Expert Systems With Applications 228 (2023) 120239

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

KNIT: Ontology reusability through knowledge graph exploration

Jorge Rodríguez-Revello [a], Cristóbal Barba-González [a], Maciej Rybinski [b], Ismael Navas-Delgado [a,*]

[a] ITIS Software, Universidad de Málaga, Málaga 29071, Spain
[b] CSIRO Data61, Sydney, NSW, Australia

ARTICLE INFO

Keywords:
Life sciences
Ontology
Ontology learning
Knowledge graphs

ABSTRACT

Ontologies have become a standard for knowledge representation across several domains. In Life Sciences, numerous ontologies have been introduced to represent human knowledge, often providing overlapping or conflicting perspectives. These ontologies are usually published as OWL or OBO, and are often registered in open repositories, e.g., BioPortal. However, the task of finding the concepts (classes and their properties) defined in the existing ontologies and the relationships between these concepts across different ontologies – for example, for developing a new ontology aligned with the existing ones – requires a great deal of manual effort in searching through the public repositories for candidate ontologies and their entities. In this work, we develop a new tool, *KNIT*, to automatically explore open repositories to help users fetch the previously designed concepts using keywords. User-specified keywords are then used to retrieve matching names of classes or properties. *KNIT* then creates a *draft* knowledge graph populated with the concepts and relationships retrieved from the existing ontologies. Furthermore, following the process of ontology learning, our tool refines this first draft of an ontology. We present three BioPortal-specific use cases for our tool. These use cases outline the development of new knowledge graphs and ontologies in the sub-domains of biology: genes and diseases, virome and drugs.

1. Introduction

Over the last decades, knowledge has become one of the main assets of organisations. Knowledge management and processing have consequently turned into critical challenges. The Semantic Web is an enabling technology for the Life Sciences because it can significantly enhance the efficiency and effectiveness of the integration of diverse biological resources. In particular, the adoption of ontologies is widespread, thus leading to increased research activity, in medicine and biology (Ashburner et al., 2000; Beisswanger, Schulz, Stenzhorn, & Hahn, 2008; Roldán-García, García-Godoy, & Aldana-Montes, 2016; Smith, 2004).

The primary function of a domain ontology is to accurately represent knowledge by formally defining the concepts that surround and act within a domain (Guarino, 1995). Life Sciences is the product of the union of a large number of research fields that look at life as a field of study, placing plant, animal and human organisms in the focus (National Research Council et al., 2012). There are multiple works related to the application of ontologies in Life Sciences. Zhang, Sun, Diao, Zhao, and Shu (2021b) propose a method based on a knowledge graph integration, which predicts adverse drug reactions. Furthermore, Wang et al. (2021) define knowledge graphs to make drug–drug interaction predictions. Other studies, e.g., Al-Saleem et al. (2021) and Xiong, Huang, Wang, Liu, and Zhang (2021), leverage prior knowledge extracted from knowledge graphs to improve drug reuse. All these works point to the advantages stemming from building new ontologies to express task-specific knowledge.

However, the design of ontologies is not a trivial task. It requires coordinating experts' domain knowledge with the knowledge of semantic Web technologies. Nevertheless, obtaining concepts related to a domain is challenging due to the heterogeneity of existing conceptualisations and the availability of ontologies. Furthermore, the ontologies' documentation or license is often deficient or even not accessible (Fernández-López, Poveda-Villalón, Suárez-Figueroa, & Gómez-Pérez, 2019). In this context, the most widely adopted methodology is the *Ontology 101 methodology* (Noy et al., 2001), which states that every ontology must be created following seven fundamental steps: (1) determining the domain and scope of the ontology, (2) considering the reuse of existing ontologies, (3) listing the essential terms in the ontology, (4) defining classes and hierarchies, (5) defining class properties,

* Corresponding author.
E-mail addresses: rodriguezrevellojj@uma.es (J. Rodríguez-Revello), cbarba@uma.es (C. Barba-González), maciek.rybinski@csiro.au (M. Rybinski), ismael@uma.es (I. Navas-Delgado).

https://doi.org/10.1016/j.eswa.2023.120239
Received 16 January 2023; Received in revised form 10 April 2023; Accepted 22 April 2023
Available online 8 May 2023

(6) defining facets, and finally, (7) creating instances. Another methodology, *Methontology* (Gómez-Pérez & Rojas-Amaya, 1999), focuses on searching for previously developed ontologies to check whether any of the knowledge they contain could be reused. Thus, the standard step in those design processes is ontology reuse during the design stage (Doran, Tamma, & Iannone, 2007). Additionally, ontology engineering good practices recommend reusing as many terms from the same ontology as possible rather than importing them from a large set of ontologies (Alharbi, Tamma, & Grasso, 2021). This recommendation is focused on increasing the overall coherence of the developed ontology.

The eXtensible ontology development (XOD) (He et al., 2018) proposes four principles for ontology development: (1) ontology term reuse, (2) ontology semantic alignment, (3) ontology design patterns, (4) usage for new term generation and existing term editing and community extensibility. Another ontology development strategy is the Minimum Information to Reference an External Ontology Term (MIREOT) (Courtot et al., 2011) principle, which proposes using the tiniest information of an external ontology term of direct interest to a target ontology. Thus, MIREOT presents the following steps: (1) source ontology URI; (2) source term URI; and (3) target direct superclass URI. However, none of the proposed methodologies indicates how this reuse should be approached or automated.

Ontologies are usually collected in systems like ontology directories, repositories, libraries, and archives, such as Bioportal (Noy et al., 2009), Cupboard (d'Aquin & Lewen, 2009), the OBO Foundry initiative (Smith et al., 2007) and OLS (Côté, Jones, Martens, Apweiler, & Hermjakob, 2008). The main objective of those web-based systems is to make it possible for users to locate and utilise one or more ontologies from a 'live' library of ontologies (d'Aquin & Noy, 2012). The main approaches for linking to existing knowledge are based on two key ideas: the automated discovery of links between ontologies using mappings between their classes (Euzenat et al., 2007) and the search for existing ontologies based on ontology-level relationships (Allocca, d'Aquin, & Motta, 2009). While the availability of ontology repositories represents an important step towards reusability, little attention has been given to automatically steering the ontology design process towards reusability. This is also crucial in the FAIR principles (Wilkinson et al., 2016). The FAIR principles are a collection of guidelines by which to improve the Findability, Accessibility, Interoperability, and **Reusability** of data objects. However, these repositories do not provide mechanisms to support knowledge reuse from the registered ontologies in the design process.

In recent years, numerous studies have focused on semi-automatic ontology creation. Most concentrate on developing ontologies based on text analysis through text mining and deep learning (Al-Aswadi, Chan, & Gan, 2020; Asim, Wasim, Khan, Mahmood, & Abbasi, 2018; Dahab, Hassan, & Rafea, 2008; Kaushik & Chatterjee, 2018). However, they do not provide a mechanism to reuse other ontologies automatically, as the effectiveness of automatically discovering the relations between concepts is still unsatisfactory (Albukhitan, Helmy, & Alnazer, 2017; Browarnik & Maimon, 2015).

In this work, we propose *KNIT* (KNowledge recyclIng neTworks), an ontology design tool system with two primary goals. Firstly, we propose a methodology to seek and rank concepts (i.e., classes, objects and data properties) in existing knowledge structures, given a domain context. And secondly, our system facilitates developing an ontology with the classes and properties retrieved in the above step. Thus, KNIT implements a workflow geared towards detecting the relevant knowledge (classes, data and object properties) in existing ontologies, analysing their interconnections and relationships, representing them with a knowledge graph, and finally transforming it into an ontology in *OWL* format.

As a proof-of-concept, we show how *KNIT* searches and reuses concepts from ontology repositories in the *BioPortal* ecosystem. In this implementation, the graph is built using *Neo4j*[1] graph data platform due to their features for managing graphs.

Thus, our work covers the research gap by providing a solution to discover and retrieve the most relevant ontologies in a process geared towards reusing their entities in the design process. The main contributions of this work can be summarised as follows:

- A methodology to reuse existing knowledge by ad-hoc creation of knowledge graphs, interlinking relevant classes and properties from existing ontologies to help domain experts automatically build the first version of the ontology required in their domain.
- An open-source tool for supporting the design of ontologies based on reusing concepts from existing ontologies in BioPortal.
- An evaluation of the proposed tool in the context of five realistic use cases, all of them related to Life Sciences data: comorbidities in patients with COVID-19, the impact of the virome on human intestinal health, a challenge of the Semantic Web and, and finally, two case studies using two target ontologies hosted on Bioportal.

2. Background and related work

This section describes background concepts in the Semantic Web field to make this paper self-contained. A state-of-the-art review is also provided to highlight the main differences between the related works and the proposed approach.

2.1. Background concepts and technologies

Under Noy et al. (2001), an *Ontology* provides a formal representation of the real world, which defines an explicit description of the concepts of a concrete domain expressed as classes, properties related to the concepts and constraints derived from the described properties. Ontologies are integrated into the *W3C* standard stack of the Semantic Web.[2]

Knowledge graphs employ a graph-based data architecture to store knowledge in scenarios where integrating, organising, and extracting value from several data sources at scale is necessary (Hogan et al., 2021). A knowledge graph is a semantic graph composed of vertices (or nodes) and edges. The vertices represent concepts or entities, which refer to the main categories of objects, such as molecule and protein. An entity is a physical object in the real world, such as an illness (e.g. cancer). The edges describe the semantic relationships between concepts or entities (Ji, Pan, Cambria, Marttinen, & Philip, 2021).

RDF Resource Description Framework (McBride, 2004) is a W3C recommendation which defines a language capable of describing resources on the web as graphs. Thus, RDF graphs are defined in terms of triples consisting of subject-predicate and object. Similarly, RDF Schema (RDFS) can define vocabularies using the RDF description (Staab & Studer, 2010).

OWL is the standard web ontology language proposed by W3C and extends RDF and RDFS with more vocabulary. OWL is employed by applications to describe terms and their interrelationships (Ding & Peng, 2004).

OWL-DL is syntactic description that gives maximum expressiveness while retaining computational completeness and decidability (McGuinness et al., 2004).

SPARQL is a query language for accessing RDF stores. It is the query language recommended by W3C (Harris, Seaborne, & Prud'hommeaux, 2013) to manage RDF graphs (Prud et al., 2006), due to supporting queries and web data sources identified by URIs.

Cypher is a query language for property graphs. Originally designed and implemented as part of the Neo4j graph database however, today is also used by a number of commercial database companies and researchers (Francis et al., 2018)

[1] https://neo4j.com/

[2] https://www.w3.org/standards/semanticweb/

2.2. Related work

Ontology learning is the effort to automate or semi-automate ontology creation processes (Khadir, Aliane, & Guessoum, 2021). In Cimiano, Mädche, Staab, and Völker (2009), ontology learning is contemplated as a data mining problem, giving as a solution to the issue a system built on four modules: a management module, a coordination module, a processing module, and an algorithm module.

The authors in Gil and Martin-Bautista (2014) approach the question from a systematic methodology for learning ontologies, using a semi-automated and user-centred approach. They allow human intervention at different moments of the creation process. This systematic methodology offers a visual and manipulable workflow to study the methods and tools used for the various resources. The work focuses on analysing the fusion of existing knowledge and the enrichment of existing ontologies. However, the systematic methodology does not focus on creating ontologies from scratch.

To address this issue, Kumar, Kumar, and Singh (2016) present an approach based on the extraction of patterns through linguistic and statistical studies. Their method relies on the construction of certain rules and patterns related to the ontological components to be extracted. Following the linguistic and statistical approach, in Kietz, Maedche, and Volz (2000), the authors propose a semi-automatic ontology acquisition method which iteratively improves the output ontology. This process is focused on a central ontology, an existing generic ontology, which is modified by employing rules such as taxonomical term dictionaries, statistical techniques and association rules, among others.

The *RENT* algorithm (Kaushik & Chatterjee, 2018) is based on the extraction of terms utilising regular expressions and textual processing techniques. RENT attributes different weights to words based on the frequency of use in the study corpus and patterns of relationships between terms. This work proposes two approaches for relation extraction: the *mOIE* approach (based on modified open information extraction) and the *RelExOnt* approach (based on relation extraction for ontology).

Regarding automatic processes, there are two main areas of ongoing research, automated learning and deep learning. Regarding the first field, we note the *DL-FOIL* (Fanizzi, d'Amato, & Esposito, 2008) tool, which can learn from the logical descriptions of the concepts using a supervised process. The authors in Jiang, Huang, Nickel, and Tresp (2012) present a combination of deductive reasoning and automatic learning.

