INTEGRATE-KG: A Workflow For Unifying Heterogeneous Data Driven by Shared Languages

Nahed Abu Zaid*, Kara Schatz†, Kimberly Bourne‡, Darrell Harry*, Christine Hendren‡, Anna-Maria Marshall§, Khara Grieger*, Jacob Jones*, Alexey V. Gulyuk*, Yaroslava G. Yingling*, and Rada Chirkova*
*North Carolina State University, Raleigh, North Carolina, USA
Emails: {naabuzai,diharry,kdgriege,jljone21,agulyuk,ygyingli,rychirko}@ncsu.edu
†Xavier University, Cincinnati, Ohio, USA
Email: schatzk@xavier.edu
‡Appalachian State University, Boone, North Carolina, USA
Emails: {bournekd,hendrenco}@appstate.edu
§University of Illinois Urbana-Champaign, Urbana, Illinois, USA
Email: amarshll@illinois.edu

2024 IEEE International Conference on Big Data (BigData) | USGov | DOI: 10.1109/BigData62323.2024.10825736

***Abstract*—In large-scale multidisciplinary consortia endeavors that address problems of research, industry, and public-good significance, it is typically a priority to integrate the heterogeneous data contributed by the consortia participants into a unified data representation. Knowledge graphs (KGs) are a typical choice for the data model of the resulting data repositories. To overcome potential issues with terminology misalignment, consortia commonly dedicate resources to the development of shared languages (vocabularies), with the intent of enabling diverse participants to understand and build on each other's work. Our research focus in this paper is on the challenge of automating integration into unified KGs of diverse data that potentially use different terminology, with the help of the available shared languages to resolve terminology clashes.**

To address the challenge, we introduce a data-integration workflow called INTEGRATE-KG that is domain agnostic, yet domain aware through opportunities for the involvement of humans-in-the-loop. A key feature of the approach is in its use of the synonyms available for the shared languages to automate semantics-level terminology alignment across the individual data contributions *after* they have been submitted for integration. INTEGRATE-KG also includes a module for automatically enriching the available shared languages, with opportunities for domain experts to provide semantic corrections and feedback. We present the workflow, report on our experiences with applying it to experimental, survey, and shared-language data on phosphorus sustainability, and provide suggestions for involving domain experts in INTEGRATE-KG as humans-in-the-loop.

***Index Terms*—Knowledge graphs for big scientific and experimental data; knowledge-graph construction; knowledge-graph applications.**

I. Introduction

Major projects addressing problems of research, industry, and public-good significance are commonly approached by consortia of stakeholders that have diverse backgrounds. Having a multidisciplinary team of research scientists study a topic of shared interest, such as phosphorus sustainability, from different perspectives and at a variety of scales would be a typical example. On such consortia projects, it is common for different consortium subgroups to provide their data contributions to the shared information pool in a variety of formats, such as spreadsheets, text, and images. Further, the diversity of the participants' backgrounds means that different subgroups within the consortium are not necessarily using the same terminology for the items being worked on independently in the subgroups. As an example, the terms "Aluminum Oxide" and "alumina" are used in different subareas of materials science to refer to the same chemical Al_2O_3.

In large-scale multidisciplinary consortia endeavors, it is typically a priority to integrate the heterogeneous data contributed by the subgroups into a unified data representation. As terminology issues might serve as a barrier to the free sharing of the contributed information in pursuit of new insights and discoveries, consortia commonly dedicate resources to the development of shared languages (vocabularies), with the intent of enabling diverse participants to understand and build on each other's work. Our research focus in this paper is on the problem of automating integration into unified data repositories of diverse data contributions that potentially use different terminology, with the help of the commonly developed shared languages to resolve terminology clashes.

Consider a motivating example that stems from the use case that we showcase in this paper. Fig. 1 shows some of the large-scale data coming from teams that work in the Science and Technologies for Phosphorus Sustainability (STEPS) Center,[1] a multidisciplinary consortium with a unique large-scale data portfolio and unique challenges associated with it. In STEPS, researchers and research teams, stakeholders, and clients coming from different backgrounds and multiple disciplines generate and analyze data in a number of formats, at a variety of scales, and typically with different structures. The data related to phosphorus sustainability obtained at the STEPS Center range from numerical databases to qualitative data sets and include text, graphical information, and spatial data. Integrating such diverse data for further analysis and collaboration is challenging, due to inconsistencies in data standards, the ways the data are stored and shared, and how they can be interpreted by researchers coming from different backgrounds due to the terminology differences. Fig. 1 also shows the information-integration objective for the consortium, with the integrated data being unified both format-wise and terminology-wise, thus enabling accelerated scientific discovery through shared knowledge.

Many multidisciplinary consortia, including the STEPS

This research has been supported by the Science and Technologies for Phosphorus Sustainability (STEPS) Center under National Science Foundation Grant No. CBET-2019435.

[1]https://steps-center.org

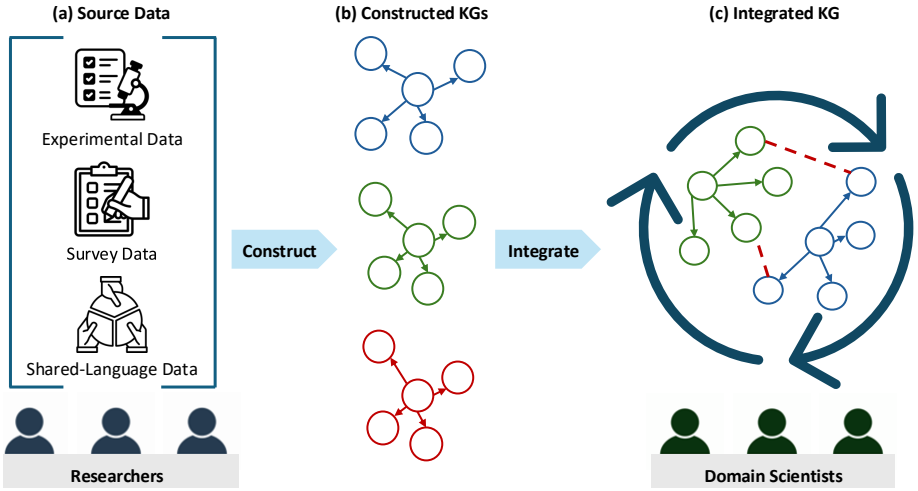

Fig. 1. Large-scale data coming from teams that work in the Science and Technologies for Phosphorus Sustainability (STEPS) Center, as well as the information-integration objective for the Center, with the integrated data being unified format-wise and terminology-wise, thus enabling accelerated scientific discovery through shared knowledge. Fig. 1(a) depicts the types of source data: experimental data, survey data, and shared-language data. Fig. 1(b) shows the individual knowledge graphs (KGs) constructed from these source data, and Fig. 1(c) depicts the integrated KG, open to refinements by humans-in-the-loop.

Center, choose knowledge graphs (KGs) to integrate their data. This choice is justified by the flexible triple-based "subject-predicate-object" KG representation of the data that enables explicit modeling of relationships between entities. The KG data model also makes it easy to incorporate metadata and reasoning into the data [1]. The domain-agnostic approach that we introduce in this paper, called INTEGRATE-KG, is designed to automate integration of diverse data using the flexible metadata-enabling KG data model as the unified data representation in the envisioned integrated repository.

The main focus of the proposed INTEGRATE-KG workflow is that of resolving potential terminology clashes between the data contributions made by participants with differing backgrounds. To this end, INTEGRATE-KG takes advantage of any available shared language developed in the project. A straightforward approach to the needed terminology alignment would call on the data contributors to translate their data using the agreed-on shared terminology prior to submitting their data for integration. We recognize that requiring such an effort could be a burden on the project participants. As a result, a key feature of INTEGRATE-KG is in its use of the synonyms available for the shared languages to automate terminology alignment across the individual data contributions after they have been submitted for integration. In addition, INTEGRATE-KG includes a module for automatically enriching the available shared languages, with opportunities for domain experts to provide corrections and feedback.

