

Methods and Metrics for Knowledge Base Engineering and Integration

Giorgos Stoilos¹, David Geleta¹, Szymon Wartak¹, Sheldon Hall¹, Mohammad Khodadadi¹, Yizheng Zhao², Ghadah Alghamdi², and Renate A. Schmidt²

¹ Babylon Health, London, SW3 3DD, UK
`firstname.lastname@babylonhealth.com`

² School of Computer Science, The University of Manchester, UK
`firstname.lastname@manchester.ac.uk`

Abstract. Today a wealth of knowledge and data are distributed using Semantic Web standards. Especially in the (bio)medical domain several sources like SNOMED, NCI, MedDRA, MeSH, ICD-10, and many more are also distributed in RDF and OWL. These can be aligned and integrated in order to create one large medical Knowledge Base. However, integrating different and largely heterogeneous sources is far from trivial. First, although distributed in OWL many of them may not strictly follow the semantics of `subClassOf` as originally intended to be used as faceted search or thesauri. Second, even if they do follow strict ontological properties, different ontologies may conceptualise the same domain in radically different ways. Analysing and understanding them before integration is a crucial issue. Third, monitoring and understanding how the structure of the Knowledge Base changes (evolves) after the integration is also crucial since changes in its structure may affect applications that are built on top of it. In the current paper we report on our Knowledge Base construction pipeline which is based on ontology integration. We focus on the various metrics, techniques, and tools we have developed in order to assist in analysing this large-scale integration tasks. Our work was motivated by the need for a medical KB to be used to support digital healthcare services developed in Babylon Health. We present results on our use of metrics to analyse various sources and the results of running the pipeline on several medical ontologies.

1 Introduction

Today a wealth of knowledge and data are distributed using Semantic Web technologies and standards. For example, the Linked Open Vocabularies effort [15] contains more than 600 ontologies for various subjects like geography, multimedia, security, geometry, and more. Especially in the biomedical domain, a large number of sources have been developed during the previous decades like SNOMED,³ NCI [4], MedDRA⁴, MeSH⁵, and many more, while BioPortal [10]

³ <https://www.snomed.org/>

⁴ <https://www.meddra.org/>

⁵ <https://meshb.nlm.nih.gov/>

is a repository of more than 600 biomedical ontologies. Identifying the common entities between these vocabularies and integrating them [5] is beneficial for building ontology-based applications as one could unify complementary information that these vocabularies contain building a “complete” Knowledge Base (KB). Such correspondences can be identified using *ontology matching (alignment)* [11].

However, identifying correspondences between them is just the first step towards performing the actual integration. Besides classes with their respective labels, sources distributed in Semantic Web standards usually come with a class hierarchy. Depending on the purpose for which they were initially created these hierarchies may exhibit significant incompatibilities. First, it is not uncommon that sources are created with the intention to be used for supporting faceted search or act as thesauri. In this case the semantics of `subClassOf` may not be the intended sub-set ones. An example of such sources in the biomedical domain is coding systems like Read Codes, MeSH, and ICD-10. Second, even if the sources follow the strict semantics of RDF(S) and OWL and even if they model the same domain they may still exhibit structural incompatibilities due to the way they conceptualise the domain. For example, in NCI proteins are declared to be disjoint from anatomical structures whereas in FMA proteins are subclasses of anatomical structures. In this case a naive integration can lead to many undesired logical consequences like unsatisfiable classes [7]. This may even be the case if an ontology \mathcal{O}_e is designed as an extension of another \mathcal{O} simply because it is developed by a different independent community which decides to alter the structure of \mathcal{O} . Last but not least, various services built on top of the KB may also impose requirements on the structure and properties of the KB. Consequently, a dire need to analyse, monitor, and evaluate the content of (large) Knowledge Bases is required [8, 18, 9].