Nowadays, we can observe intensified efforts focusing on deep learning approaches. There are studies at the intersection of machine learning, pattern recognition, optimisation, neural networks and graphical modelling. Regarding this intersection, in Petrucci, Ghidini, and Rospocher (2016) the authors present the idea of creating logical description formulas (DL) from natural language through the use of neural networks. On this point, Casteleiro et al. (2016) introduce a methodology where some semantic distribution models are utilised together with word2vec embeddings (Mikolov, Chen, Corrado, & Dean, 2013; Mikolov, Sutskever, Chen, Corrado, & Dean, 2013). A similar method, which also uses word2vec, is proposed in the work (Albukhitan et al., 2017) with the addition of seed terms and the extraction of related terms after setting an experimental similarity threshold.

In Life Sciences, in Arguello Casteleiro et al. (2018) the authors employ the *word2vec* to automatically identify acceptable free-text terms for genes and proteins in a corpus of millions of scientific publications. Additionally, they use the biological knowledge of cardiovascular disease ontology (*CVDO*) to improve the performance of word2vec.

Ontology reuse is creating new ontologies using existing ontological information as input (Caldarola & Rinaldi, 2016). One can differentiate between ontology merging and integration depending on the information in the knowledge sources and if their domains coincide. Current ontology engineering approaches address just a tiny portion of reusability difficulties (Pinto & Martins, 2001).

Table 1
KNIT utilises basic OWL-DL semantic syntax to define ontologies formally.

Descriptions	Abstract syntax	DL Syntax
Operators	$intersection(C_1, C_2, \ldots, C_n)$	$C_1 \sqcap C_2 \sqcap \cdots \sqcap C_n$
	$union(C_1, C_2, \ldots, C_n)$	$C_1 \sqcup C_2 \sqcup \cdots \sqcap C_n$
Restrictions	for at least 1 value V from C	$\exists V.C$
	for all values V from C	$\forall V.C$
	R is Symmetric	$R \equiv R^-$
Class Axioms	$A\ partial(C_1, C_2, \ldots, C_n)$	$A \sqsubseteq C_1 \sqcap C_2 \sqcap \cdots \sqcap C_n$
	$A\ complete(C_1, C_2, \ldots, C_n)$	$A \equiv C_1 \sqcap C_2 \sqcap \cdots \sqcap C_n$

At the same time, many information sources do not explicitly accept extra ontological primitives like attributes and axioms. Domain experts have trouble understanding such complicated structures, making integrating them into the target ontology challenging (Doran et al., 2007). In this sense, *MoodTool* (Doran et al., 2007) is software to extract entities from ontologies to reuse them. Nevertheless, a flaw in its implementation is that the concepts inherited from other ontologies are not obtained.

OntoFox (Xiang, Courtot, Brinkman, Ruttenberg, & He, 2010) fetches user-specified terms and their annotations from source ontologies and assigns them under defined superclass(es) in target ontologies. Ontodog (Zheng, Xiang, Stoeckert, & He, 2014) and Ontobull (Edison, Jie, Smith, Yongqun, et al., 2016) are based on OntoFox, and are designed for obtaining terms and axioms from a set of ontologies. Many other tools also support ontology term reuse. The Protégé MIREOT plugin Hanna et al. (2012) and OntoMaton (Maguire, González-Beltrán, Whetzel, Sansone, & Rocca-Serra, 2013) also facilitate terms reuse. Furthermore, ROBOT (Overton, Dietze, Essaid, Osumi-Sutherland, & Mungall, 2015) is a Java command-line utility that supports extracting ontology words. Besides, Ontobee (Ong et al., 2017), on its web page,[3] has a function that displays all other ontologies that utilise a specific term.

In contrast to the previous approaches, our system, *KNIT*, automates discovering relevant ontologies, their concepts, attributes, entities and axioms in any repository of ontologies. It also keeps track of the mappings between ontologies in the domain of the ontology repository. *KNIT* allows specifying the input terms without their URI, then helping the user with the re-utilisation of axioms. Therefore, *KNIT* addresses the research gap by providing a solution to retrieve the most relevant ontologies in a reuse process and a methodological approach to reusing existing knowledge from existing ontologies to build the core of the desired ontology for a domain specified via keywords. Furthermore, the results provided by our approach could be extended using any of the related ontologies to enhance the resulting ontology.

3. KNIT framework for semi-automated ontology building

KNIT follows the OWL-DL semantic syntax to define new ontologies. Table 1 summarises the OWL-DL syntax, which includes various types of operators, restrictions, and axioms to represent the concepts and relationships within a domain. Thus, *KNIT* has a rich syntax that provides for different types of operators, restrictions, and axioms to represent the concepts and relationships within a domain. These features allow *KNIT* to define complex knowledge formally and logically, making it helpful in developing ontologies with the characteristics of knowledge representation, semantic web development, and automated reasoning.

KNIT takes a list of keywords as input and produces an OWL ontology importing the reused artefacts (concepts and properties) as a result. *KNIT* framework for ontology construction consists of four steps for building a knowledge graph that will be translated to OWL:

[3] https://ontobee.org/

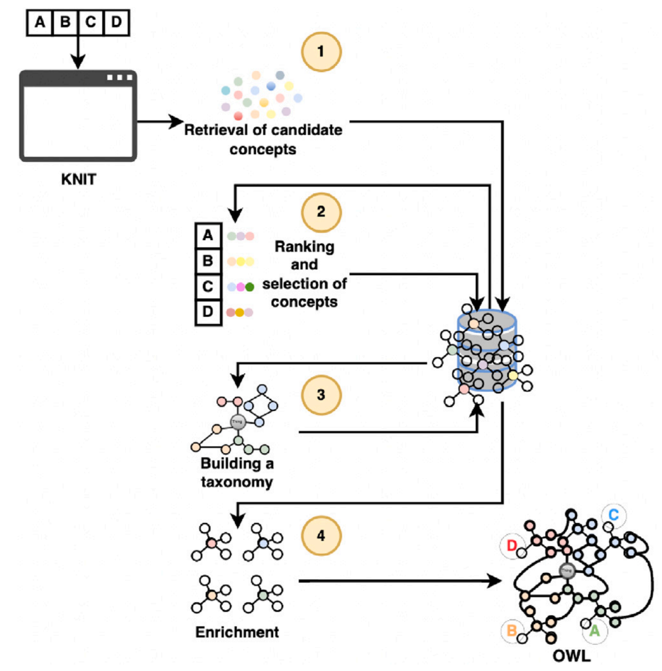

A B C D
KNIT
1
Retrieval of candidate concepts
2
A
B
C
D
Ranking and selection of concepts
3
Building a taxonomy
4
Enrichment
OWL
D
C
B
A

Fig. 1. Structure of *KNIT* algorithm. There are described the four steps of the algorithm — first, retrieval of candidate concepts. Second, ranking and selecting concepts. Later, building a taxonomy and finally enrichment and creating the owl file.

1. Retrieval of candidate concepts;
2. Selection of concepts;
3. Taxonomy building;
4. Retrieval of other artefacts/ enrichment step.

An overview of the process is represented in Fig. 1. *KNIT* takes the list of keywords or search terms (so, a multi-query specified by the user) that are to be matched 1:0..1 to concepts from all source ontologies of an *ontology repository*. The output ontology is built around these concepts. To convey an intuition of our approach, we assume that the ontology repository allows for four types of operations: (a) retrieving a ranked list of *matching*[4] concepts and their provenance, i.e., their source ontology, given a single input term; (b) retrieving all parent concepts of a given concept in a specific source ontology; (c) retrieving equivalent concepts across all source ontologies given a specific concept (d) retrieving all properties of a given concept in a specific source ontology.

Step 1: Retrieval of candidate concepts. For the first step, we carry out operation (a) in the ontology repository for each search term. For each term, we record only the first candidate concept, ranked by similarity, from each of the source ontologies, thus obtaining candidate concepts (Fig. 2).

Step 2: Selection of concepts. To finalise the mapping from input terms to concepts, we rerank candidate concepts according to the coverage of their source ontologies in descending order (Fig. 3). We define the coverage as the number of candidate concepts retrieved from a single source ontology. We then assign a top candidate to each of the input terms with a non-empty candidate list. If there is a tied ranking, we assign priority to a concept with more properties in its source ontology.

[4] We define the notion *matching* as a black box – i.e., we rely on a third party ontology repository to provide a functionality of ranking concepts, across all the source ontologies reflected in the repository, based on a textual query supplied by the user.

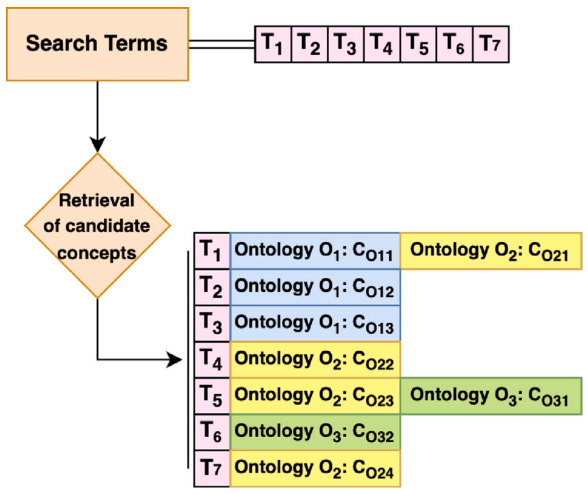

Search Terms
T_1 T_2 T_3 T_4 T_5 T_6 T_7
Retrieval of candidate concepts
T_1 Ontology O_1: C_{O11} Ontology O_2: C_{O21}
T_2 Ontology O_1: C_{O12}
T_3 Ontology O_1: C_{O13}
T_4 Ontology O_2: C_{O22}
T_5 Ontology O_2: C_{O23} Ontology O_3: C_{O31}
T_6 Ontology O_3: C_{O32}
T_7 Ontology O_2: C_{O24}

Fig. 2. Retrieval of candidate concepts. In this step, *KNIT* fetches the terms in the ontology repository and returns a set of tuples of terms and ontologies ranked by similarity.

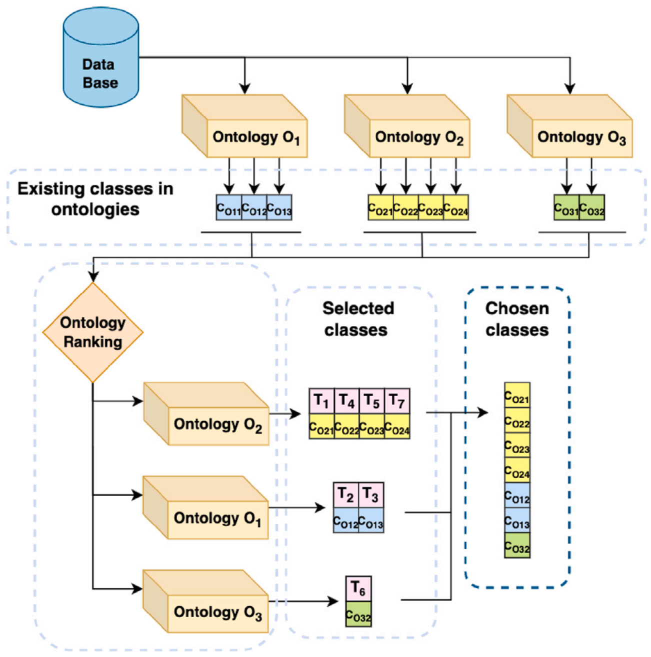


Fig. 3. Selection of concepts. This step assigns a candidate concept to each term. The candidate is selected according to the coverage of its source ontologies.

Step 3: Taxonomy building. To convert the selected concept set into a taxonomy, we recursively retrieve parent concepts from the respective source ontologies for each of the selected concepts. The retrieved parent nodes and corresponding taxonomy relationships are included in our structure. The process iterates through selected candidates from step 2 in descending rank order. It retrieves parent concepts until one of three stopping conditions are met: a root, Thing node, is reached; a newly retrieved parent node has an equivalent node among other selected nodes and already retrieved parent nodes, via operation (c); a candidate node has a parent node among already retrieved parent nodes or selected candidates, via operation (b). The process is illustrated in Fig. 4.