To the best of our knowledge, the project that is the closest to ours in its aims and methods is the BUILD-KG effort by Schatz and colleagues [2]. The BUILD-KG approach is designed to automate integration into the KG format of data in a variety of representations. After the data have all been converted into the KG format, the last step in BUILD-KG is to connect the resulting individual KG fragments using the same terms that occur across these KGs. (E.g., the vertices for "alumina" that occur in several individual KGs would all get merged into a single vertex.) From our experience with the STEPS Center use case, this approach to integrating KGs arising from different data sources can miss significant opportunities for enabling data interoperability and richer analytics, with much of the data connections potentially remaining untapped. The key feature of the proposed INTEGRATE-KG approach that we discussed earlier addresses this challenge by providing enriched *semantics-oriented* ways to connect individual KGs arising from different sources into a holistic KG. (E.g., INTEGRATE-KG would not only connect all the vertices for "alumina," but will also connect them with all the vertices for "Aluminum Oxide," which is a synonym for "alumina.") As a result, the KGs returned by INTEGRATE-KG can open new avenues for efficient data navigation, pattern discovery, and analytics across the entire multidisciplinary organizations that have contributed their data to the KGs.

Our specific contributions in this paper are as follows:

- We propose a domain-agnostic, humans-in-the-loop workflow called INTEGRATE-KG to construct KGs from structured and unstructured data with the use of the shared languages adopted within the given use cases, which makes INTEGRATE-KG domain aware;
- We introduce domain-agnostic procedures for automating semantic terminology alignment across the individual data contributions in a given use case after they have been submitted for integration, with the help of the synonyms available for the terms in the shared languages;
- We propose an approach for automatically enriching the shared languages developed for particular use cases, with opportunities for humans-in-the-loop to provide domain-aware semantics-level corrections and feedback;
- We outline our implementation of INTEGRATE-KG, and report on our experiences with applying it to experimental, survey, and shared-language STEPS-Center data; and
- We report on our experiences working with STEPS researchers, and provide tips on involving scientists as humans-in-the-loop in the INTEGRATE-KG workflow.

The remainder of this paper is organized as follows. We review related work in Section II and provide a problem statement in Section III. In Section IV, we introduce our proposed domain-agnostic yet domain-aware framework, illustrating it in Section V with a real-world STEPS-Center use case. In Section VI, we describe the role of humans-in-the-loop in the INTEGRATE-KG workflow. We conclude in Section VII.

II. Related Work

Our work is related to efforts in the fields of knowledge graph (KG) construction and integration, see, e.g., [2]–[12].

Most KG construction efforts focus on extracting information from text and converting it to the KG format, e.g., [3], [4]. The approaches of [2], [5] focus on constructing KGs from data formats beyond text, e.g., spreadsheet data, images, and code. Our proposed INTEGRATE-KG workflow also focuses on constructing KGs from data format beyond text, but expands upon [2], [5] to include support for two new data formats: survey data and shared-language data. The work of [13] also handles multiple data types, specifically various file types, but it handles them individually rather than as collections of related data that may overlap. This is in contrast to INTEGRATE-KG, which specifically addresses the case where data may overlap. [13] also does not involve domain experts in the construction process, as INTEGRATE-KG does, which may limit its applicability to data where domain expertise is required to understand the semantics.

Many KG construction efforts are domain-specific, i.e., tailored to the needs of a particular field [14]. For instance, [7] focuses on cybersecurity education, while [8], [9] apply to biology and biomedicine, [10] specializes in the Food, Energy, Water (FEW) domain, and [11] handles judicial cases. While effective, these methods are often limited to their respective domains. In contrast, the proposed INTEGRATE-KG workflow is entirely domain-agnostic. While we demonstrate INTEGRATE-KG through a use case in materials science, it can be easily applied to other domains.

Existing KG-integration approaches, e.g., [4], [5], [12], perform entity alignment via textual similarity or embeddings, and can struggle to effectively handle complex data. In contrast, INTEGRATE-KG relies on shared-language data from domain scientists to introduce relationships between synonymous vertices of interest. INTEGRATE-KG also accomodates incremental updates, as recommended by [15].

The work of [2] also considers the integration of KGs from multiple sources. However, it requires the input KGs to use unified terminology and only handles the case where vertices have the same name. In contrast, our proposed INTEGRATE-KG framework extends that of [2] by integrating KGs using shared-language data from various sources, not requiring the terminology to be unified ahead of time.

III. Background and Problem Statement

Definition 1 (Knowledge Graph): A *knowledge graph (KG)* is a 5-tuple $G = (V, T, \tau, P, E)$, where V is the set of *vertices (entities)*, T is the set of *vertex types*, $\tau : V \to T$ is the *vertex-type labeling function* that assigns to each vertex a vertex type in T, P is the set of *predicates*, and $E \subseteq V \times P \times V$ is the set of *triples (edges)*. Each element $(s, p, o) \in E$ is called a *triple (edge)* and encodes the real-world fact in the form of the relationship $p \in P$ between the *subject* vertex $s \in V$ and the *object* vertex $o \in V$. For example, the triple $(\texttt{phosphate}, \texttt{is\_part\_of}, \texttt{fertilizer})$ encodes the fact that "phosphate is part of fertilizer."

Definition 2 (KG Construction): KG construction involves transforming a data set D into a KG G. The process is formalized as $\mathcal{C} : D \times \mathcal{K}(D) \to G$, where $\mathcal{K}(D)$ denotes background information $\mathcal{K}$ applied to D to facilitate construction. $\mathcal{K}$ can include, e.g., domain-specific rules and synonym matching. The function $\mathcal{C}$ creates the vertices V, vertex types T, vertex-type labeling function τ, predicates P, and triples E, ensuring that G aligns with the semantics encoded in $\mathcal{K}$.

Definition 3 (KG Integration): KG integration is the process of merging a set of multiple KGs $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$ into a single, unified KG K. This process is formalized as $\mathcal{I} : \mathcal{G} \times G_L \to K$, where G_L is a shared-language KG that facilitates the integration process. The function $\mathcal{I}$ connects vertices across the various input KGs based on the triples in G_L, resulting in an integrated KG $K = (V, T, \tau, P, E)$ that synthesizes the information from the individual KGs.

We now introduce the problem statement for our work: Given a set $\mathcal{D}$ of heterogeneous data sources and a shared-language repository L that connects related terms across the sources in $\mathcal{D}$, integrate the data in $\mathcal{D}$ into a unified KG $K = (V, T, \tau, P, E)$ that could enable richer analytics not supported by the sources taken in isolation. The *shared-language repository* L includes terms with their meanings and synonyms. The repository is formatted as a spreadsheet, where rows represent individual terms and columns include the term name, meaning and synonyms. Terms and synonyms can be encoded as KG vertices with edges connecting them.

To limit the scope of this problem, we focus on two source types for $\mathcal{D}$: (1) *experimental data* and (2) *survey data*. These types have been selected due to their prominent usage by domain scientists. *Experimental data* include observations and measurements that can be encoded as KG vertices. They are stored in tables or spreadsheets, where rows represent individual experiments, and columns specify attributes (observations and measurements) that are recorded. Relationships between attributes are provided in a separate sheet and can be encoded as KG edges. *Survey data* include questions and responses that can be encoded as KG vertices, along with response counts and percentages. They are stored in tables or spreadsheets, where each sheet details a question and its responses. Relationships between questions and responses can be encoded as KG edges. More detailed descriptions of these data sources can be found in Section IV.