A significant effort in creating a large medical KB recently started in Babylon Health⁶ using ontology matching and integration [14]. This medical KB is intended to support other services within Babylon like text annotation, understanding, and reasoning, drug prescribing, clinical healthcare, and more. As a critical domain the content of the KB needs to be consistent, accurate, and of high quality, raising the need to monitor the evolution process closely. This is far from trivial as some of the used sources are quite large and manual inspection is impossible. Consequently, a set of metrics have been implemented to analyse the content and structure of the KB and ensure integrity along various dimensions. Some of these metrics are inspired by well-known logic-based measures (like consistency and coherently) while others by services that depend on the KB. For example, text annotation and Named Entity Disambiguation services require that the KB exhibits a low level of “ambiguity” in order to be as easy as possible to associate classes (iris) from the KB to words in user text. The collected statistics are inspected in order to determine if the integration was successful or the pipeline needs to re-run with different parameters. The metrics are also used to assist us in analysing candidate source for integration by

⁶ <https://www.babylonhealth.com/>

assessing their structural compatibility with the current KB. Some of these metrics have been presented previously in the literature, however, some are novel or provide refinements of previous metrics in order to adapt them to our case.

In the current paper we present our ontology integration pipeline with an emphasis on the evaluation and analysis steps presenting the metrics we have implemented. We have used these to analyse the structure of many well-known medical sources and we report on the results. Next, we report on statistics about using the pipeline to integrate many popular medical ontologies like SNOMED, NCI, and FMA. To the best of our knowledge the *full* versions of these ontologies have never been actually integrated (unified) before; only smaller fragments of them have been aligned⁷ and, moreover, the computed mappings were never used to actually merge them. Finally, we developed a highly optimised logical difference analysis tool to analyse the Australian and Canadian country extensions with respect to the base SNOMED International version. Surprisingly, the Australian extension modifies the structure of the base SNOMED international ontology.

2 Ontologies and Ontology Matching

For brevity reasons, throughout the paper we will mostly use Description Logic notation. However, sometimes we will also use a triple notation, e.g., instead of $A \sqsubseteq B$ we may write $\langle A \text{ rdfs:subClassOf } B \rangle$ and instead of $A \sqsubseteq \exists R.B$ we may write $\langle A R B \rangle$. For a set of real numbers S we use $\oplus S$ to denote the sum of its elements. For p an ontology prefix and C some class if we wish to quantify the ontology in which C appears we use the notation $p:C$. Hence, for IRI prefixes $p_1 \neq p_2$, $p_1:C$ and $p_2:C$ denote different classes. For an ontology \mathcal{O} we use $\text{Sig}(\mathcal{O})$ to denote the set of classes that appear in \mathcal{O} . Given an ontology \mathcal{O} we assume that all classes C in \mathcal{O} have at least one triple of the form $\langle C \text{ skos:prefLabel } v \rangle$ and zero or more triples of the form $\langle C \text{ skos:altLabel } v_i \rangle$. For a given class C function $\text{pref}(C)$ returns the string value v in the triple $\langle C \text{ skos:prefLabel } v \rangle$. An ontology is called *coherent* if every $C \in \text{Sig}(\mathcal{O})$ with $C \neq \perp$ is satisfiable.

In the literature, the notion of a Knowledge Base is almost identical to that of an ontology, i.e., a set of axioms describing the entities of a domain. In the following, we loosely use the term “Knowledge Base” (\mathcal{KB}) to mean a possibly large ontology that has been created by integrating various other ontologies but, formally speaking, a \mathcal{KB} is an OWL ontology.

Ontology matching (or *ontology alignment*) is the process of discovering correspondences (mappings) between the entities of two ontologies \mathcal{O}_1 and \mathcal{O}_2 . To represent mappings we use the formulation presented in [7]. That is, a mapping between \mathcal{O}_1 and \mathcal{O}_2 is a 4-tuple of the form $\langle C, D, \rho, n \rangle$, where $c \in \text{Sig}(\mathcal{O}_1)$ $d \in \text{Sig}(\mathcal{O}_2)$, $\rho \in \{\equiv, \sqsupseteq, \sqsubseteq\}$ is the *mapping type*, and $n \in (0, 1]$ is the confidence value of the mapping. Moreover, we interpret mappings as DL axioms—that is, $\langle C, D, \rho, n \rangle$ can be seen as the axiom $C \rho D$ with the degree attached as an