Step 4: Retrieval of other artefacts. For each of the nodes of the taxonomy created in the previous step, we retrieve properties from the source ontologies for data properties and instances. This process is

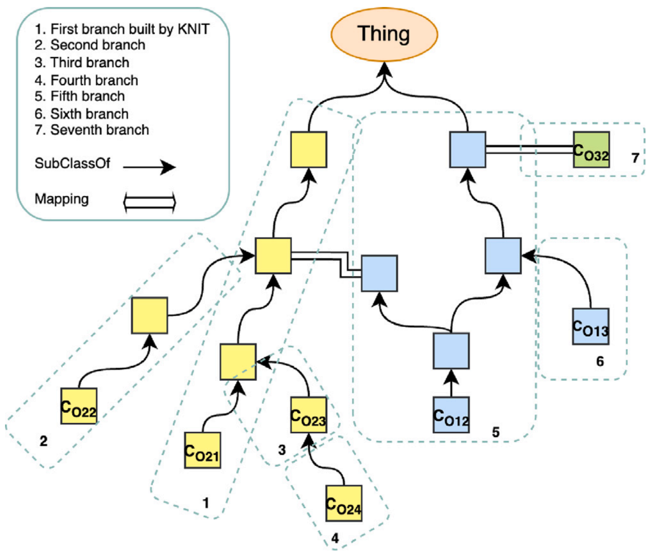


Fig. 4. Taxonomy building. The third step of *KNIT* transforms the concept set in a taxonomy, creating a graph with nodes and relationships between them. Each number in the branch means the order of creation of the graph.

straightforward, as they are transferred directly into the output ontology. For object properties, we import the 'target' concepts into the output ontology as stubs. Specifically, we do not recursively import the properties of these stub concepts — we keep them as boundary items in the proposed output ontology.

3.1. Formal description of the algorithm

More formally the algorithm can be represented as follows. We denote the multi-query supplied by the user as $Q = \{t_1, t_2, \dots, t_n\}$, where t_i represents an individual search-term. For simplicity, in the formal definition of the algorithm, we represent the repository of ontologies as a single directed (multi-)graph $G = \{V, E\}$, where V denotes a collection of the nodes of the graph that is to say, concepts of the ontologies, and E denotes the collection of edges between them. Vertices and edges of G are all vertices and edges of ontologies $O_1, O_2, \dots, O_m$. Moreover, we denote individual concepts as $C_{i,j}$, so that the notation conveys the provenance of the node by tying it to a specific ontology represented in the repository G specifically, $C_{i,j}$ denotes jth concept of the ith ontology. Individual edges of E are denoted as $e_{l,C_{i,j},C_{k,l}}$, where l is the identifier of the edge type, property URI, $C_{i,j}$ is the source node and $C_{k,l}$ is the target node.

The operation of retrieving a ranked list of matching nodes (i) can be defined via a scoring function (1), where *string[]* is used to denote a set of all possible user-supplied keyword queries.

The set of retrieved candidate concepts for a term t_k can be therefore defined as function (2), where ϵ denotes a minimal strength of a match relationship (e.g., $\epsilon = 0$). Set of all candidates S is then defined as function (3).

Coverage, used to rank per-search-term candidates, can be then defined formally as function (4).

Let $d^o_{C_{i,j}}$ denote out-degree of $C_{i,j}$ in G. Selection of candidate concepts can be defined as a function H over the sets of candidate concepts S_{t_k}: function (5)

We denote a set of selected concepts as function (6). We note that theoretically, H does not provide a 1:0,1 mapping from search-terms to concepts; however, the approach has been effective to this end in our empirical evaluations, i.e. no additional tie-breaker mechanisms were needed in examples presented further in this work.

$$\rho(t_k, C_{i,j}) : (string[], V) \mapsto \mathbb{R}_{\geq 0} \tag{1}$$

$$S_{t_k} = \{C_{i,j} : \forall_{i,j \wedge j \neq l} \rho(t_k, C_{i,j}) > \rho(t_k, C_{i,l}) \wedge \rho(t_k, C_{i,j}) > \epsilon\} \tag{2}$$

$$S = S_{t_1} \bigcup S_{t_2} \bigcup \dots \bigcup S_{t_n}. \tag{3}$$

$$h(t_k, C_{i,j}) = |C_{k,l} : \forall_{l \wedge k = i} C_{k,l} \in S| \tag{4}$$

$$H(S_{t_k}) = C_{i,j} : C_{i,j} \in S_{t_k} \wedge \forall_{C_{l,p} \in S_{t_k}} h(C_{i,j}) \geq h(C_{l,p}) \wedge$$

$$\nexists Cl, p \in S_{t_k} \setminus C_{i,j}(d^o_{C_{i,j}} \leq d^o_{C_{l,p}} \wedge h(C_{l,p}) = h(C_{i,j})) \tag{5}$$

$$N = H(S_{t_1}) \bigcup H(S_{t_2}) \bigcup \dots \bigcup H(S_{t_n}) \tag{6}$$

```
# N - set of selected 'seed' concepts C
# T - taxonomy in construction
# G - ontology repository
# C - set of concepts

def build_taxonomy_branch(C, G, T):
  if C in T:
    return
  T.vertices.add(C)
  equivalence_found=False
  for v in G.get_equivalent_classes(C):
    if v in T:
      T.edges.add(C, v, owl.equivalence)
      equivalence_found=True
  if equivalence_found:
    return
  for p in G.get_superclasses(C):
    build_taxonomy_branch(p, G, T)
    T.edges.add(p, C, owl.subclass)

T=Graph(V={owl.Thing})
for C in N:
  build_taxonomy_branch(C, G, T)
```

Listing 1: Pseudocode illustrating taxonomy creation from selected concepts. For each concept in the set of selected concepts, *KNIT* searches equivalented concepts in the ontology repository and create a relationship between them.

The process of building a graph T i.e., taxonomy building step, is represented in Fig. 4, but it can be described more formally with the pseudocode shown in Listing 1.

The retrieval of other artefacts, i.e., properties and instances, is then carried out for each concept of the taxonomy T created in the previous step. It results in a creation of a draft ontology O and can be presented as shown in Listing 2. The resulting graph structure, O is then returned to the user in OWL format.

```
# T - taxonomy is constructed in the previous step
# O - ontology in construction
# G - ontology repository

edge_types={owl.ObjectProperty, owl.DataProperty,
     owl.Annotation,
            owl.Disjoint, owl.Type, owl.Equivalent}
O = copy(T)

for C in T.concepts:
  edges=G.get_properties(C)
  for e in edges:
    if e.type in edge_types:
      if e.target not in O:
        O.vertices.add(e.target)
    O.edges.add(e)
```

Listing 2: Pseudocode illustrating taxonomy enrichment of the taxonomy with properties from the ontology repository. For each concept, *KNIT* fetches the instances, data and object properties from the ontology repository. Then, all the elements found are included in the new ontology.

4. Use cases: Knowledge graph exploration

4.1. Implementation of the KNIT system

In order to show the use of *KNIT* in real scenarios, we have developed several use cases taking advantage of a set of existing ontology repositories in the Life Sciences: the *Bioportal* ecosystem. BioPortal[5] is an open-source repository for biological ontologies in OWL, RDF, and OBO. Ontologies can be browsed, searched for, and viewed. Besides, Bioportal allows users to annotate ontology terms, map terms to one another, and review ontologies based on elements like usability, domain coverage, content quality, documentation, and support (Noy et al., 2009). The BioPortal ecosystem covers the following databases[6]:

- *BioPortal*: Repository of biomedical ontologies with 1.018 ontologies, 14.740.276 classes and a total of 79.636.946 properties.
- *AgroPortal LIRMM*: Repository of semantic resources for the agriculture domain with 147 ontologies, 938.638 classes, 2.963.171 individuals and more than 323 users.
- *BioPortal SIFR*: Repository of French biomedical ontologies with 40 ontologies, 888.324 classes, 563.312 properties, and more than 160 users.
- *EcoPortal*: Repository created by LifeWatch ERIC of semantic resources for the ecological domain with 25 ontologies and more than 815 classes.
- *MedPortal*: Repository of semantic resources for the medical domain with 54 ontologies and more than 2.000.000 classes.

The *KNIT* methodology has been adapted to the particularities of these databases, including a final process responsible for converting the resulting graph to *OWL* format, thus facilitating its study and reuse by the life sciences research community.

In this case, BioPortal is accessed using the provided *REST-API*.[7] The *KNIT* tool takes advantage of the functionalities provided by the *BioPortal* ecosystem: keyword search and ontology mapping retrieval. Then, specific information related to a given ontology is retrieved using *SPARQL* queries to the particular *SPARQL Endpoint* of each database.

With the information collected from the *REST-API* and the *SPARQL Endpoints* of the different databases of the *BioPortal* ecosystem, our tool imports the data through massive *queries* to *Neo4j*, as presented in the Cypher queries *listing 3, listing 4, listing 5, listing 6* and *listing 7*. The MATCH clause in cypher lets users describe the patterns that the database should look for. Besides, MERGE allows new data to be created and bound in the graph.[8]

```
MERGE (a {name:'name',label:'label', uri:'uri',synonym:'[
    synonym]', definition:'definition', ontology: '
    ontology'})
SET a:ontology acronym
RETURN a
```

Listing 3: Cypher query that adds a new node in a graph given a name, a label, a URI, a set of synonyms and the graph itself, which represents an ontology.

[5] http://bioportal.bioontology.org

[6] Statistics as of November 2022.

[7] http://data.bioontology.org/documentation

[8] https://neo4j.com/developer/cypher/

```
MATCH (a {uri:'uri'}),(b)
MERGE(a)-[:PROPERTY {{uri:"http://www.w3.org
    /1999/02/22-rdf-syntax-ns#type"}}]->(b)
RETURN a,b
```

Listing 4: A query that assigns each node its type where b represents the type of the node.

```
MATCH (a {{uri: 'uri'}),(b {uri:'uri parent'})
MERGE (a)-[:SCO {uri: 'http://www.w3.org/2000/01/rdf-
    schema#subClassOf'}}]->(b)
RETURN a,b
```

Listing 5: Cypher query whose goal is to create the relationships subClassOf between two classes.

```
MATCH (a {uri: 'uri1'}),(b {uri:'uri2'})
MERGE (a)-[:mapping {uri:'http://www.w3.org/2002/07/
    owl#equivalentClass'}]->(b)-[:mapping {uri:'http
    ://www.w3.org/2002/07/owl#equivalentClass'}]->(a)
RETURN a,b
```

Listing 6: Cypher query that creates the relationship equivalentClass between two classes.

```
MATCH (a {uri:'uri1'}),(b {uri:'uri2'})
MERGE (a)-[:PROPERTY {uri:'uri property', label: 'label
    property', type:' type property'}]->(b)
RETURN a,b
```

Listing 7: Cypher query, whose aim is to create custom relationships between two classes.

Once the creation of the graph is finished, *Neo4j* allows studying the ontologies to which the different nodes belong and how the other ontologies are aligned. After incorporating the graph into *Neo4j*, the tool converts the graph to *OWL* format so that it is easier to study, thus completing the workflow of the tool.

It is worth noting that the *KNIT* tool does not limit the number of maximum keywords in the list of search terms as *KNIT* groups the keywords to search by batches of terms.

4.2. Use cases

This section illustrates realistic use cases for the design of ontologies based on the reuse of existing knowledge. Three use cases made from actual research data are presented below. These case studies have been developed using *KNIT* in conjunction with the *BioPortal* database. As far as there is no reference ontology, these cases have been evaluated by an expert to analyse the usefulness of the produced solutions.

4.2.1. Use case 1: Genes and diseases

This use case is based on the work of Singh, Mobeen, Chandra, Joshi, and Ramachandran (2021), which addresses the issue of diseases that aggravate the adverse effects of SARS-CoV 19 and the genes that contribute to the development of these diseases. Thus we aim to design an ontology in the domain of this research work, related to proteins, genes and diseases. Therefore the keywords that *KNIT* will use are *MRPS25, MRPS27, SRP72, FBLN5, FBN1, FBN2, Leigh syndrome, Kearns–Sayre syndrome, Williams syndrome, Marfan syndrome, congenital contractural arachnodactyly* and *acute myeloid leukaemia*.

The *retrieval of candidate concepts* step used the BioPortal API using the input terms obtaining classes from the following ontologies: National Cancer Institute Thesaurus (NCIT) (Kumar & Smith, 2005),

Table 2
Candidate concepts obtained from the different ontologies for the creation of use case 1.