IV. The INTEGRATE-KG Workflow

We now introduce our proposed domain-agnostic workflow called INTEGRATE-KG for integrating heterogeneous data sources into a unified knowledge graph (KG) that could enable richer analytics not supported by the sources taken in isolation. Sections IV-A, IV-B, and IV-C, describe the processes for constructing KGs from experimental, survey, and shared-language data, respectively. Then, Section IV-D, details the procedure for integrating individual KGs into a unified, cohesive KG by using the shared-language KG.

A. Constructing a KG from Experimental Data

The proposed domain-agnostic process for KG construction from experimental data requires the data to be in the format described in Section IV-A1. The output KG and its ontology are discussed in Section IV-A2. The necessary experimental-data preprocessing is detailed in Section IV-A3, and the construction process is presented in Section IV-A4.

1) The Input Data: The process accepts experimental data in a specific format, which was first introduced in [2] and can be achieved through collaboration with domain scientists, see Section VI-A for the details. The format introduced in [2]

consists of: (1) a relation sheet R, and (2) a triple sheet S. The relation sheet R is analogous to a relational table, i.e., each row corresponds to a single data object and each column corresponds to an attribute. Each cell entry in R is a desired vertex in the resulting KG whose vertex type will be the column header. The triple sheet S consists of three columns: ***Subject***, ***Predicate***, and ***Object***. Each row $(st, p, ot) \in S$ is a desired triple type in the resulting KG. Here, st and ot are column headers from R and indicate the subject and object vertex types that have the relationship p. Thus, the role of the triple sheet S is to enumerate the relationships (edges) between the entities (vertices) in the relation sheet R. See Section V-C1 for example input data.

2) The Output KG: The KG $G = (V, T, \tau, P, E)$ generated by the construction process of Section IV-A4 contains the data from the relation sheet R organized according to the triple sheet S. The vertex set V consists of all the unique entries in R; entries that appear in multiple rows correspond to a single vertex. The set of vertex types T consists of the column headers of R, and the vertex-type labeling function τ maps each entry in R to its corresponding column header. The predicate set P consists of all the entries from the ***Predicate*** column of S, and the set of triples E consists of a single triple of each triple type (st, p, ot) in S for each data object d in R. Here, the subject and object of the triple are the entries from d in columns st and ot.

3) The Data Preprocessing: After formatting the input data according to the specifications in Section IV-A1, the data are preprocessed to ensure successful KG construction, First, the relation sheet R and the triple sheet S are checked for compatibility, i.e., to ensure that the triple types found in S use vertex types found in the column headers of R. Next, missing values are addressed through data imputation, ensuring data completeness without compromising the integrity of the information. These values are also flagged for expert revision.

4) The Construction Process: The process for constructing a KG from experimental data was first presented in [2]. The algorithm takes as inputs the relation sheet R and the triple sheet S, formatted as specified in Section IV-A1, and outputs the KG G described in Section IV-A2.

The algorithm first makes an empty KG G. Then, vertices are added to G: one vertex v for each unique entry e in the relation sheet R. The vertex v is given vertex type t, where t is the column header corresponding to e. The creation of v can be accomplished via the following query:[2]

$$\texttt{MERGE}\ (v : t\ \{\texttt{name} : e\}) \tag{1}$$

This query ensures that the vertex v is created only if it does not already exist, thereby avoiding duplicates.

We expand upon the approach of [2] for the case of relation-sheet entries that were flagged during the data preprocessing of Section IV-A3. For such an entry e, its corresponding vertex v is flagged by modifying (1) to include the following clause:

$$\texttt{SET}\ v.\texttt{flag} = True \tag{2}$$

Next, edges are added to G: one edge for each triple type in the triple sheet S for each data object in the relation sheet R. That is, the algorithm iterates over each row, i.e., triple type (st, p, ot), in S, and over each row, i.e., data object d, in R, creating the triple (s, p, o) by extracting s and o from the entries of the st and ot columns of d. The creation of (s, p, o) can be accomplished via the following query:

$$\begin{aligned}&\texttt{MATCH}\ (v_s : st\ \{\texttt{name} : s\}), (v_o : ot\ \{\texttt{name} : o\})\\ &\texttt{MERGE}\ (v_s) - [: p] \rightarrow (v_o)\end{aligned} \tag{3}$$

This query identifies the vertices corresponding to s and o, and ensures that an edge of type p between is created between them if it does not already exist, thereby avoiding duplicates.

After creating these vertices and edges, G contains the data from relation sheet R organized according to triple sheet S, and is returned as the output of the construction process.

B. Constructing a KG from Survey Data

The proposed domain-agnostic process for KG construction from survey data requires the data to be in the format described in Section IV-B1. The output KG and its ontology are discussed in Section IV-B2. The necessary survey-data preprocessing is detailed in Section IV-B3, and the construction process is presented in Section IV-B4.

1) The Input Data: The process accepts survey data as a set of spreadsheets $\mathcal{S}$, where each sheet details a single survey question and its responses. The first row of each sheet contains the question posed to participants. Subsequent rows list the responses, with columns ***Response Text***, ***Count***, and ***Percent*** indicating the response text, the numerical count of participants who selected the response, and the percentage that this count represents out of the total number of responses. See Section V-D1 for example input data.

2) The Output KG: The KG $G = (V, T, \tau, P, E)$ generated by the construction process of Section IV-B4 contains the data from the set of survey sheets $\mathcal{S}$, along with keywords extracted during the data preprocessing of Section IV-B3. The vertex set V consists of all the unique questions and responses from $\mathcal{S}$, and all the unique keywords. The set of vertex types $T = \{\texttt{Question}, \texttt{Response}, \texttt{Keyword}\}$, and the vertex-type labeling function τ maps each question to the type `Question`, each response to the type `Response`, and each keyword to the type `Keyword`. The predicate set $P = \{\texttt{HAS\_RESPONSE}, \texttt{HAS\_KEYWORD}\}$, and the set of triples E consists of a single triple with predicate `HAS_RESPONSE` between each question and each of its corresponding responses, and a single triple with predicate `HAS_KEYWORD` between each question and each of its keywords.

3) The Data Preprocessing: After formatting the input data according to the specifications in Section IV-B1, the data are preprocessed to extract keywords from the survey questions. The purpose of these keywords is primarily to enhance the resulting KG and its ability to integrate with other KGs.

The data-preprocessing steps are outlined in Algorithm 1, which takes as input the set $\mathcal{S}$ of raw survey sheets, formatted as specified in Section IV-B1, and produces as output a set $\mathcal{S}'$ of preprocessed survey sheets.

First, the set $\mathcal{S}'$ is initialized as the empty set (line 1). Then, the algorithm iterates over each sheet s in $\mathcal{S}$ (lines 2–7). The first row of the sheet, $s[0]$, contains the question text q, which is extracted (line 3) via the following regular expression:[3]

```
\b(Do|Have|How|What|Which|Why)\b.*?\?
```

[2]All queries are presented in the Cypher [16] query language.

[3]In practice, this expression can be expanded to support a specific use case by including additional question-starting words separated by the delimiter `|`.

Algorithm 1: Preprocessing Survey Data.

Input: Set of raw survey sheets $\mathcal{S}$.

Output: Set of preprocessed survey sheets $\mathcal{S}'$.

1: $\mathcal{S}' \leftarrow \emptyset$;
2: **for** sheet $s \in \mathcal{S}$ **do**
3: $q \leftarrow$ parse($s[0]$); // `parse out the question text`
4: $s[0] \leftarrow q$;
5: *keywords* $\leftarrow$ extract_keywords(q);
6: $s \leftarrow s$.append(*keywords*);
7: $\mathcal{S}' \leftarrow \mathcal{S}' \cup \{s\}$;
8: **return** $\mathcal{S}'$;

This pattern searches for common question-starting words, e.g. `Do`, `How`, `Have`, and `What`, and extracts the entire sentence up to the first question mark. It ensures that the complete question text is retrieved, even if it contains complex or variable structures. The extracted question text q is placed back in $s[0]$ for use during construction (line 4).

