: Database and Expert Systems Applications (2024). https://doi.org/10.1007/978-3-031-68309-1_5
27. Rowley, J.: The wisdom hierarchy: representations of the DIKW hierarchy. Journal of Information Science (2007). <https://doi.org/10.1177/0165551506070706>
28. Salem, S., Benchikha, F.: Lodquma: A free-ontology process for linked (open) data quality management. Journal of King Saud University - Computer and Information Sciences (2022). <https://doi.org/10.1016/j.jksuci.2021.06.001>
29. Setyawan, D.C., Kusumasari, T.F., Alam, E.N.: Data Cleansing Processing Using Pentaho Data Integration. In: International Conference on Science and Technology (2020). <https://doi.org/10.1109/ICST50505.2020.9732824>

30. Sousa, G., Lima, R., Trojahn, C.: Combining Word and Sentence Embeddings with Alignment Extension for Property Matching. In: Proceedings of the 18th International Workshop on Ontology Matching (2023)
31. Stocker, M., et al.: SKG4EOSC - Scholarly Knowledge Graphs for EOSC: Establishing a Backbone of Knowledge Graphs for FAIR Scholarly Information in EOSC. Research Ideas and Outcomes (2022). <https://doi.org/10.3897/rio.8.e83789>
32. Traag VA, Waltman L, v.E.N.: From Louvain to Leiden: guaranteeing well-connected communities. Sci Rep (2019). <https://doi.org/10.1038/s41598-019-41695-z>
33. Xue, B., Zou, L.: Knowledge Graph Quality Management: A Comprehensive Survey. IEEE Transactions on Knowledge and Data Engineering (2023). <https://doi.org/10.1109/TKDE.2022.3150080>
34. Zhao, X., Zeng, W., Tang, J., Wang, W., Suchanek, F.M.: An experimental study of state-of-the-art entity alignment approaches. IEEE Transactions on Knowledge and Data Engineering (2022). <https://doi.org/10.1109/TKDE.2020.3018741>