Keywords	Data found
SRP72	**Found class**: SIGNAL RECOGNITION PARTICLE, 72-KD **Synonymous**: SRP72 **Uri**: http://purl.bioontology.org/ontology/OMIM/602122 **Ontology**: OMIM
FBLN5	**Found class**: FBLN5 **Synonymous**: FBLN5, ADCL2, EVEC, DANCE, FIBL-5, UP50, ARMD3, ARCL1A **URI**: http://identifiers.org/ncbigene/10516 **Ontology**: GEXO
Williams syndrome	**Found class**: Williams Syndrome **Synonymous**: Williams Syndrome, Williams–Beuren Syndrome (WBS), Williams–Beuren Syndrome, Williams syndrome **URI**: http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl#C85232 **Ontology**: NCIT
FBN2	**Found class**: FBN2 gene **Synonymous**: FBN2 Gene, FBN2, Fibrillin 2 Gene **URI**: http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl#C120566 **Ontology**: NCIT

Online Mendelian Inheritance in Man (OMIM) (Hamosh, Scott, Amberger, Valle, & McKusick, 2000), Gene Expression Ontology (GEXO),[9] and SNOMED-CT.[10]

As previously explained, the knowledge structure is generated from the input data (keywords or search terms). An example is presented in Table 2, where we can observe the search terms and what is retrieved by the tool. It is interesting to note the case of the *SPR72* keyword, for which our tool has designated the *Signal Recognition* concept as the correct mapping since this concept has a synonym in its ontology that matches our search term.

The *ranking and selection of concepts* step provided a ranking of these ontologies based on the number of input terms represented in them:

1. NCIT [7 terms: fbn1, fbn2, leigh syndrome, Kearns–Sayre syndrome, williams syndrome, marfan syndrome, congenital contractural arachnodactyly].
2. GEXO [6 terms: mrps25, mrps27, fbln5, fbn1, fbn2, leigh syndrome].
3. SNOMEDCT [4 terms: Kearns–Sayre syndrome, williams syndrome, congenital contractural, arachnodactyly, acute myeloid leukaemia].
4. OMIM [4 terms: srp72, leigh syndrome, Kearns–Sayre syndrome, marfan syndrome].

The *building a taxonomy* step produced a Neo4j graph with the retrieved concepts (see Fig. 5) and their sub-taxonomies in the reused ontologies (see Fig. 6).

During the *enrichment* step, this Neo4j graph is extended with additional information from the reused ontologies. This step produces the final representation of the resulting graph in the Neo4j database, as seen in Fig. 7.

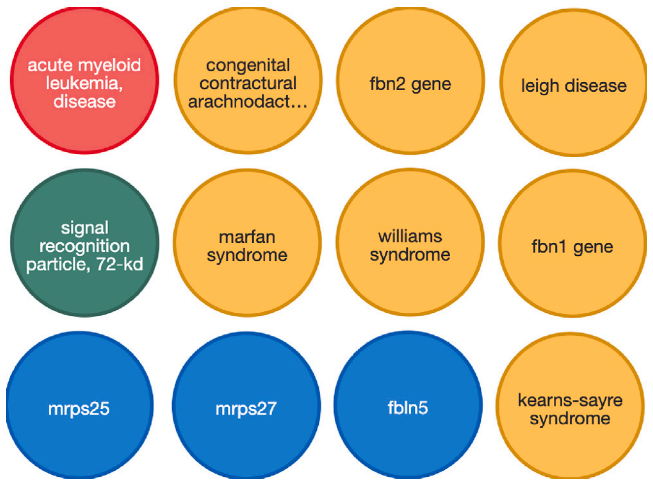

acute myeloid leukemia, disease
congenital contractural arachnodact…
fbn2 gene
leigh disease
signal recognition particle, 72-kd
marfan syndrome
williams syndrome
fbn1 gene
mrps25
mrps27
fbln5
kearns-sayre syndrome

Fig. 5. Use Case 1: Ontology classes. The found classes are from the following ontologies: NCIT (Yellow), GEXO (Blue), OMIM (Green) and SNOMEDT (Red).

As stated, the resulting graph is translated to *OWL* in a final step. This resulting ontology takes advantage of the alignments to interrelate the different branches. In Fig. 8 we see a representation of the new ontology generated by the *Protégé* software. This result is relevant for researchers working in this domain. However, the expert evaluation highlights some minor aspects of repeating the "protein" concept extracted from different ontologies. This issue could be solved by aligning the reused ontologies to discover the similarity of concepts.

4.2.2. Case 2: Viroma ontology

The second use case reuses the dataset introduced by Gregory et al. (2020). As the authors explain, gut microbes profoundly affect humans, but the viruses that affect them are often overlooked due to the limitations of the reference database. To resolve this situation, the researchers developed a human intestinal virome (GVD) database from 2697 viral particles or microbial metagenomes from 1986 people representing 16 countries.

The *retrieval of candidate concepts* step used the BioPortal API using the input terms obtaining classes from the following ontologies: Medical Subject Headings (MESH),[11] National Cancer Institute Thesaurus (NCIT) (Kumar & Smith, 2005), Biological and Environmental Research Ontology (BERO),[12] National Center for Biotechnology Information (NCBI) Organismal Classification (NCBITAXON)[13] and Interlinking Ontology for Biological Concepts (IOBC) (Kozaki, Kushida, Yamamoto, & Takagi, 2019).

The *ranking and selection of concepts* step provided a ranking of these ontologies based on the number of input terms represented in them:

1. MESH [22 terms: adenoviridae, anelloviridae, asfarviridae, astroviridae, caliciviridae, circoviridae, geminiviridae, lipothrixviridae, herpesviridae, inoviridae, iridoviridae, microviridae, myoviridae, papillomaviridae, parvoviridae, picornaviridae, podoviridae, polyomaviridae, poxviridae, rudiviridae, siphoviridae, archaeal virus].
2. BERO [18 terms: alphaflexiviridae, anelloviridae, asfarviridae, bicaudaviridae, circoviridae, geminiviridae, inoviridae, iridoviridae, lipothrixviridae, microviridae, myoviridae, papillomaviridae, podoviridae, polyomaviridae, rudiviridae, siphoviridae, virgaviridae, archaeal virus].

[9] https://bioportal.bioontology.org/ontologies/GEXO
[10] https://bioportal.bioontology.org/ontologies/SNOMEDCT
[11] https://bioportal.bioontology.org/ontologies/MESH
[12] https://bioportal.bioontology.org/ontologies/BERO
[13] https://bioportal.bioontology.org/ontologies/NCBITAXON

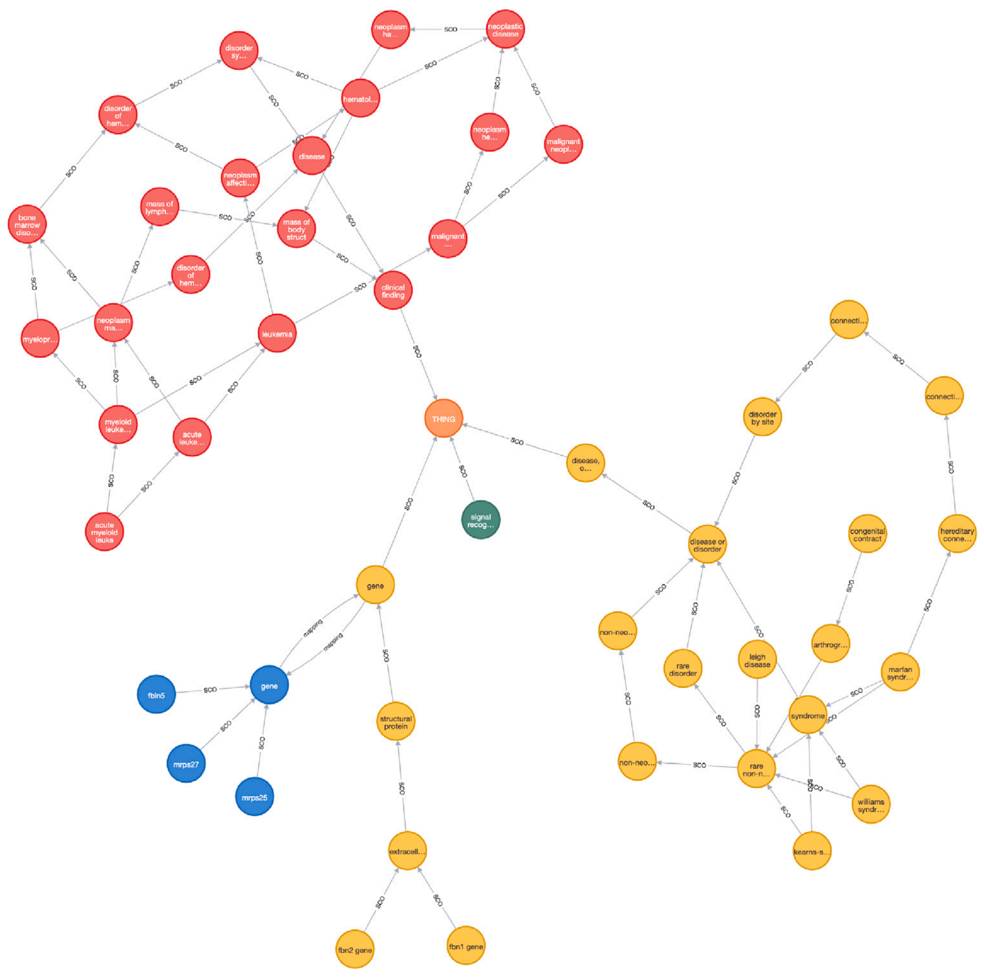


Fig. 6. Use Case 1: Taxonomy graph. NCIT in yellow, GEXO in blue, OMIM in green, SNOMEDT in Red and THING in orange.

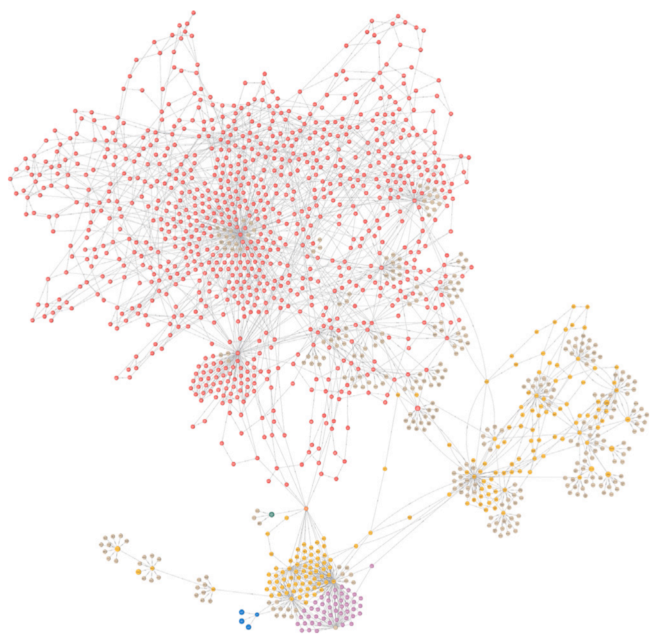


Fig. 7. Use Case 1: Enrichment. The hierarchy is extended with data and object properties from the source ontologies.

3. NCIT [17 terms: study, individual, type, age, western, healthy, base pair, adenoviridae, astroviridae, caliciviridae, herpesviridae, parvoviridae, picornaviridae, poxviridae, unassigned, bacteriophage].
4. NCBITAXON [16 terms: caliciviridae, cruliviridae, genomoviridae, herpesviridae, inoviridae, iridoviridae, lipothrixviridae, microviridae, myoviridae, papillomaviridae, parvoviridae, poxviridae, rudiviridae, siphoviridae, smacoviridae, virgaviridae].
5. IOBC [16 terms: multiple displacement amplification, base pair, anelloviridae, lipothrixviridae, asfarviridae, circoviridae, geminiviridae, inoviridae, iridoviridae, microviridae, myoviridae, papillomaviridae, podoviridae, polyomaviridae, rudiviridae, siphoviridae]

A sample of the term relationships generated by *KNIT* can be seen in Table 3.

The *building a taxonomy* step produced a Neo4j graph with the retrieved concepts (see Fig. 9) and their taxonomies in the reused ontologies (see Fig. 10).

During the *enrichment* step this Neo4j graph is extended with additional information from the reused ontologies. This step produces the final representation of the resulting graph in the Neo4j database as can be seen in Fig. 11.

In Fig. 12 we can see a representation generated by the Protégé software of the final ontology. The resulting ontology includes concepts that cover the target terms used in the search, including a reasonable structure connecting them, thanks to the limited number of ontologies used in the reuse process.