Next, keywords are extracted from the question text q via natural language processing (NLP) techniques (line 5). The keyword extraction involves the following steps:

1) The text q is tokenized, i.e., split into individual words.
2) Stop words, e.g., "the," "is," and "in," are removed because they are insignificant to the meaning of q.
3) The remaining words are tagged with their part of speech (POS), which identifies their grammatical roles, e.g., noun (`NN` tag), verb (`VB` tag), or adjective (`JJ` tag).
4) Single and compound nouns are extracted as keywords by identifying POS-tag patterns `JJ` + `NN` and `NN` + `NN`.

The extracted keywords are appended to the end of the sheet s (line 6), which is then added to the set $\mathcal{S}'$ (line 7).

After iterating through all sheets, the complete set $\mathcal{S}'$ is returned as the output of Algorithm 1 (line 8).

4) The Construction Process: The process for constructing a KG from survey data is outlined in Algorithm 2. The algorithm takes as input the set of preprocessed survey sheets $\mathcal{S}'$ (output by Algorithm 1) and outputs the KG G described in Section IV-B2.

First, an empty KG G is made (line 1). Then, the algorithm iterates over each sheet s in $\mathcal{S}'$ (lines 2–16). The first row of the sheet contains the question text q (line 3). A corresponding vertex v of type `Question` is created (line 4) with q stored in the property `text` (line 5). The creation of v can be accomplished via the following query:

$$\texttt{MERGE}\ (v : \texttt{Question}\ \{\texttt{text} : q\}) \tag{4}$$

This query creates a new vertex v of type `Question` with property $v.\texttt{text} = q$, if such a vertex does not already exist.

Next, the algorithm iterates over the remaining rows of the sheet s (lines 6–11), except for the last line, which contains the keywords. Each row corresponds to a response r. A vertex w of type `Response` is created (line 7) with the property `text` that is extracted from the ***Response Text*** (line 8) column of the row. The creation of w can be accomplished via a query similar to (4). Also, an edge e between the question vertex v and its response vertex w is created (line 9) with predicate `HAS_RESPONSE` and properties `count` and `percent` that are extracted from the ***Count*** (line 10) and ***Percent*** (line 11)

Algorithm 2: Constructing a KG from Survey Data.

Input: Set of preprocessed survey sheets $\mathcal{S}'$.

Output: KG G containing the data from $\mathcal{S}'$.

1: $G \leftarrow \emptyset$; // `initialize an empty KG` G
2: **for** sheet $s \in \mathcal{S}'$ **do**
3: $q \leftarrow s[0]$; // `extract the question text`
4: $v \leftarrow G$.create_vertex(`Question`);
5: $v.\texttt{text} \leftarrow q$;
6: **for** row $r \in s[1:-1]$ **do**
7: $w \leftarrow G$.create_vertex(`Response`);
8: $w.\texttt{text} \leftarrow r$[***Response Text***];
9: $e \leftarrow G$.create_edge(v, `HAS_RESPONSE`, w);
10: $e.\texttt{count} \leftarrow r$[***Count***];
11: $e.\texttt{percent} \leftarrow r$[***Percent***];
12: *keywords* $\leftarrow s[-1]$; // `extract the keywords`
13: **for** keyword $k \in$ *keywords* **do**
14: $x \leftarrow G$.create_vertex(`Keyword`);
15: $x.\texttt{name} \leftarrow k$;
16: G.create_edge(v, `HAS_KEYWORD`, x);
17: **return** G;

columns of the row. This edge creation can be accomplished via the following query:

$$\begin{aligned} &\texttt{MATCH}\ (v : \texttt{Question}\ \{\texttt{text} : q\}),\\ &\qquad\quad (w : \texttt{Response}\ \{\texttt{text} : r\})\\ &\texttt{MERGE}\ (v) - [e : \texttt{HAS\_RESPONSE}] \rightarrow (w)\\ &\texttt{SET}\ e.\texttt{count} = c,\ e.\texttt{percent} = p \end{aligned} \tag{5}$$

This query identifies the vertices v, w and ensures that an edge of type `HAS_RESPONSE` with properties $e.\texttt{count} = c$ and $e.\texttt{percent} = p$ is created between them.

Next, the algorithm extracts the keywords from the final row of the sheet s (line 12), and iterates over each keyword k (lines 13–16). A vertex x of type `Keyword` is created (line 14) with k stored in the property `name` (line 15). The creation of x can be accomplished via a query similar to (4). Also, an edge with predicate `HAS_KEYWORD` is created between the question vertex v and its keyword vertex x (line 16). This edge creation can be accomplished via a query similar to (5).

After creating these vertices and edges, G contains the data from the set of survey sheets $\mathcal{S}'$, and is returned as the output of the construction process of Algorithm 2 (line 17).

C. Constructing a KG from Shared-Language Data

The proposed domain-agnostic process for KG construction from shared-language data requires the data to be in the format described in Section IV-C1. The output KG and its ontology are discussed in Section IV-C2. The necessary shared-language data preprocessing is detailed in Section IV-C3, and the construction process is presented in Section IV-C4.

1) The Input Data: The process accepts shared-language data as a spreadsheet L with three columns: ***Term***, ***Meaning***, and ***Synonyms***. Each row corresponds to a single term with the entry in the ***Term*** column specifying the term name, the entry in the ***Meaning*** column providing a brief description or explanation of the term, and the entry/entries in the ***Synonyms*** column listing synonyms of the term. Synonyms can also be

Algorithm 3: Preprocessing Shared-Language Data.

Input: Shared-language sheet L and similarity threshold σ.
Output: Preprocessed shared-language sheet L'.

1: $L' \leftarrow L$;
2: **for** row $r \in L'$ **do**
3: $\quad$ *keywords* $\leftarrow$ extract_keywords(r[***Meaning***]);
4: $\quad$ *synonyms* $\leftarrow$ filter_keywords(r[***Term***], *keywords*, σ);
5: $\quad$ r[***Synonyms***] $\leftarrow$ r[***Synonyms***] $\cup$ *synonyms*;
6: **return** L';

extracted automatically (see Section IV-C3 for the details). See Section V-E1 for example input data.

2) The Output KG: The KG $G = (V, T, \tau, P, E)$ generated by the construction process of Section IV-C4 contains the data from the shared-language sheet L with appropriate relationships between synonyms. The vertex set V consists of all the unique entries from the ***Term*** and ***Synonyms*** columns of L. The set of vertex types $T = \{\texttt{Term}\}$, and the vertex-type labeling function τ maps all vertices to the type `Term` since synonyms are considered terms themselves. The set of predicates $P = \{\texttt{HAS\_SYNONYM}\}$, and the set of triples E consists of a single triple with predicate `HAS_SYNONYM` between each term t from the ***Term*** column and each of its synonyms s from the ***Synonyms*** column.

3) The Data Preprocessing: After formatting the input data according to the specifications in Section IV-C1, the data are preprocessed to extract additional synonyms above a threshold. While experts can contribute high-quality synonyms in the input, their contributions can require non-negligible manual efforts. Thus, automatic extraction of additional synonyms is beneficial. The purpose of these synonyms is primarily to enhance the resulting KG and its ability to facilitate the integration of multiple KGs (see Section IV-D for the details).

The data-preprocessing steps are outlined in Algorithm 3, which takes as input the shared-language sheet L, formatted as specified in Section IV-C1, along with the threshold σ that can be tuned via expert collaboration, see Section VI-C, and produces as output a preprocessed shared-language sheet L'.