⁷ <http://www.cs.ox.ac.uk/isg/projects/SEALS/oei/2017/results/>

annotation. Hence, for a mapping $\langle C, D, \rho, n \rangle$ when we write $\mathcal{O} \cup \{\langle C, D, \rho \rangle\}$ we mean $\mathcal{O} \cup \{C \rho D\}$ while for a set of mappings \mathcal{M} , $\mathcal{O} \cup \mathcal{M}$ denotes the set $\mathcal{O} \cup \{m \mid m \in \mathcal{M}\}$. When not relevant and for simplicity we will often omit ρ and n and simply write $\langle C, D \rangle$. A **matcher** is an algorithm that takes as input two ontologies and returns a set of mappings.

3 Building Large Knowledge Bases

Large KBs can be constructed by integrating existing, complementary, and possibly overlapping ontologies. For example, in the biomedical domain, ontologies for diseases, drugs, drug side-effects, genes, and so on, exist that can be integrated in order to build a large medical KB. However, before putting two sources together it would be beneficial to discover their overlapping parts and establish mappings (correspondences) between their equivalent entities.

Example 1. Consider an ontology-based medical application that is using the SNOMED ontology $\mathcal{O}_{\text{snmd}}$ as a KB. Although SNOMED is a large and well-engineered ontology it is still missing relevant medical information like relations between diseases and symptoms. For example, for the disease “Ewing Sarcoma” SNOMED only contains the axiom $\text{snmd:EwingSarcoma} \sqsubseteq \text{snmd:Sarcoma}$ and no relations to signs or symptoms. In contrast, the NCI ontology \mathcal{O}_{nci} contains the following axiom about this disease:

$$\text{nci:EwingSarcoma} \sqsubseteq \exists \text{nci:mayHaveSymptom.nci:Fever}$$

We can use ontology matching to establish links between the related entities in $\mathcal{O}_{\text{snmd}}$ and \mathcal{O}_{nci} and then integrate the two sources in order to enrich our KB. More precisely, using some matching algorithm we can identify the following mappings:

$$\begin{aligned} m_1 &= \langle \text{snmd:EwingSarcoma}, \text{nci:EwingSarcoma}, \equiv \rangle \\ m_2 &= \langle \text{snmd:Fever}, \text{nci:Fever}, \equiv \rangle \end{aligned}$$

and hence replace our KB with $\mathcal{O}'_{\text{snmd}} := \mathcal{O}_{\text{snmd}} \cup \mathcal{O}_{\text{nci}} \cup \{m_1, m_2\}$. Then, $\mathcal{O}'_{\text{snmd}}$ contains the knowledge that “Ewing sarcoma may have fever as a symptom”. \diamond

Unfortunately, besides bringing new information into our existing KB, the above integration may also lead to various undesired consequences. Due to differences in the structure of the two ontologies, the integration may introduce unsatisfiable classes [7] or simply structural changes to the initial KB [5].

Example 2. Consider again the SNOMED and NCI ontologies. Both ontologies contain classes for the notion of “soft tissue disorder” and “epicondylitis”. Hence, it is reasonable for a matching algorithm to compute the following mappings:

$$\begin{aligned} m_1 &= \langle \text{snmd:SoftTissueDisorder}, \text{nci:SoftTissueDisorder}, \equiv \rangle \\ m_2 &= \langle \text{snmd:Epicondylitis}, \text{nci:Epicondylitis}, \equiv \rangle \end{aligned}$$

Algorithm 1 KnowledgeBaseConstruction($\mathcal{KB}, \mathcal{O}, \text{Config}$)