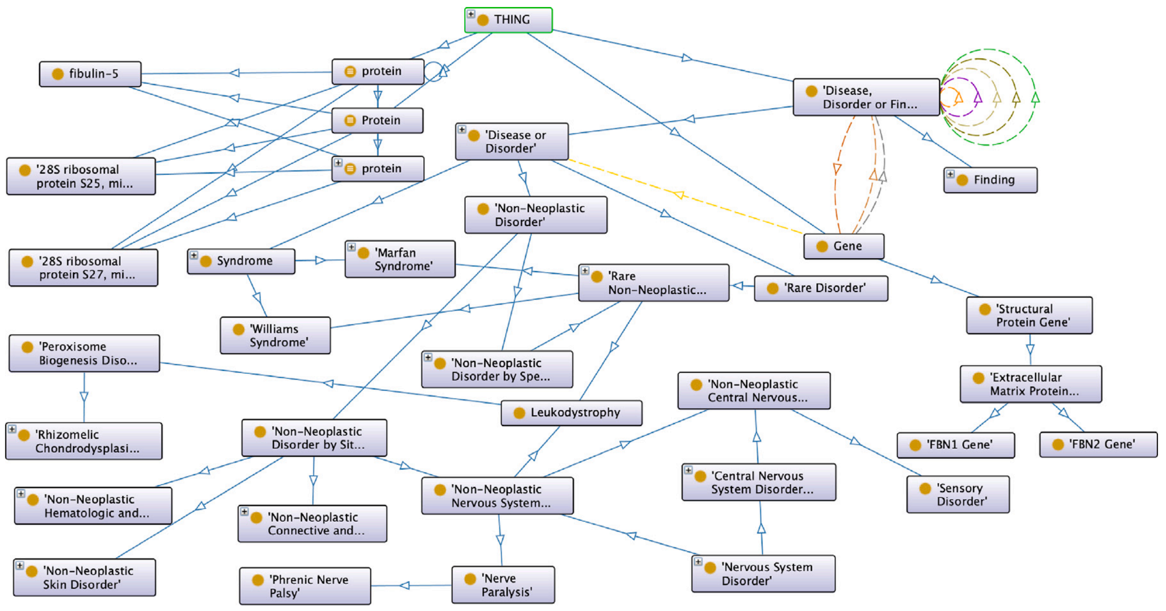

THING
protein
Protein
protein
fibulin-5
'28S ribosomal protein S25, mi...
'28S ribosomal protein S27, mi...
'Disease, Disorder or Fin...
'Disease or Disorder'
Finding
Gene
'Non-Neoplastic Disorder'
Syndrome
'Marfan Syndrome'
'Rare Non-Neoplastic...
'Rare Disorder'
'Structural Protein Gene'
'Williams Syndrome'
'Peroxisome Biogenesis Diso...
'Non-Neoplastic Disorder by Spe...
'Extracellular Matrix Protein...
'Non-Neoplastic Central Nervous...
Leukodystrophy
'Rhizomelic Chondrodysplasi...
'Non-Neoplastic Disorder by Sit...
'FBN1 Gene'
'FBN2 Gene'
'Central Nervous System Disorder...
'Sensory Disorder'
'Non-Neoplastic Nervous System...
'Non-Neoplastic Hematologic and...
'Non-Neoplastic Connective and...
'Non-Neoplastic Skin Disorder'
'Phrenic Nerve Palsy'
'Nerve Paralysis'
'Nervous System Disorder'

Fig. 8. Use Case 1: OWL Ontology. Base schema of the ontology developed automatically.

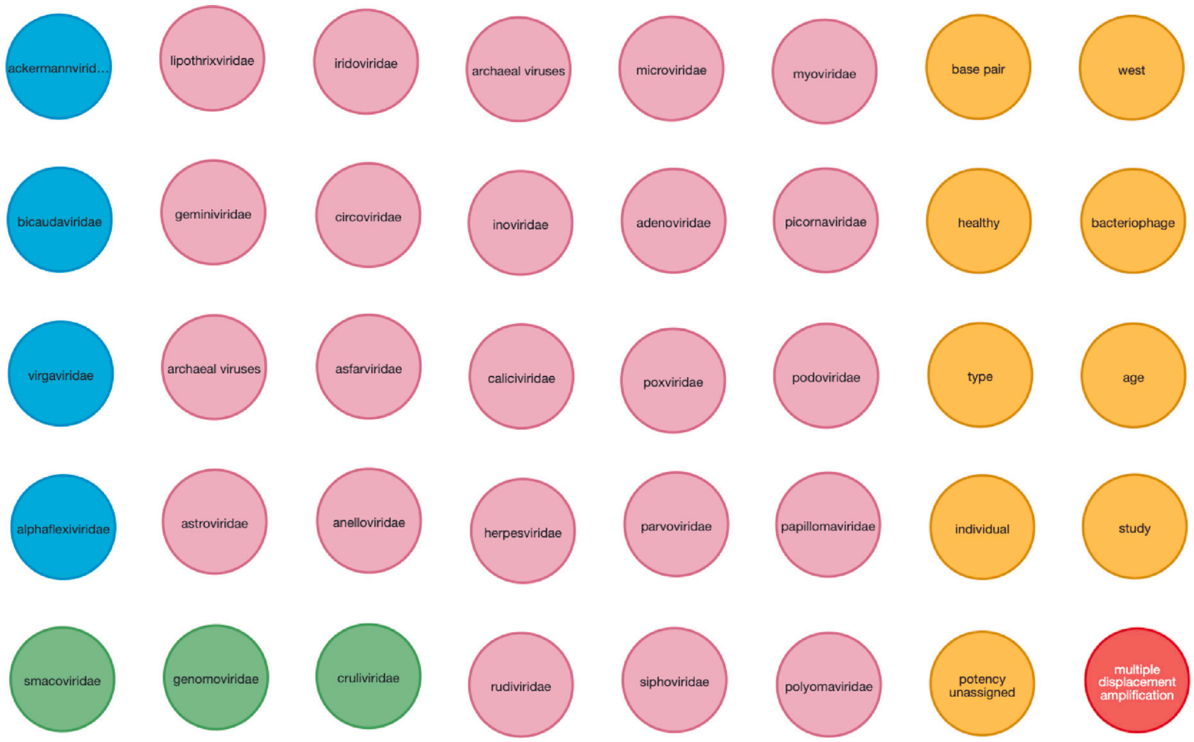


Fig. 9. Use Case 2: Ontology classes. The found classes are from the following ontologies: MESH (pink), NCIT (Yellow), BERO (Blue), NCBITAXON (Green) and IOBC (Red).

4.2.3. *Case 3: Ontological medicines*

The third use case is part of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching.[14] For this use case, the dataset name in the challenge[15] has been selected. Using this dataSet, the *KNIT* tool characterises and represents the keywords: *Medicine, ATC Code, Route of administration.*

The *retrieval of candidate concepts* step used the BioPortal API using the input terms obtaining classes from the following ontologies: National Cancer Institute Thesaurus (NCIT) (Kumar & Smith, 2005), Bioscientific data analysis ontology (EDAM) (Black et al., 2021). Given the necessary knowledge from the keywords we obtain the Table 4.

The *ranking and selection of concepts* step provided a ranking of these ontologies based on the number of input terms represented in them:

1. NCIT [2 terms: medicine, route of administration]
2. EDAM [2 terms: medicine, atc code]

The *building a taxonomy* step produced a Neo4j graph with the retrieved concepts (see Fig. 13) and their taxonomies in the reused ontologies (see Fig. 14).

[14] https://www.cs.ox.ac.uk/isg/challenges/sem-tab/2021/index.html
[15] TOUGH_WEB_MISSP_drugs.csv

Table 3
Candidate concepts obtained from the different ontologies for the creation of use case 2.

Keywords	Data found
Base pair	**Found class**: Base Pair **Synonymous**: base pair, Base Pair **URI**: http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl#C94634 **Ontology**: NCIT
Bicaudaviridae	**Found class**: Bicaudaviridae **Synonymous**: archaeal virus, prokaryotic virus **URI**: http://purl.obolibrary.org/obo/NCBITaxon_423358 **Ontology**: BERO
Caliciviridae	**Found class**: Caliciviridae **Synonymous**: Nebovirus **URI**: http://purl.bioontology.org/ontology/MESH/D002139 **Ontology**: MESH
Multiple displacement amplification	**Found class**: multiple displacement amplification **Synonymous** MDA **URI**: http://purl.jp/bio/4/id/200906031228917655 **Ontology**: IOBC

Table 4
Candidate concepts obtained from the different ontologies for the creation of use case 3.

Keywords	Data found
Route of administration	**Found class**: Route of Administration **Synonymous**: ROUTE DETAIL, Route of Drug Administration, route of administration (ROA), ROUTE, Drug Route of Administration **URI**: http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl#C38114 **Ontology**: NCIT
atc code	**Found class**: ATC code **URI**: http://edamontology.org/data_3103 **Ontology**: EDAM
Medicine	**Found class**: Medicine **Synonymous**: Medicine, medicine **URI**: http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl#C1683 **Ontology**: NCIT

During the *enrichment* step this Neo4j graph is extended with additional information from the reused ontologies. This step produces the final representation of the resulting graph in the Neo4j database as can be seen in Fig. 15. In Fig. 16 we can see a representation generated by the Protégé software of the final ontology. This ontology includes the terms used in the search process. These concepts are well connected in the ontology structure.

5. Evaluation

In the previous section, we have shown realistic examples of the usage of the proposed methodology. However, the desired ontology is unknown, making the evaluation of the algorithm subjective. This section focuses on two synthetic use cases to evaluate *KNIT*. We pick two target ontologies from BioPortal. Our choice of target ontologies is based on the availability of mappings of the target ontology to other ontologies in the repository. For each evaluation scenario, we restrict *KNIT* from using the target ontology directly by adding it to a 'black' list. We use the concept labels of the target ontology as keywords. Finally, we evaluate the algorithm on the similarity of the output ontology to the target ontology. In other words, we would want *KNIT* to re-create the target ontology, given its concept labels as inputs.

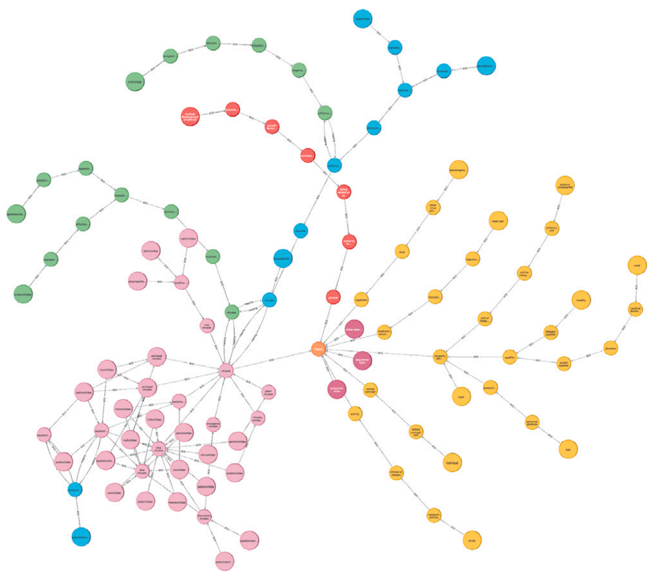


Fig. 10. Use Case 2: Taxonomy graph. MESH in pink, NCIT in yellow, BERO in blue, NCBITAXON in green, IOBC in Red and THING in orange.

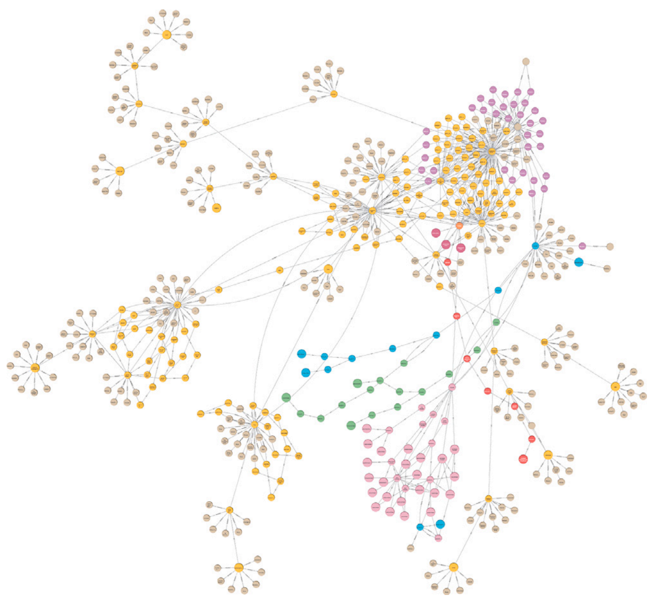


Fig. 11. Use Case 2: Enrichment. The hierarchy is extended with data and object properties from the source ontologies.