First, the sheet L' is initialized with the data from the sheet L (line 1). Then, the algorithm iterates over each row, i.e., term t, of L' (lines 2–5). Keywords are extracted from the meaning of term t, which is found in the ***Meaning*** column of the row (line 3), via the same process used in Section IV-B3.

The keywords are then filtered down to a list of true synonyms (line 4) by computing the similarity s between the term t found in ***Term*** column of the row and each keyword k, and keeping only those keywords whose scores exceed the similarity threshold σ. The similarity is computed via a combination of three measures, which each capture different aspects of the relationship between terms and keywords.

1) The *TF-IDF [17] similarity measure* F quantifies the importance of a keyword based on its frequency and uniqueness in text. For the term t and the keyword k, TF-IDF captures the frequency of k in the meaning of t relative to its frequency in all the meanings found in the ***Meaning*** column of the shared-language sheet L'.
2) The *Word2Vec [18] similarity measure* W captures the semantic similarity of a term and keyword based on textual context. The Word2Vec model is pretrained on the meanings found in the ***Meaning*** column of the sheet L'. The model is then used to generate vector representations of the term t and the keyword k, and the cosine similarity [19] of the two vectors is calculated.
3) The *BERT [20] similarity measure* B provides a deep, context-aware comparison of the term and keyword. The BERT model was pretrained on the English language in [20]. The model is used to generate vector representations of the term t and the keyword k, and the cosine similarity [19] of the two vectors is calculated.

The final similarity between the keyword k and the term t is:

$$s(t,k) = \alpha \times F(t,k) + \beta \times W(t,k) + \gamma \times B(t,k) \quad (6)$$

Here, the weights α, β, γ can be tuned for a given domain and/or use case. For instance, in complex domains a greater weight γ can be assigned to the context-aware measure B to capture nuanced semantics, while in less specialized domains the statistical measure F can be prioritized via assigning a greater value to α. These weights enable the similarity function (6) to adapt to diverse domains.

After computing the similarity scores and applying the threshold σ, the remaining keywords (synonyms) are added to the ***Synonyms*** column of the row (line 5).

After iterating through all rows, i.e., terms, the sheet L' is returned as the output of Algorithm 3 (line 6).

4) The Construction Process: The process for constructing a KG from shared-language data is outlined in Algorithm 4. The algorithm takes as input the preprocessed shared-language sheet L' (output by Algorithm 3) and outputs the KG G described in Section IV-C2.

First, an empty KG G is made (line 1). Then, the algorithm iterates over each row, i.e., term, of L' (lines 2–8). A corresponding vertex v of type `Term` is created (line 3) with properties `name` and `meaning` extracted from the ***Term*** and ***Meaning*** columns of the row (line 4). The creation of v can be accomplished via the following query:

$$\texttt{MERGE}\ (v : \texttt{Term}\ \{\texttt{name} : n, \texttt{meaning} : m\}) \quad (7)$$

This query creates a new vertex v of type `Term` with properties $v.\texttt{name} = n$ and $v.\texttt{meaning} = m$, if such a vertex does not already exist.

Next, the algorithm iterates over each synonym s from the ***Synonyms*** column of the row (lines 5–8). A vertex w

Algorithm 4: Constructing a KG from Shared-Language Data.

Input: Preprocessed shared-language sheet L'.
Output: KG G containing the data from L'.

1: $G \leftarrow \emptyset$; // `initialize an empty KG` G
2: **for** row $r \in L'$ **do**
3: $\quad$ $v \leftarrow G$.create_vertex(`Term`);
4: $\quad$ v.`name` $\leftarrow r$[***Term***]; v.`meaning` $\leftarrow r$[***Meaning***];
5: $\quad$ **for** synonym $s \in row$[***Synonyms***] **do**
6: $\quad\quad$ $w \leftarrow G$.create_vertex(`Term`);
7: $\quad\quad$ w.`name` $\leftarrow s$;
8: $\quad\quad$ G.create_edge(v, `HAS_SYNONYM`, w);
9: **return** G;

Algorithm 5: Integrating Multiple KGs.

Input: Set of KGs $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$ and shared-language KG G_L.

Output: Integrated KG K.

1: $K \leftarrow G_1 \cup G_2 \cup \cdots \cup G_n$; // `initialize the KG` K
2: $synonym\_pairs \leftarrow$ find_synonym_pairs(G_L);
   // `Create edges between synonyms`
3: **for** pair $(A, B) \in synonym\_pairs$ **do**
4: **if** $A, B \in K$ **then**
5: K.create_edge(A, `HAS_SYNONYM`, B);
6: **return** K;

of type `Term` is created (line 6) with w.`name` $= s$ (line 7) to represent s. The creation of w can be accomplished via a query similar to (7). Also, an edge with predicate `HAS_SYNONYM` is created between the term vertex v and its synonym vertex w (line 8). This edge creation can be accomplished via the following query:

$$\begin{aligned} &\texttt{MATCH } (v : \texttt{Term } \{\texttt{name} : n\}), (w : \texttt{Term } \{\texttt{name} : s\}) \\ &\texttt{MERGE } (v) - [: \texttt{HAS\_SYNONYM}] \rightarrow (w) \end{aligned} \tag{8}$$

This query identifies the vertices v, w corresponding to the term and its synonym, and ensures that an edge of type `HAS_SYNONYM` is created between them.

After creating the vertices and edges, G contains the data from the shared-language sheet L', and is returned as the output of the construction process of Algorithm 4 (line 9).

D. Integrating Multiple KGs Using the Shared-Language KG

The integration process for combining multiple input KGs into a single, unified KG takes as input the KGs that are derived via the construction processes of Sections IV-A and IV-B from various data sources, such as experimental data and survey data. The integration is facilitated by a shared-language KG, constructed via the process of Section IV-C, that provides synonym mappings to ensure consistent alignment between terms across the input KGs.

The integration process is outlined in Algorithm 5, which takes as inputs the set of KGs $\mathcal{G}$ and the shared-language KG G_L, and outputs the KG K that integrates all the KGs from $\mathcal{G}$ according to the data in G_L.

First, the integrated KG K is populated with all the data from the input KGs in $\mathcal{G}$ (line 1). Then, the pairs of vertices connected via a `HAS_SYNONYM` edge are extracted from the shared-language KG G_L (line 2) via the following query:

$$\begin{aligned} &\texttt{MATCH } (A : \texttt{Term}) - [: \texttt{HAS\_SYNONYM}] \rightarrow (B : \texttt{Term}) \\ &\texttt{RETURN } A.\texttt{name}, \ B.\texttt{name} \end{aligned} \tag{9}$$

Next, the algorithm iterates over each synonym pair (A, B) (lines 3–5). If vertices corresponding to A and B exist in K (line 4), then an edge with predicate `HAS_SYNONYM` is created between them (line 5) via the following query:

$$\begin{aligned} &\texttt{MATCH } (v_A \ \{\texttt{name} : A\}), (v_B \ \{\texttt{name} : B\}) \\ &\texttt{MERGE } (v_A) - [: \texttt{HAS\_SYNONYM}] \rightarrow (v_B) \end{aligned} \tag{10}$$

This query identifies the vertices v_A, v_B corresponding to the terms A and B, and ensures that an edge of type `HAS_SYNONYM` is created between them.

After all appropriate `HAS_SYNONYM` edges are created between synonyms, the integrated KG K is returned as the output of the integration process of Algorithm 5 (line 6).

V. The Use Case with STEPS-Center Data

We now describe our implementation of the proposed INTEGRATE-KG workflow and detail its application to our STEPS-Center[4] use case in the materials science domain. The tools used to implement the workflow are presented in Section V-A, and the source data for the use case is described in Section V-B. Our instantiations of the workflow for the STEPS experimental, survey, and shared-language data are detailed in Sections V-C, V-D, and V-E, respectively. Finally, the integration of the STEPS use-case data sets into a unified, cohesive knowledge graph (KG) is discussed in Section V-E.