Input: The current KB \mathcal{KB} , a new ontology \mathcal{O} and a configuration Config .

```
1:  $\mathcal{O} = \text{saturate}(\mathcal{O})$ 
2:  $\text{Mappings} := \emptyset$ 
3: for all  $\text{matcher} : \text{Config.Align.Matchers}$  do
4:   for all  $\langle C, D, \rho, n \rangle \in \text{matcher}(\mathcal{KB}, \mathcal{O})$  do
5:      $\text{Mappings} := \text{Mappings} \cup \{\langle C, D, \rho, n, \text{matcher} \rangle\}$ 
6:   end for
7: end for
8:  $\mathcal{M}_f := \emptyset$ 
9:  $w = \oplus\{\text{matcher}.w \mid \text{matcher} \in \text{Config.Align.Matchers}\}$ 
10: for all  $\langle C, D, \rho, \neg, \cdot \rangle \in \text{Mappings}$  such that no  $\langle C, D, \rho, n \rangle$  exists in  $\mathcal{M}_f$  do
11:    $n := \oplus\{n_i \times \text{matcher}.w \mid \langle C, D, \rho, n_i, \text{matcher} \rangle \in \text{Mappings}\} / w$ 
12:   if  $n \geq \text{Config.Align.thr}$  then
13:      $\mathcal{M}_f := \mathcal{M}_f \cup \{\langle C, D, \rho, n \rangle\}$ 
14:   end if
15: end for
16:  $\langle \mathcal{O}', \mathcal{M}_f \rangle := \text{postProcess}(\mathcal{KB}, \mathcal{O}, \mathcal{M}_f, \text{Config})$ 
17:  $\text{Model}_{\mathcal{KB}} := \text{analyse}(\mathcal{KB})$ 
18:  $\text{Model}_{\mathcal{KB}'} := \text{analyse}(\mathcal{KB} \cup \mathcal{O}' \cup \mathcal{M}_f)$ 
19:  $\text{DiffModel} := \text{diff}(\text{Model}_{\mathcal{KB}}, \text{Model}_{\mathcal{KB}'})$ 
20:  $\text{Report} := \text{produceReport}(\text{DiffModel}, \text{Config.Expectations})$ 
21: return  $\mathcal{KB} \cup \mathcal{O}' \cup \mathcal{M}_f$ 
```

However, in NCI we have $\mathcal{O}_{\text{nci}} \models \text{nci:Epicondylitis} \sqsubseteq \text{nci:SoftTissueDisorder}$ while in SNOMED $\mathcal{O}_{\text{snmd}} \not\models \text{snmd:Epicondylitis} \sqsubseteq \text{snmd:SoftTissueDisorder}$. Hence, for $\mathcal{KB}^{\text{int}} := \mathcal{O}_{\text{snmd}} \cup \mathcal{O}_{\text{nci}} \cup \{m_1, m_2\}$ the integrated ontology we will have:

$$\mathcal{KB}^{\text{int}} \models \text{snmd:Epicondylitis} \sqsubseteq \text{snmd:SoftTissueDisorder}$$

introducing a relation between classes of $\mathcal{O}_{\text{snmd}}$ that did not originally hold.

To repair the above issue the typical approach followed in literature is to remove some of the computed mappings, i.e., either m_1 or m_2 [7, 5, 12]. But removing mappings will cause the KB to contain two different classes for the same real world entity with a large overlap in their labels. An alternative approach studied in detail in [14] is to remove axioms from the new ontology. In this example, we can compute $\mathcal{KB}_2^{\text{int}} := \mathcal{KB}_1^{\text{int}} \setminus \{\text{nci:Epicondylitis} \sqsubseteq \text{nci:SoftTissueDisorder}\}$ and hence $\mathcal{KB}_2^{\text{int}} \not\models \text{snmd:Epicondylitis} \sqsubseteq \text{snmd:SoftTissueDisorder}$ as desired. \diamond

In addition, various services that are build on top of the KB may impose other types of requirements on the KB and integrating sources may impact these in a negative way.

Example 3. Assume that our SNOMED-based KB is used to support medical text annotation and Named Entity Disambiguation services. These take a user text like “I have a severe pain in my head” and annotate it with classes from the KB. More precisely, words “severe”, “pain”, and “head” would be associated

with the respective classes in the SNOMED ontology assigning meaning to text. For example, the word “severe” would be associated to SNOMED class **Severe**.