We measure the overlap between both ontologies using the Jaccard score (Essayeh & Abed, 2015; Sun, Ma, & Wang, 2015). The Jaccard's coefficient for the similarity of two sets A and B is calculated as follows:

$$Jaccard(A, B) = \frac{|A \bigcap B|}{|A \bigcup B|}$$

We can measure the distance between A and B, with values between 0 and 1 (values closer to 0 mean better results in approximating the target ontology) as follows:

$$DistanceJ(A, B) = 1 - Similarity(A, B) = \frac{|A \bigcup B| - |A \bigcap B|}{|A \bigcup B|}$$

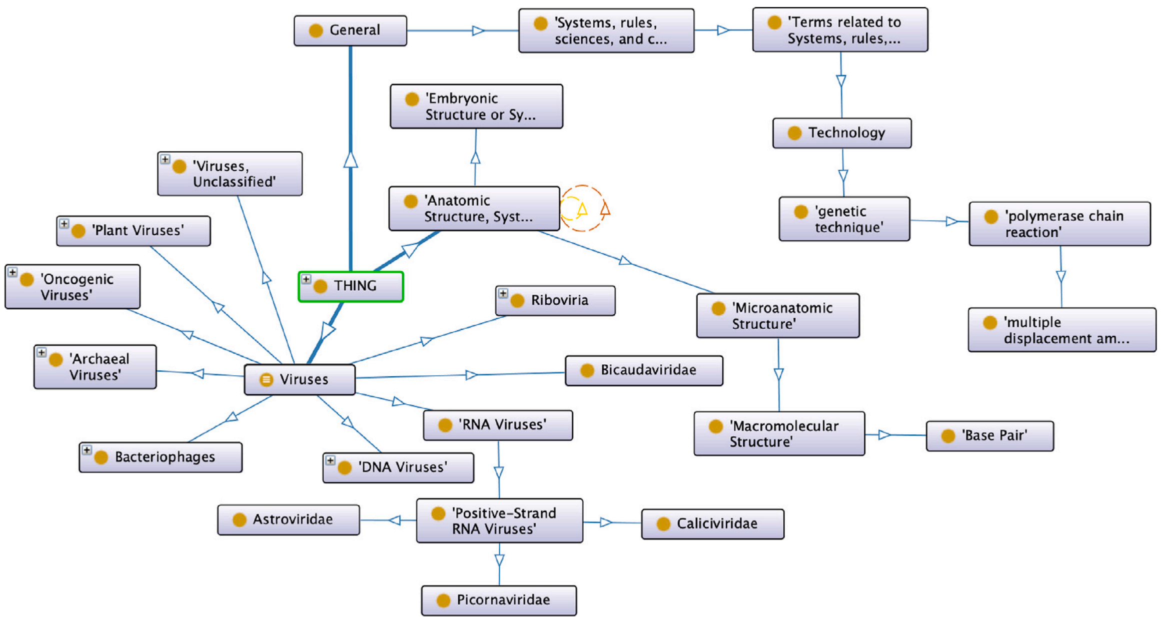

General
'Systems, rules, sciences, and c...
'Terms related to Systems, rules,...
'Embryonic Structure or Sy...
Technology
'Viruses, Unclassified'
'Anatomic Structure, Syst...
'genetic technique'
'polymerase chain reaction'
'Plant Viruses'
THING
'Oncogenic Viruses'
Riboviria
'Microanatomic Structure'
'multiple displacement am...
'Archaeal Viruses'
Viruses
Bicaudaviridae
'RNA Viruses'
'Macromolecular Structure'
'Base Pair'
Bacteriophages
'DNA Viruses'
Astroviridae
'Positive-Strand RNA Viruses'
Caliciviridae
Picornaviridae

Fig. 12. Use Case 2: OWL Ontology. Base schema of the ontology developed automatically.

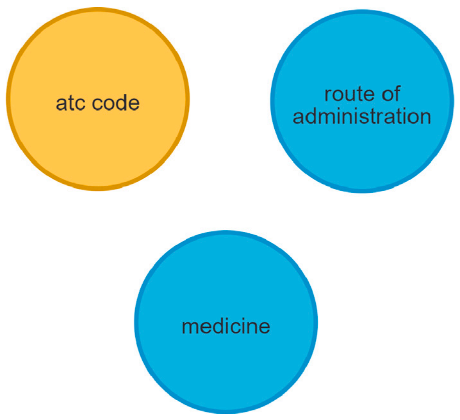


Fig. 13. Use Case 3: Ontology classes. The found classes are from the following ontologies: NCIT in blue and EDAM in yellow.

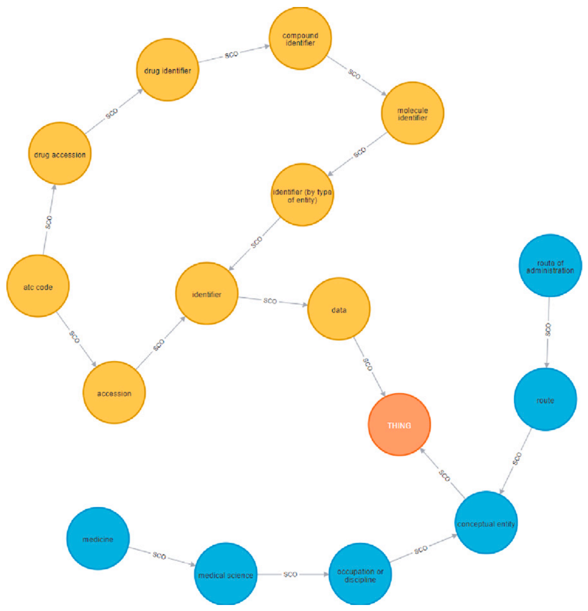


Fig. 14. Use Case 3: Taxonomy graph.NCIT in blue and EDAM in yellow.

In our case A and B are the set of Classes and Properties from the ontologies to be compared. In order to calculate the intersection, we need to know which classes or properties mean the same from a semantic point of view. This is needed as the URI of the classes and properties in different ontologies could be not the same even when they refer to the same conceptual piece of knowledge. Ontology Alignment, also known as Ontology Matching, refers to the process of identifying entities in two or more ontologies that share similarities or matches. This technique enables the determination of relationships between multiple ontologies. Various tools have been developed to facilitate Ontology Alignment. However, in this work, we have followed a manual alignment process.

For evaluations we use MINERAL[16] ontology and GFO[17] ontology. MINERAL includes mappings with more than 74 different ontologies. Meanwhile, GFO includes mappings with more than 400 different ontologies.

5.1. MINERAL

The following words have been used as Keywords: Calcite, Evaporite, Feldspar, Hematite, Magnetite, Malic, Mineraloid, NonmetallicMineral, Olvine, Pyrolite, Pyroxine, Quartz, Glass. These words have been selected from the labels of leaf classes that make up the MINERAL ontology.

As Fig. 17 shows, the resulting ontology is similar to the target one, with the resulting ontology containing additional parent concepts due to the inclusive implementation of the MIREOT principle. The concepts of the resulting ontology were all retrieved from SWEET ontology. This ontology contains concepts matching all the query terms. The Jaccard

[16] https://bioportal.bioontology.org/ontologies/MINERAL/
[17] https://bioportal.bioontology.org/ontologies/GFO

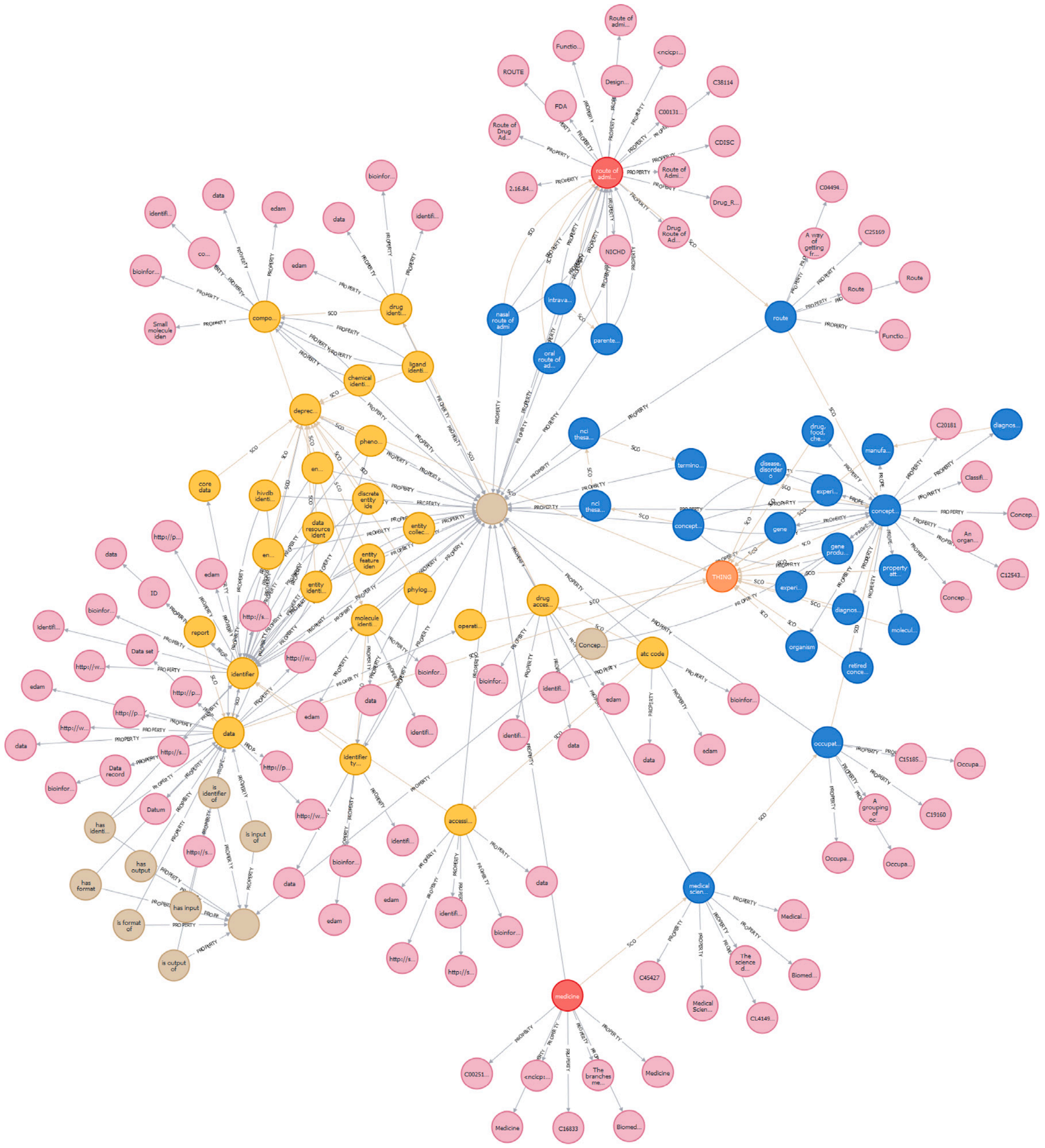


Fig. 15. Use Case 3: Enrichment. The hierarchy is extended with data and object properties from the source ontologies.

score between the result and the target is 0.84, which, we believe, represents a good approximation.

5.1.1. GFO

The following keywords have been used as inputs: Category, Concept, Symbol structure, Universal, Individual, Abstract, Concrete, Perpetuant, Presential, Amount of substrate, Material boundary, Material object, Processual Structure, Occurrent, Process, Property, Relational role, Relator, Role, Processual role, Relational role, Social role, Space time entity, Space entity, Spatial boundary, Line, Point, Surface, Spatial region, Topoid, Time entity, Temporal region, Chronoid, Time boundary, Left time boundary, Right time boundary.

The structure of the GFO ontology (target) is shown in Fig. 18. As shown in Fig. 19, all the elements have been recovered using the two different ontologies. NCIT[18] was chosen as a primary source. Since not all keywords were matched to NCIT concepts, *KNIT* filled in the gaps with concepts from a second ontology (PMD[19]). The Jaccard overlap between the target and resulting ontology is 0.55. Again, most 'errors' result from *KNIT* importing entire branches for its implementation of the MIREOT principle. Notably, 93% (40 of 43) of GFO elements are retrieved.