A. Tools Used in our Implementation of INTEGRATE-KG

Tools required to implement the INTEGRATE-KG workflow include a graph data-management system (DBMS), a graph query language, and a programming language with various specialized libraries. In our implementation, we used the Neo4j graph DBMS [21] with the Cypher graph query language [16]. We implemented the workflow in Python [22], using the `py2neo` [23] library for connecting to Neo4j and executing Cypher queries within Python code, and using the `pandas` [24], [25] library for reading and preprocessing the source data. Additionally, we used the `re` [26] library for the regular-expression matching mentioned in Section IV-B3, the `NLTK` [27] library for the natural language processing (NLP) steps described in Sections IV-B3 and IV-C3, and the `sklearn` [28], `gensim` [29], and `transformers` [30] libraries, respectively, for the *TF-IDF*, *Word2Vec*, and *BERT* similarity measures discussed in Section IV-C3.

B. The Source Data in the STEPS-Center Use Case

As a use case for testing the proposed INTEGRATE-KG workflow, we applied the workflow to data provided by researchers across multiple disciplines and teams within the STEPS Center. These data included experimental results, survey responses, and a shared-language repository. Integrating these data sets via the INTEGRATE-KG workflow generated a single, unified KG that gives STEPS scientists the ability to make significant new discoveries. Leveraging the shared-language data of Data Set 3 (see Section V-B3) enabled harmonization of the data from the different research disciplines and teams within STEPS.

The datasets generated by STEPS scientists are as follows:

1) Data Set 1: Experimental Data: This data was generated from various phosphate-sorption experiments using different materials. The data consist of spreadsheets detailing various experimental settings and measurements, e.g., sorbent materials, sorbent dosages, and phosphate-removal percentages. The results provide insights into the phosphate-binding efficacy of different materials, e.g., Al_2O_3, Fe_2O_3, and CeO_2.

2) Data Set 2: Survey Data: This data was collected as detailed in [31]. An online survey underwent IRB review (NC State, IRB protocol 25,268) and was then completed anonymously by 96 expert stakeholders in the U.S. and abroad who were familiar with and currently worked in a variety of fields and sectors related to phosphorus management. The

[4]https://steps-center.org

TABLE I
FRAGMENT OF THE RELATION SHEET FROM DATA SET 1 (SEE SECTION V-B1) USED WITH TABLE II AS INPUT TO THE KG CONSTRUCTION PROCESS OF SECTION IV-A4. EACH ROW DESCRIBES A SINGLE DATA OBJECT, AND THE COLUMNS REPRESENT DIFFERENT ATTRIBUTES.

Exp. ID	*Material*	*Sorbent Dosage (g/L)*	*Initial P Conc. (mg/L)*	*Avg. % P Removed*
E1	Al_2O_3	1	5	15.94
E2	Fe_2O_3	1	5	8.94
E3	CeO_2	1	5	92.32

TABLE II
FRAGMENT OF THE TRIPLE SHEET FROM DATA SET 1 (SEE SECTION V-B1) USED WITH TABLE I AS INPUT TO THE KG CONSTRUCTION PROCESS OF SECTION IV-A4. EACH ROW IS A TRIPLE OF THE FORM (st, p, ot), WHERE st AND ot ARE VERTEX TYPES AND p IS A PREDICATE.

Subject	*Predicate*	*Object*
`Exp. ID`	`USED`	`Material`
`Material`	`USED_IN_DOSAGE`	`Sorbent Dosage (g/L)`
`Exp. ID`	`STARTED_AT`	`Initial P Conc. (mg/L)`
`Exp. ID`	`RESULTED_IN`	`Avg % P Removed`
`Material`	`AFFECTED`	`Avg % P Removed`

survey aimed to elicit information regarding stakeholder perceptions, concerns, and needs related to achieving phosphorus sustainability. The results from this work offered a nuanced understanding of the landscape of phosphorus sustainability, revealing key stakeholder concerns and priorities. The data consist of survey questions and corresponding responses.

3) Data Set 3: Shared-Language Data: This data consists of a bank of terms identified by STEPS participants as unknown or polysemous, i.e., having multiple meanings across disciplines. The data include contextual meanings and synonyms from STEPS participants to assist in the resolution of terminological discrepancies across disciplines and to create a shared language within STEPS.

C. Constructing a KG from STEPS Experimental Data

We now discuss our experience of constructing a KG from the experimental data in our STEPS use case. Those data come from Data Set 1, see Section V-B1 for the details.

1) The Input Data: To arrive at the required input format specified in Section IV-A1, we used the relation sheet R provided by Data Set 1 and generated a triple sheet S with the help of domain scientists. Table I shows a fragment of the relation sheet R, which contains data about the experiments conducted. Each data object (row) in R contains information about a single experiment, e.g., the experiment (exp.) ID, the material used, and the sorbent dosage. For example, the first row of Table I details the experiment with ID *E1*, which used the material Al_2O_3 at a sorbent dosage of 1 *g/L*; the experiment had an initial Phosphorus (P) concentration (conc.) of 5 *mg/L* and resulted in an average P removal of 15.94%.

Table II shows a fragment of the triple sheet S, which specifies the triple types desired in the resulting KG. Each row contains the subject type, predicate, and object type for a single triple type. For example, the first row specifies the triple type (`Exp. ID`, `USED`, `Material`), which indicates that each `Exp. ID` and the corresponding `Material` are connected via the `USED` relationship.

2) The Output KG: The KG $G_E = (V_E, T_E, \tau_E, P_E, E_E)$ produced by the construction process of Section IV-A4 contains the data from the relation sheet R organized according to the triple sheet S. The vertex set V_E consists of all the unique entries in R, i.e., $V_E = \{$*E1*, *E2*, *E3*, Al_2O_3, Fe_2O_3, CeO_2, 1, 5, 15.94, 8.94, 92.32$\}$. The set of vertex types T_E consists of the column headers of R, i.e., $T_E = \{$`Exp. ID`, `Material`, `Sorbent Dosage (g/L)`, `Initial P Conc. (mg/L)`, `Avg. % P Removed`$\}$. The vertex-type labeling function τ_E maps each entry in R to its corresponding column header, e.g., $\tau($*E1*$)$ = `Exp. ID` and $\tau(Al_2O_3)$ = `Material`. The predicate set P_E consists of all the entries from the ***Predicate*** column of S, i.e., $P = \{$`USED`, `USED_IN_DOSAGE`, `STARTED_AT`, `RESULTED_IN`, `AFFECTED`$\}$. The set of triples E_E consists of a single triple (s, p, o) of each triple type (st, p, ot) in S for each data object d in R, where s and o are the entries from d in the columns st and ot. For instance, $($*E1*, `USED`, $Al_2O_3) \in E_E$, representing that the experiment *E1* used the material Al_2O_3.

3) The Data Preprocessing: Following the data-preprocessing steps of Section IV-A3, we verified that the relation sheet R and the triple sheet S of Section V-C1 were compatible. That is, the triple types in S used vertex-type headers from R. We also used data imputation to fill in missing values, flagging the imputed values for expert revision, see Section VI-A for the details.

4) The Construction Process: We implemented the construction process of Section IV-A4 using the tools mentioned in Section V-A. Using the relation sheet R and the triple sheet S of Section V-C1 as inputs, the process generated the KG $G_E = (V_E, T_E, \tau_E, P_E, E_E)$ described in Section V-C2.

D. Constructing a KG from STEPS Survey Data

We now discuss our experience of constructing a KG from the survey data in our STEPS use case. Those data come from Data Set 2, see Section V-B2 for the details.