Assume now that NCI is integrated with SNOMED. NCI contains class **SevereAdverseEvent** with an alternative (synonymous) label “severe”. Consequently, after the integration there are two different classes that can be associated to word “severe”. The choice of which class to use may have significant impact on the application as different choices assign different meaning to the text. \diamond

Based on the above we have designed a knowledge base construction algorithm (Algorithm 1) that is extensively using quality and ontology evaluation metrics in order to monitor various dimensions and properties of the enrichment process. The algorithm accepts as input the current KB \mathcal{KB} , a new ontology \mathcal{O} which will be used to enrich \mathcal{KB} and a configuration **Config**, which is used to tune and change various parameters like thresholds and weights. In brief, the algorithm first saturates the input ontology using an OWL reasoner like *HermiT* [3] as well as in-house implemented saturation rules. As the KB is loaded to a triple-store for scalable SPARQL query answering this step is meant to improve completeness of SPARQL query answering [13]. Subsequently, it applies a set of matchers in order to compute a set of mappings between \mathcal{KB} and \mathcal{O} and then aggregates them removing those that fall below a certain threshold (**Config.Align.thr**). As mentioned previously, structural changes are eliminated and this is performed in method **postProcess**. This method is quite involved and details can be found in [14].

Finally, a set of analysers are applied on the knowledge base before and after the integration. These analysers compute various metrics on the given KBs and populate two models on which a diff operation is applied. These differences are then compared against “expectations” that are set before starting the pipeline. For example, if we integrate an ontology which is rich in alternative labels then we expect that such types of axioms in the initial KB increase to a proportional amount. The various statistics and metrics that are used in function **analyse** are presented in detail in the next section.

4 Analysing Knowledge Bases

We have grouped our metrics according to two categories. The first one contains metrics that characterise the integrity of the KB, that is, the “correctness” of its content according to either well-known notions or service induced properties. The second category regards computing statistics about the actual content of the KB like assessing its completeness. Many of our metrics are inspired by work on ontology and knowledge base evaluation [16, 9, 2] or Linked Data quality analysis [18]; some are adapted or extended to fit our use case or are newly proposed.

4.1 KB Integrity

Integrity can be measured using standard logic-based notions but also additional application specific metrics are relevant to our use case. In the following we present in detail the metrics we have defined grouping them in various sub-categories.

KB Coherency Perhaps one of the most fundamental properties of every KB is that it is coherent—that is, it does not contain unsatisfiable classes. Formally, for \mathcal{KB} our KB and A any class in \mathcal{KB} we should have $\mathcal{KB} \not\models A \sqsubseteq \perp$. This check can be performed using existing OWL reasoners. Unfortunately, when we are dealing with large KBs possibly containing billions of statements, such checks are at-least time-consuming if at all possible. Consequently, our coherency-checking algorithms are based on approximate methods and techniques inspired by the DL-Lite language [1]. More precisely, function `analyse` internally computes the following set:

$$\{C \in \mathcal{KB} \mid \mathcal{KB} \models_{\text{rdfs}} C \sqsubseteq A \sqcap B \text{ where } A \sqcap B \sqsubseteq \perp \in \mathcal{KB}\}$$

where \models_{rdfs} means only those entailments under the RDFS-semantics. This check is implemented by the following SPARQL query:

```
select ?a ?b ?s where {
  ?s rdfs:subClassOf ?a ; rdfs:subClassOf ?b .
  { select ?a ?b where { ?a owl:disjointWith ?b . } }
} group by ?a ?b ?s
order by ?a
```

In a similar way we can have checks for Functional and/or InverseFunction-Properties, however, no such axioms are currently declared in our KB.

Entailment Invariability Metrics related to entailment invariability capture the aspects discussed in Example 2. Although such changes are eliminated by function `postProcess` this dimension is still analysed in order to ensure that everything worked as desired in the pipeline. The amount of entailment changes between the original and a new ontology can be captured by the notion of logical difference [6].

Definition 1. *Let Σ be a signature and let \mathcal{O} and \mathcal{O}' be two OWL 2 ontologies. The Σ -deductive difference between \mathcal{O} and \mathcal{O}' (denoted $\text{diff}_{\Sigma}(\mathcal{O}, \mathcal{O}')$) is defined as the set of axioms α satisfying: (i) $\text{Sig}(\alpha) \subseteq \Sigma$, (ii) $\mathcal{O} \not\models \alpha$, and (iii) $\mathcal{O}' \models \alpha$.*

This notion can be used as follows: given the initial KB \mathcal{KB} , a new ontology \mathcal{O} and a set of mappings \mathcal{M}_f computed between them, we ideally want that $\text{diff}_{\Sigma}(\mathcal{KB}, \mathcal{KB} \cup \mathcal{O} \cup \mathcal{M})$ where $\Sigma = \text{Sig}(\mathcal{KB})$. Computing logical differences between expressive let alone large KBs is at-least challenging if possible at all.