6. Discussion

The proposed system facilitates generating ontologies for a given design need, with a focus on reusing the existing knowledge. Due to the difficulty that this idea implies, specific design decisions were taken in the *KNIT* design; the goal of these decisions was to improve the speed, versatility and quality of the results. Example of these decisions

[18] https://bioportal.bioontology.org/ontologies/NCIT

[19] https://bioportal.bioontology.org/ontologies/PMD

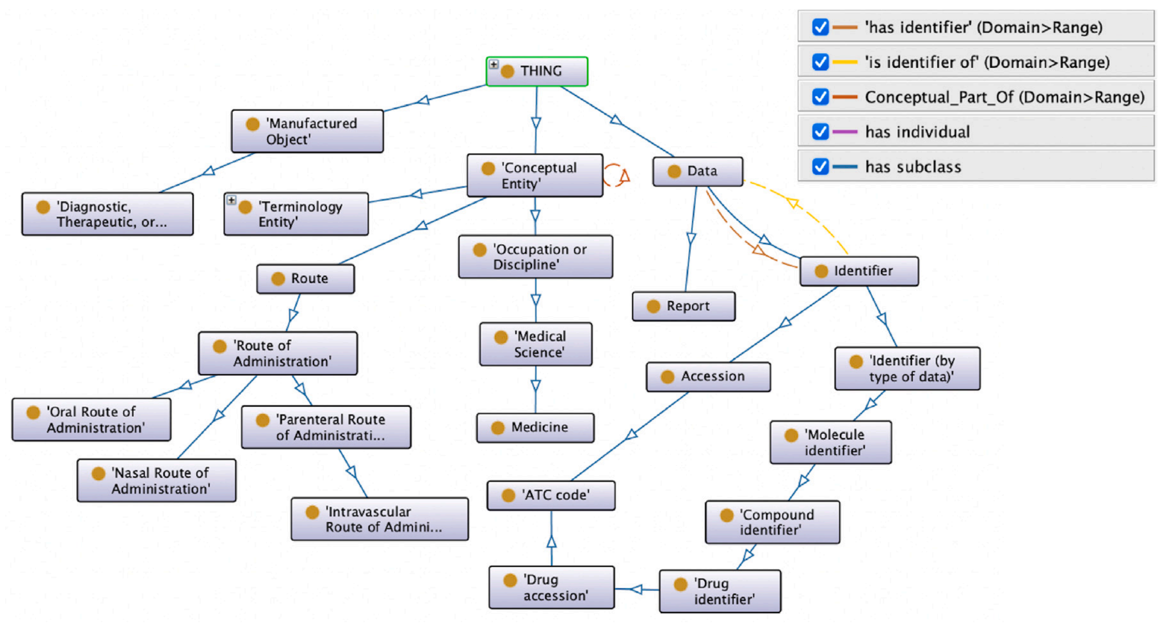
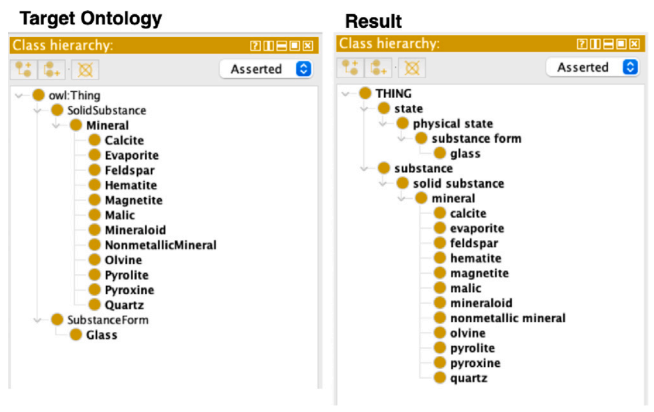

'has identifier' (Domain>Range)
'is identifier of' (Domain>Range)
Conceptual_Part_Of (Domain>Range)
has individual
has subclass
THING
'Manufactured Object'
'Conceptual Entity'
Data
'Diagnostic, Therapeutic, or...
'Terminology Entity'
'Occupation or Discipline'
Route
Report
Identifier
'Route of Administration'
'Medical Science'
Accession
'Identifier (by type of data)'
'Oral Route of Administration'
'Parenteral Route of Administrati...
Medicine
'Molecule identifier'
'Nasal Route of Administration'
'ATC code'
'Intravascular Route of Admini...
'Compound identifier'
'Drug accession'
'Drug identifier'

Fig. 16. Use Case 3: OWL Ontology. Base schema of the ontology developed automatically.


Target Ontology

Class hierarchy:

Asserted

- owl:Thing
  - SolidSubstance
    - Mineral
      - Calcite
      - Evaporite
      - Feldspar
      - Hematite
      - Magnetite
      - Malic
      - Mineraloid
      - NonmetallicMineral
      - Olvine
      - Pyrolite
      - Pyroxine
      - Quartz
  - SubstanceForm
    - Glass

Result

Class hierarchy:

Asserted

- THING
  - state
    - physical state
      - substance form
        - glass
  - substance
    - solid substance
      - mineral
        - calcite
        - evaporite
        - feldspar
        - hematite
        - magnetite
        - malic
        - mineraloid
        - nonmetallic mineral
        - olvine
        - pyrolite
        - pyroxine
        - quartz

Fig. 17. Comparative figure between the base structures of Mineral Ontology and the *KNIT* ontology created with SWEET.

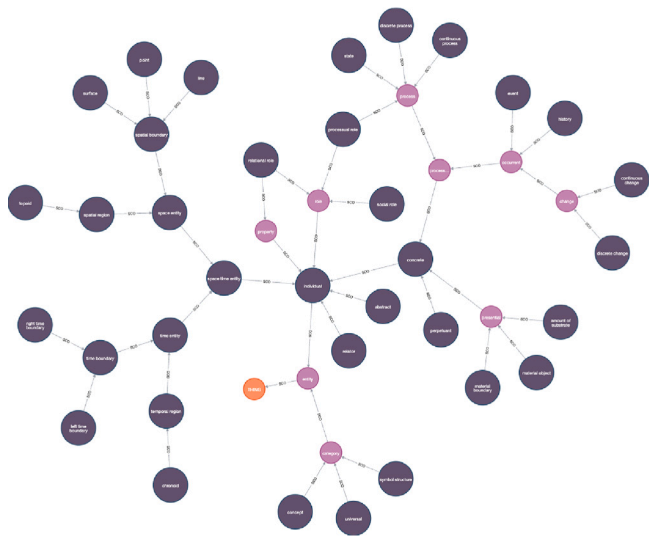


Fig. 18. Target Ontology (GFO).

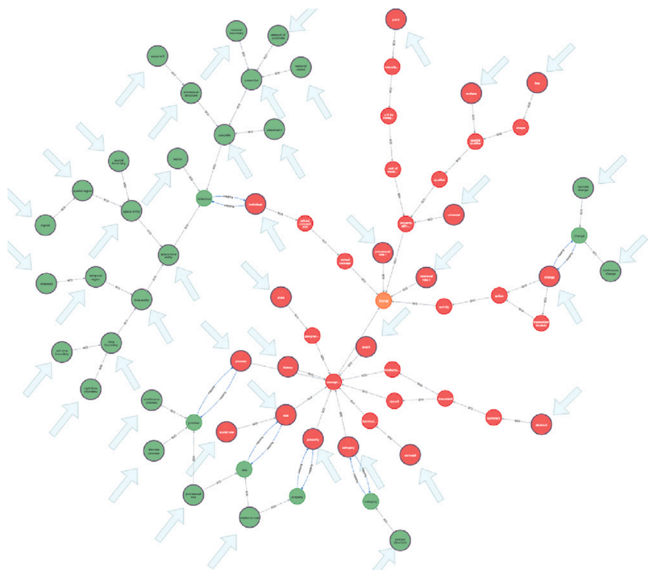


Fig. 19. Resulting ontology for the GFO evaluation; the green colour represents the PMD classes, and the red colour belongs to NCIT.

includes the strategy to retrieve the matching concept within a set of ontologies or the 'tiebreaker' in the ontology ranking process.

The case of the 'tiebreaker', a circumstance that occurs when two o more ontologies contain the same number of candidate concepts matched to keywords, is not a trivial issue. In our case, said tiebreaker is resolved by studying the 'richness' of the ontologies in terms of properties that describe the candidate concepts. Thus, in case of a 'tie', we opt to choose a concept with more information.

As shown, *KNIT* needs an ontology repository. The larger said repository is and the greater its thematic richness, the better the performance and retrieval effectiveness of *KNIT* would be. Thus, our case studies have been based on BioPortal to have a large set of interrelated ontologies in a given domain. However, *KNIT* can be applied to any ontology repository since its benefits are not based on the API but on knowledge reuse.

As can be seen in the presented use cases, *KNIT* has been able to retrieve and reconnect diverse knowledge to provide coherent results. This retrieval and reconnection are evident in the taxonomy figures of the use cases, where we can easily observe the existence of mappings between the various branches of our graphs.

However, despite the fact that our results appear to be coherent, quantifying their quality is an open problem. This paper presents a minimal evaluation using synthetic use cases, attempting to reconstruct an ontology artificially removed from the repository. While it does not amount to a complete evaluation, we believe it would make for a reasonable starting point for future research on the topic. We plan to address the issue of evaluating automated retrieval-based ontology creation more systematically in our future work.

We also note that our evaluation lacks comparison to baseline effectiveness. This is primarily because *KNIT* effectively proposes a novel workflow, so comparisons to existing tools are problematic. For example, similar results could be reached with OntoFox, which implements the MIREOT principle and is the conceptually closest existing tool. Nonetheless, reaching them with OntoFox would require the user to search for the individual concepts semi-manually, while *KNIT* automates the discovery process. On the other hand, a naive baseline of importing best matching concepts via BioPortal concept search will not result in a taxonomical structure. It will likely retrieve concepts from multiple source ontologies. The latter is an undesired characteristic known to hamper the coherence of the resulting structure (Alharbi et al., 2021).

The proposed solution takes as input a set of keywords defining the domain. This set is taken to retrieve those ontologies that are close to the provided *context*. Thus, the selection of keywords is relevant for getting good results. For example, when the number of keywords is close to one, the context cannot be properly determined due to the lack of elements for selecting candidate ontologies. We will address the challenge of correctly inferring the user's intent from limited information as a part of future work.

7. Conclusions

In this work, we have presented *KNIT*, a solution for embedding the reusability component in ontology design. Our approach provides a way of designing new ontologies from existing knowledge. *KNIT* depends on the existence of ontology repositories. The building of the output ontology is done by accessing the ontology repository to collect the terms of interest for the ontology designer. Thus, the process is oriented to reconstruct the part of a knowledge graph that could solve the application or domain needs.

As a proof of concept, we have implemented the proposed approach in the field of Life Sciences. We present case studies using the *BioPortal* repositories. Neo4j has been used as the intermediate tool for managing knowledge graph reconstruction. This project is open to other developers to contribute to the tool and leverage it in their own projects.[20]

Overall, we note that our initial hypothesis was met. Our work in this space does not end here since we are presented with new lines of research in relation to the validation of the ontologies generated by our algorithm.

It should be noted that the algorithm is able to retrieve the context starting from the user input (presented as a set of keywords) in a chosen domain. Thus, this solution could be used as part of a data characterisation process using the existing metadata as input. In this sense, our approach would support other artificial intelligence solutions for data characterisation through reusing existing knowledge. The 2021 AI Index Report published by the AI Index Steering Committee of the Human-Centered AI Institute (Stanford University) (Zhang, Mishra, et al., 2021a), exposes this problem as a vital challenge to create rich models that integrate data specific to the domain and adjacent knowledge. In this sense, our algorithm can be presented as an advanced and a powerful starting point for future studies that address ontological learning and data characterisation as effective solutions.

[20] https://github.com/ProyectoAether/KNIT

CRediT authorship contribution statement

Jorge Rodríguez-Revello: Conceptualization, Methodology, Software, Investigation, Data curation, Writing – original draft, Visualization. **Cristóbal Barba-González:** Conceptualization, Validation, Writing – review & editing. **Maciej Rybinski:** Conceptualization, Validation, Writing – review & editing. **Ismael Navas-Delgado:** Conceptualization, Validation, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

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

Data availability

We have shared the link to my data.

Acknowledgements

This work has been funded by grant PID2020-112540RB-C41[21], AETHER-UMA (A smart data holistic approach for context-aware data analytics: semantics and context exploitation). Funding for open access charge: Universidad de Málaga / CBUA.

References

Al-Aswadi, F. N., Chan, H. Y., & Gan, K. H. (2020). Automatic ontology construction from text: a review from shallow to deep learning trend. *Artificial Intelligence Review*, *53*, 3901–3928.

Al-Saleem, J., Granet, R., Ramakrishnan, S., Ciancetta, N. A., Saveson, C., Gessner, C., et al. (2021). Knowledge graph-based approaches to drug repurposing for COVID-19. *Journal of Chemical Information and Modeling*, *61*(8), 4058–4067.

Albukhitan, S., Helmy, T., & Alnazer, A. (2017). Arabic ontology learning using deep learning. In *Proceedings of the international conference on web intelligence* (pp. 1138–1142).

Alharbi, R., Tamma, V., & Grasso, F. (2021). Characterising the gap between theory and practice of ontology reuse. In *Proceedings of the 11th on knowledge capture conference* (pp. 217–224).