1) The Input Data: To align with the required input format specified in Section IV-B1, we used the set of survey sheets $\mathcal{S}$ provided by Data Set 2, where each sheet details a single survey question and its responses. Table III shows a sample survey sheet for the question stated in the first row. The subsequent rows list the participant responses, e.g., "Yes," along with the number of participants who selected the response and the percentage out of the total number of responses.

TABLE III
SAMPLE SURVEY SHEET FROM DATA SET 2 (SEE SECTION V-B2) USED AS INPUT TO THE DATA PREPROCESSING OF SECTION IV-B3. THE FIRST ROW INCLUDES THE SURVEY QUESTION AND SUBSEQUENT ROWS PROVIDE THE DISTRIBUTION OF THE PARTICIPANT RESPONSES.

Participant involvement in P management. Responses to "Do you currently work in, conduct research, and/or are involved in activities related to phosphorus management?"		
Response Text	***Count***	***Percent***
Yes	78	81.3%
Maybe	4	4.2%
No	13	13.5%
Prefer not to answer	1	1%

2) The Output KG: The KG $G_S = (V_S, T_S, \tau_S, P_S, E_S)$ produced by the construction process of Section IV-B4 contains the data from the set of survey sheets $\mathcal{S}$, along with keywords extracted during the preprocessing of Section V-D3. The vertex set V_S consists of all the unique questions and responses from $\mathcal{S}$, and all the unique keywords, i.e., $V_S = \{$*"Do*

TABLE IV
FRAGMENT OF THE SHARED-LANGUAGE SHEET FROM DATA SET 3 (SEE SECTION V-B3) USED AS INPUT TO THE DATA PREPROCESSING OF SECTION IV-C3. EACH ROW DETAILS A SINGLE TERM, INCLUDING ITS NAME, MEANING, AND SYNONYMS.

Term	***Meaning***	***Synonyms***
Phosphorus management	Managing phosphorus flux in ecosystems, focusing on its movement and removal to maintain environmental balance.	P Management, P Removal
Alumina	Chemical compound aluminum oxide that is widely used as an adsorbent, as a catalyst, and in various industrial applications due to its hardness and high melting point.	Al_2O_3, Activated alumina

... management?", *Yes*, *Maybe*, *No*, *Prefer not to answer*, *work*, *research*, *phosphorus management*$\}$. The vertex-types set $T_S = \{$`Question`, `Response`, `Keyword`$\}$. The vertex-type labeling function τ_S maps each question, response, and keyword to its corresponding type in T_S, e.g., $\tau_S($*Yes*$) =$ `Response` and $\tau_S($*research*$) =$ `Keyword`. The predicate set $P_S = \{$`HAS_RESPONSE`, `HAS_KEYWORD`$\}$. The set of triples E_S consists of a single triple $(q,$ `HAS_RESPONSE`$, r)$ between each question q and each of its responses r, and a single triple $(q,$ `HAS_KEYWORD`$, k)$ between each question q and each of its keywords k. For instance, (*"Do ... management?"*, `HAS_RESPONSE`, *Yes*$) \in E_S$, representing that the question *"Do ... management?"* has the response *Yes*.

3) The Data Preprocessing: We implemented the data-preprocessing steps of Section IV-B3 using the tools mentioned in Section V-A. Using the set $\mathcal{S}$ of survey sheets of Section V-D1 as input, the preprocessing generated the set $\mathcal{S}'$ of preprocessed sheets. For example, the question *"Do ... management?"* and keywords *work*, *research*, and *phosphorus management* were extracted from the sheet of Table III.

4) The Construction Process: We implemented the construction process of Section IV-B4 using the tools mentioned in Section V-A. Using the set $\mathcal{S}'$ of preprocessed survey sheets of Section V-D3 as input, the process generated the KG $G_S = (V_S, T_S, \tau_S, P_S, E_S)$ described in Section V-D2.

E. Constructing a KG from STEPS Shared-Language Data

We now discuss our experience of constructing a KG from the shared-language data in our use case. Those data come from Data Set 3, see Section V-B3 for the details.

1) The Input Data: To align with the input format specified in Section IV-C1, we used the shared-language sheet L provided by Data Set 3. Table IV shows a fragment of L, which contains terms, meanings, and synonyms related to phosphorus sustainability. Each row details a single term, giving the term name, meaning, and synonyms. For example, the first row of Table IV specifies the term *phosphorus management*, which means *"managing phosphorus flux in ecosystems..."* and has synonyms *P management* and *P removal*.

2) The Output KG: The KG $G_L = (V_L, T_L, \tau_L, P_L, E_L)$ generated by the construction process of Section IV-C4 contains the data extracted from the shared-language sheet L. The vertex set V_L consists of all the terms and synonyms in L, i.e., $V_L = \{$*Phosphorus management*, *P management*, *P removal*, *Alumina*, *Al_2O_3*, *Activated alumina*$\}$. The vertex-types set $T_L = \{$`Term`$\}$. The vertex-type labeling function τ_L maps each vertex in V_L to the type `Term`, e.g., $\tau($*Alumina*$) =$ `Term`. The predicate set $P_L = \{$`HAS_SYNONYM`$\}$. The set of triples E_L consists of a single triple $(t,$ `HAS_SYNONYM`$, s)$ between each term t from the ***Term*** column of L and each of its synonyms s from the ***Synonyms*** column of L. For instance, (*Alumina*, `HAS_SYNONYM`, *Al_2O_3*$) \in E_L$, representing that the term *Alumina* has the synonym *Al_2O_3*.

3) The Data Preprocessing: We implemented the data-preprocessing steps of Section IV-C3 using the tools mentioned in Section V-A. Using the shared-language sheet L of Section V-E1 and the threshold $\sigma = 80\%$ as inputs, the preprocessing generated the preprocessed shared-language sheet L'. The weights $\alpha = 0.2$, $\beta = 0.3$, and $\gamma = 0.5$ were used to compute the similarity s of Eq. 6.[5] For example, the synonym *aluminum oxide* was extracted from the meaning of the term *alumina* of the shared-language sheet of Table IV.

4) The Construction Process: We implemented the construction process of Section IV-C4 using the tools mentioned in Section V-A. Using the preprocessed shared-language sheet L' of Section V-E3 as input, the process generated the KG $G_L = (V_L, T_L, \tau_L, P_L, E_L)$ described in Section V-E2.

F. Integrating Multiple KGs Using the Shared-Language KG

For our use case, it would be helpful to STEPS scientists if the experimental and survey KGs G_E and G_S were integrated into a single KG K that could enable them to investigate connections between their previously disparate data.

We implemented the integration process of Section IV-D using the tools mentioned in Section V-A. Using the set $\mathcal{G} = \{G_E, G_S\}$ of KGs and the shared-language KG G_L as inputs, the process generated the integrated KG K.

VI. HUMANS-IN-THE-LOOP IN THE WORKFLOW

We now discuss the role of domain scientists as humans-in-the-loop in INTEGRATE-KG. Their involvement can be beneficial before, during, and after knowledge graph (KG) construction and integration to ensure that the resulting KG will be both accurate and useful for their specific needs.

A. Constructing a KG from Experimental Data

Domain scientists can assist in KG-construction from experimental data by determining the relationships between data-sheet entries, which form the triple sheet, and by handling missing data. To build the triple sheet, we recommend following the data-preprocessing steps outlined in [2] with humans-in-the-loop. To handle missing data, domain scientists can intervene after KG construction. During construction, missing data are flagged, so the scientists can review the flagged vertices in the KG and submit revisions at any point.

B. Constructing a KG from Survey Data

Domain scientists can assist in KG-construction from survey data by refining the keywords that are automatically extracted from the survey questions. To do this, we recommend providing the survey questions and keywords to domain scientists and allowing them to submit revisions, e.g., additions, modifications, or deletions, before KG construction. This ensures the accuracy of the final set of keywords, preventing any spurious keywords from making it into the resulting KG. In our use case, domain scientists added the keyword *phosphorus removal* for the survey question presented in Table III because phosphorus management (the focus of the question) can lead to phosphorus removal.

[5]The weights were chosen based on performance during validation tests.

C. Constructing a KG from Shared-Language Data

Domain scientists can assist in KG-construction from shared-language data by refining the synonyms that are automatically extracted from the term meanings. To do this, we recommend following the same strategy used in Section VI-B. They can also assist in tuning the similarity threshold σ used to filter the synonyms. Their assistance ensures that the final set of synonyms used when constructing the KG aligns with the expectations and understanding of the domain scientists. In our use case, domain scientists added the synonym *sapphire* for the term *alumina* originating from the shared-language data of Table IV because sapphire is the single crystalline form of alumina.

D. Refining the Integrated KG

After integrating the disparate data sets into a single, unified KG, domain scientists can continue to enrich the resulting KG. In their review and use of the KG, they can submit additional relationships between entities across the data sets, providing richer connections that can be used in their further data analysis and exploration. For instance, in our STEPS-Center KG K (see Section V), domain scientists added the relationship `MAY_FACILITATE` between the 92.32 vertex of type `Avg. % P Removed` originating from the experimental data of Table I and the *phosphorus management* vertex of type `KEYWORD` originating from the survey data of Table III. This relationship bridges experimental findings with stakeholder priorities reported in the survey responses.

E. Using the Integrated KG

Domain scientists can leverage the integrated KG to explore queries across their heterogeneous data. In our use case, STEPS scientists are now able to query connections between material performance, which was originally reported in their experimental data, and stakeholder feedback, which was originally reported in their survey data. With the continuing ability to refine the KG, it can become a dynamic, interdisciplinary tool for data exploration, as opposed to a static data repository. Data analytics over such a tool can yield actionable insights and informed decision-making across domains.

VII. Conclusion

In this paper, we presented the INTEGRATE-KG workflow for integrating heterogeneous data from multiple sources into a single unified knowledge graph (KG) that can open new avenues for data analytics. The workflow aims to address the challenge of automating integration into unified KGs of diverse data that potentially use different terminology, with the help of the available shared languages to resolve terminology clashes. While domain-agnostic, INTEGRATE-KG is also domain aware, due to the opportunities available for the involvement of humans-in-the-loop in the process. A key feature of INTEGRATE-KG is in its use of shared languages to automate semantics-level terminology alignment across the individual data contributions after they have been submitted for integration. In addition, INTEGRATE-KG includes a module for automatically enriching the shared languages, with opportunities for domain experts to provide corrections and feedback. We introduced the workflow, reported on our experiences with applying it to experimental, survey, and shared-language data in the STEPS Center focusing on phosphorus sustainability, and provided suggestions for involving domain scientists in INTEGRATE-KG as humans-in-the-loop. Future work includes adapting the presentation of analytical results on the KGs generated by INTEGRATE-KG, such that the terminology used in the presentation would be tailored to audiences across different disciplines.

Acknowledgment

The authors would like to thank Ashton Merck for her valuable assistance with survey-data organization and analysis.

References

[1] A. Hogan, E. Blomqvist *et al.*, "Knowledge graphs," *ACM Comput. Surv.*, vol. 54, no. 4, pp. 1–37, 2021.

[2] K. Schatz, P.-Y. Hou *et al.*, "BUILD-KG: Integrating Heterogeneous Data Into Analytics-Enabling Knowledge Graphs," in *IEEE Big Data*, 2023, pp. 2965–2974.

[3] J. L. Martinez-Rodriguez, I. Lopez-Arevalo, and A. B. Rios-Alvarado, "OpenIE-based approach for Knowledge Graph construction from text," *Expert Syst. Appl.*, vol. 113, pp. 339–355, 2018.

[4] P. Yang, B. Song, and Z. Zhang, "Research on knowledge graph construction methods for news domain," *AJST*, vol. 11, no. 1, pp. 58–64, 2024.

[5] A. V. Kannan, D. Fradkin *et al.*, "Multimodal knowledge graph for deep learning papers and code," in *ACM CIKM*, 2020, pp. 3417–3420.

[6] L. Yao, C. Mao, and Y. Luo, "KG-BERT: BERT for knowledge graph completion," *arXiv preprint arXiv:1909.03193*, 2019.

[7] G. Agrawal, Y. Deng *et al.*, "Building knowledge graphs from unstructured texts: Applications and impact analyses in cybersecurity education," *Information*, vol. 13, no. 11, p. 526, 2022.

[8] J. H. Caufield, T. Putman *et al.*, "KG-Hub—building and exchanging biological knowledge graphs," *Bioinformatics*, vol. 39, no. 7, 2023.

[9] A. Rossanez and J. C. dos Reis, "Generating knowledge graphs from scientific literature of degenerative diseases," in *SEPDA@ISWC*, 2019, pp. 12–23.

[10] M. Gharibi, A. Zachariah, and P. Rao, "FoodKG: A tool to enrich knowledge graphs using machine learning techniques," *Front. Big Data*, vol. 3, p. 12, 2020.

[11] B. Zhao, Y. Zhao, and Y. Mao, "A method for judicial case knowledge graph construction based on event extraction," in *ICIIT*, 2024, p. 62–69.

[12] B. Trsedya, J. Qi, and R. Zhang, "Entity alignment between knowledge graphs using attribute embeddings," in *AAAI*, 2019.

[13] L. Asprino, E. Daga *et al.*, "Knowledge Graph Construction with a *Façade*: A Unified Method to Access Heterogeneous Data Sources on the Web," *ACM TOIT*, vol. 23, no. 1, pp. 1–31, 2023.

[14] M. Kejriwal, *Domain-Specific Knowledge Graph Construction*. Springer International Publishing, 2019.

[15] M. Hofer, D. Obraczka *et al.*, "Construction of knowledge graphs: State and challenges," *arXiv preprint arXiv:2302.11509*, 2023.

[16] The Neo4j Team, "The Neo4j Cypher manual v5," 2022.

[17] K. Sparck Jones, *A statistical interpretation of term specificity and its application in retrieval*. Taylor Graham Publishing, 1988, p. 132–142.

[18] T. Mikolov, K. Chen *et al.*, "Efficient estimation of word representations in vector space," in *ICLR Workshop Track Proceedings*, 2013.

[19] A. Singhal, "Modern information retrieval: A brief overview," *IEEE Data Eng. Bull.*, vol. 24, no. 4, pp. 35–43, 2001.

[20] J. Devlin, M. Chang *et al.*, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT*, 2019, pp. 4171–4186.

[21] The Neo4j Team, "The Neo4j operations manual v5," 2022.

[22] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. CreateSpace, 2009.

[23] N. Small, "The py2neo handbook," 2023.

[24] Wes McKinney, "Data Structures for Statistical Computing in Python," in *Python in Sci. Conf.*, 2010, pp. 56 – 61.

[25] The pandas development team, "pandas-dev/pandas: Pandas," 2020.

[26] Python Software Foundation, "re - regular expression operations," 2024.

[27] S. Bird, E. Klein, and E. Loper, "Natural language processing with python: analyzing text with the natural language toolkit," 2009.

[28] F. Pedregosa, G. Varoquaux *et al.*, "Scikit-learn: Machine learning in Python," *JMLR*, vol. 12, pp. 2825–2830, 2011.

[29] R. Řehůřek and P. Sojka, "Gensim: Topic modelling for humans," 2010.

[30] T. Wolf, L. Debut *et al.*, "Transformers: State-of-the-art natural language processing," in *EMNLP: System Demonstrations*, 2020, pp. 38–45.

[31] K. Grieger, A. Merck *et al.*, "What are stakeholder views and needs for achieving phosphorus sustainability?" *Environ. Syst. Decis.*, vol. 44, pp. 114–125, 2024.