Allocca, C., d'Aquin, M., & Motta, E. (2009). Towards a formalization of ontology relations in the context of ontology repositories. In *International joint conference on knowledge discovery, knowledge engineering, and knowledge management* (pp. 164–176). Springer.

Arguello Casteleiro, M., Demetriou, G., Read, W., Fernandez Prieto, M. J., Maroto, N., Maseda Fernandez, D., et al. (2018). Deep learning meets ontologies: experiments to anchor the cardiovascular disease ontology in the biomedical literature. *Journal of Biomedical Semantics*, *9*(1), 1–24.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., et al. (2000). Gene ontology: tool for the unification of biology. *Nature Genetics*, *25*(1), 25–29.

Asim, M. N., Wasim, M., Khan, M. U. G., Mahmood, W., & Abbasi, H. M. (2018). A survey of ontology learning techniques and applications. *Database*, *2018*.

Beisswanger, E., Schulz, S., Stenzhorn, H., & Hahn, U. (2008). BioTop: An upper domain ontology for the life sciences. *Applied Ontology*, *3*(4), 205–212.

Black, M., Lamothe, L., Eldakroury, H., Kierkegaard, M., Priya, A., Machinda, A., et al. (2021). EDAM: The bioscientific data analysis ontology (update 2021)[version 1; not peer reviewed]. *F1000*.

Browarnik, A., & Maimon, O. (2015). Ontology learning from text: why the ontology learning layer cake is not viable. *International Journal of Signs and Semiotic Systems (IJSSS)*, *4*(2), 1–14.

[21] Funded by MCIN/AEI/10.13039/501100011033/.

Caldarola, E. G., & Rinaldi, A. M. (2016). An approach to ontology integration for ontology reuse. In *2016 IEEE 17th international conference on information reuse and integration* (pp. 384–393). IEEE.

Casteleiro, M. A., Prieto, M. J. F., Demetriou, G., Maroto, N., Read, W. J., Maseda-Fernandez, D., et al. (2016). Ontology learning with deep learning: a case study on patient safety using PubMed. In *SWAT4LS* (pp. 1–10).

Cimiano, P., Mädche, A., Staab, S., & Völker, J. (2009). Ontology learning. In *Handbook on ontologies* (pp. 245–267). Springer.

Côté, R. G., Jones, P., Martens, L., Apweiler, R., & Hermjakob, H. (2008). The ontology lookup service: more data and better tools for controlled vocabulary queries. *Nucleic Acids Research*, *36*(suppl_2), W372–W376.

Courtot, M., Gibson, F., Lister, A. L., Malone, J., Schober, D., Brinkman, R. R., et al. (2011). MIREOT: The minimum information to reference an external ontology term. *Applied Ontology*, *6*(1), 23–33.

Dahab, M. Y., Hassan, H. A., & Rafea, A. (2008). TextOntoEx: Automatic ontology construction from natural English text. *Expert Systems with Applications*, *34*(2), 1474–1480.

d'Aquin, M., & Lewen, H. (2009). Cupboard–a place to expose your ontologies to applications and the community. In *European semantic web conference* (pp. 913–918). Springer.

d'Aquin, M., & Noy, N. F. (2012). Where to publish and find ontologies? A survey of ontology libraries. *Journal of Web Semantics*, *11*, 96–111.

Ding, Z., & Peng, Y. (2004). A probabilistic extension to ontology language OWL. In *37th annual Hawaii international conference on system sciences, 2004. Proceedings of the* (pp. 10–pp). IEEE.

Doran, P., Tamma, V., & Iannone, L. (2007). Ontology module extraction for ontology reuse: an ontology engineering perspective. In *Proceedings of the sixteenth ACM conference on conference on information and knowledge management* (pp. 61–70).

Edison, O., Jie, Z., Smith, B., Yongqun, H., et al. (2016). Ontobull and bfoconvert: Web-based programs to support automatic ontology conversion. In *ICBO/BioCreative* (pp. 1–2).

Essayeh, A., & Abed, M. (2015). Towards ontology matching based system through terminological, structural and semantic level. *Procedia Computer Science*, *60*, 403–412.

Euzenat, J., Shvaiko, P., et al. (2007). *Ontology matching, vol. 18*. Springer.

Fanizzi, N., d'Amato, C., & Esposito, F. (2008). DL-FOIL concept learning in description logics. In *International conference on inductive logic programming* (pp. 107–121). Springer.

Fernández-López, M., Poveda-Villalón, M., Suárez-Figueroa, M. C., & Gómez-Pérez, A. (2019). Why are ontologies not reused across the same domain? *Journal of Web Semantics*, *57*, Article 100492.

Francis, N., Green, A., Guagliardo, P., Libkin, L., Lindaaker, T., Marsault, V., et al. (2018). Cypher: An evolving query language for property graphs. In *Proceedings of the 2018 international conference on management of data* (pp. 1433–1445).

Gil, R., & Martin-Bautista, M. J. (2014). SMOL: a systemic methodology for ontology learning from heterogeneous sources. *Journal of Intelligent Information Systems*, *42*(3), 415–455.

Gómez-Pérez, A., & Rojas-Amaya, M. D. (1999). Ontological reengineering for reuse. In *International conference on knowledge engineering and knowledge management* (pp. 139–156). Springer.

Gregory, A. C., Zablocki, O., Zayed, A. A., Howell, A., Bolduc, B., & Sullivan, M. B. (2020). The gut virome database reveals age-dependent patterns of virome diversity in the human gut. *Cell Host & Microbe*, *28*(5), 724–740.

Guarino, N. (1995). Formal ontology, conceptual analysis and knowledge representation. *International Journal of Human-Computer Studies*, *43*(5–6), 625–640.

Hamosh, A., Scott, A. F., Amberger, J., Valle, D., & McKusick, V. A. (2000). Online Mendelian inheritance in man (OMIM). *Human Mutation*, *15*(1), 57–61.

Hanna, J., Chen, C., Crow, W. A., Hall, R., Liu, J., Pendurthi, T., et al. (2012). Simplifying MIREOT: a MIREOT protégé plugin. In *11th international semantic web conference* (p. 25). Citeseer.

Harris, S., Seaborne, A., & Prud'hommeaux, E. (2013). SPARQL 1.1 query language. *W3C Recommendation*, *21*(10), 778.

He, Y., Xiang, Z., Zheng, J., Lin, Y., Overton, J. A., & Ong, E. (2018). The extensible ontology development (XOD) principles and tool implementation to support ontology interoperability. *Journal of Biomedical Semantics*, *9*, 1–10.

Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutierrez, C., et al. (2021). Knowledge graphs. *ACM Computing Surveys*, *54*(4), 1–37.

Ji, S., Pan, S., Cambria, E., Marttinen, P., & Philip, S. Y. (2021). A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(2), 494–514.

Jiang, X., Huang, Y., Nickel, M., & Tresp, V. (2012). Combining information extraction, deductive reasoning and machine learning for relation prediction. In *Extended semantic web conference* (pp. 164–178). Springer.

Kaushik, N., & Chatterjee, N. (2018). Automatic relationship extraction from agricultural text for ontology construction. *Information Processing in Agriculture*, *5*(1), 60–73.

Khadir, A. C., Aliane, H., & Guessoum, A. (2021). Ontology learning: Grand tour and challenges. *Computer Science Review*, *39*, Article 100339.

Kietz, J.-U., Maedche, A., & Volz, R. (2000). A method for semi-automatic ontology acquisition from a corporate intranet. In *EKAW-2000 workshop ''Ontologies and text''* (pp. 1–14).

Kozaki, K., Kushida, T., Yamamoto, Y., & Takagi, T. (2019). Building knowledge graph across different subdomains using interlinking ontology for biomedical concepts. In *Joint international semantic technology conference* (pp. 182–190). Springer.

Kumar, N., Kumar, M., & Singh, M. (2016). Automated ontology generation from a plain text using statistical and NLP techniques. *International Journal of Systems Assurance Engineering and Management*, *7*(1), 282–293.

Kumar, A., & Smith, B. (2005). Oncology ontology in the NCI thesaurus. In *Conference on artificial intelligence in medicine in Europe* (pp. 213–220). Springer.

Maguire, E., González-Beltrán, A., Whetzel, P. L., Sansone, S.-A., & Rocca-Serra, P. (2013). OntoMaton: a bioportal powered ontology widget for Google Spreadsheets. *Bioinformatics*, *29*(4), 525–527.

McBride, B. (2004). The resource description framework (RDF) and its vocabulary description language RDFS. In *Handbook on ontologies* (pp. 51–65). Springer.

McGuinness, D. L., Van Harmelen, F., et al. (2004). OWL web ontology language overview. *W3C Recommendation*, *10*(10), 2004.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, *26*.

National Research Council, et al. (2012). *Life sciences and related fields: Trends relevant to the biological weapons convention*. National Academies Press.

Noy, N. F., McGuinness, D. L., et al. (2001). *Ontology development 101: A guide to creating your first ontology*: *Stanford knowledge systems laboratory technical report KSL-01-05 and . . . .*

Noy, N. F., Shah, N. H., Whetzel, P. L., Dai, B., Dorf, M., Griffith, N., et al. (2009). BioPortal: ontologies and integrated data resources at the click of a mouse. *Nucleic Acids Research*, *37*(suppl_2), W170–W173.

Ong, E., Xiang, Z., Zhao, B., Liu, Y., Lin, Y., Zheng, J., et al. (2017). Ontobee: a linked ontology data server to support ontology term dereferencing, linkage, query and integration. *Nucleic Acids Research*, *45*(D1), D347–D352.

Overton, J. A., Dietze, H., Essaid, S., Osumi-Sutherland, D., & Mungall, C. J. (2015). ROBOT: A command-line tool for ontology development. In *ICBO* (pp. 1–2).

Petrucci, G., Ghidini, C., & Rospocher, M. (2016). Ontology learning in the deep. In *European knowledge acquisition workshop* (pp. 480–495). Springer.

Pinto, H. S., & Martins, J. P. (2001). A methodology for ontology integration. In *Proceedings of the 1st international conference on knowledge capture* (pp. 131–138).

Prud, E., Seaborne, A., et al. (2006). SPARQL query language for RDF. *W3C Recommendation*.

Roldán-García, M. d. M., García-Godoy, M. J., & Aldana-Montes, J. F. (2016). Dione: an OWL representation of ICD-10-CM for classifying patients' diseases. *Journal of Biomedical Semantics*, *7*(1), 1–16.

Singh, M. K., Mobeen, A., Chandra, A., Joshi, S., & Ramachandran, S. (2021). A meta-analysis of comorbidities in COVID-19: Which diseases increase the susceptibility of SARS-CoV-2 infection? *Computers in Biology and Medicine*, *130*, Article 104219.

Smith, B. (2004). Beyond concepts: Ontology as reality representation. In *International conference on formal ontology and information systems, vol. 4* (p. 6). Citeseer.

Smith, B., Ashburner, M., Rosse, C., Bard, J., Bug, W., Ceusters, W., et al. (2007). The OBO foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature biotechnology*, *25*(11), 1251–1255.

Staab, S., & Studer, R. (2010). *Handbook on ontologies*. Springer Science & Business Media.

Sun, Y., Ma, L., & Wang, S. (2015). A comparative evaluation of string similarity metrics for ontology alignment. *Journal of Information &Computational Science*, *12*(3), 957–964.

Wang, M., Wang, H., Liu, X., Ma, X., Wang, B., et al. (2021). Drug-drug interaction predictions via knowledge graph and text embedding: Instrument validation study. *JMIR Medical Informatics*, *9*(6), Article e28277.

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., et al. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, *3*(1), 1–9.

Xiang, Z., Courtot, M., Brinkman, R. R., Ruttenberg, A., & He, Y. (2010). OntoFox: web-based support for ontology reuse. *BMC Research Notes*, *3*, 1–12.

Xiong, Z., Huang, F., Wang, Z., Liu, S., & Zhang, W. (2021). A multimodal framework for improving in silico drug repositioning with the prior knowledge from knowledge graphs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *19*(5), 2623–2631.

Zhang, D., Mishra, S., Brynjolfsson, E., Etchemendy, J., Ganguli, D., Grosz, B., et al. (2021a). The AI index 2021 annual report. arXiv preprint arXiv:2103.06312.

Zhang, F., Sun, B., Diao, X., Zhao, W., & Shu, T. (2021b). Prediction of adverse drug reactions based on knowledge graph embedding. *BMC Medical Informatics and Decision Making*, *21*(1), 1–11.

Zheng, J., Xiang, Z., Stoeckert, C. J., Jr., & He, Y. (2014). Ontodog: a web-based ontology community view generation tool. *Bioinformatics*, *30*(9), 1340–1342.