Nevertheless, a highly optimised system, called FAME [19], has been implemented within the scope of this project. The system is based on the similar notion of uniform interpolation and is highly optimised to be able to scale on large ontologies. As we will show in the evaluation section FAME was able to compute logical differences between SNOMED versions.

Despite optimised, FAME is not yet able to handle the full scale of our integrated KB. Hence, we also considered the approximate version of the above definition introduced in [12] and the difference is only computed with respect to axioms of the form $A \sqsubseteq B$. In addition to that, we also look for differences in axioms of the form $A \sqsubseteq \exists R.A$. Such axioms imply a form of self-loop loop, e.g., saying that `Let` \sqsubseteq `partOf.Leg`, which we also want to exclude from occurring in our KB. In contrast to [12], however, we do allow for new entailments of the form $A \sqsubseteq \exists R.B$ for $B \neq A$. Actually such new knowledge is desired as it implied the enrichment of our KB with new relations (see also Example 1). Like before the above set is computed using SPARQL queries over triple-stores and no OWL reasoner is used.

Graph-based Invariability All above metrics are based on well-defined notions from logics. In addition to those, a number of graph-based metrics can also be identified and can be helpful in further analysing ontologies and comprehending their structural properties. The metrics currently implemented are the following:

1. **Path lengths:** the maximum and average path length of the subClassOf hierarchy are computed using a depth-first algorithm.
2. **Tangledness:** This notion was introduced in [2] as a way to quantify the multi-hierarchical nature of classes in an ontology, that is, how often a class has more than one parents (a fork) compared to the total number of classes in the ontology. Building on this we have defined our version of tangledness which is measuring the fork-rejoin points of each class—that is, how many times a class has more than two parents and what is their least common subsumer of them. We feel that this characterised better the multi-hierarchical nature of ontologies. First we define the least-common-subsumer (*lcs*) of two classes A and B as:

$$lcs(A, B) := \{C \mid \mathcal{KB} \models A \sqsubseteq C, B \sqsubseteq C \text{ and for every } D \text{ s.t.} \\ \mathcal{KB} \models A \sqsubseteq D, B \sqsubseteq D \text{ we have } \mathcal{KB} \models C \sqsubseteq D\}.$$

Now for every class in the KB we define the following:

$$tang(A, \mathcal{KB}) := \{E \in \mathcal{KB} \mid \{A \sqsubseteq C_1, A \sqsubseteq C_2\} \subseteq \mathcal{KB}, C_1 \neq C_2 \\ \text{and } E \in lcs(C_1, C_2)\}$$

As we will see in Section 5 these sets can help us understand the multi-hierarchical nature of data sources and assess their internal structure. A

SPARQL query computing the above set is given next:

```
select ?s ?d1 ?d2 ?anc where {
  ?s sesame:directSubClassOf ?d1 ; sesame:directSubClassOf ?d2.
  filter (?d1 != ?d2).
  ?d1 rdfs:subClassOf ?anc . ?d2 rdfs:subClassOf ?anc .
  filter not exists { ?d1 rdfs:subClassOf ?anc2 .
    ?d2 rdfs:subClassOf ?anc2 . ?anc2 rdfs:subClassOf ?anc . }
}
```

In addition, we have defined the sum of fork-rejoin points that occur on the descendants of a class. This number can help us understand parts of the KB in which large structural changes occurred after the integration.

$$\text{tang}_{\downarrow}(A, \mathcal{KB}) := \Sigma \{ \# \text{tang}(C, \mathcal{KB}) \mid \mathcal{KB} \models C \sqsubseteq A \}$$

Label integrity/ambiguity As motivated by Example 3 we wish to avoid having different classes sharing a common labels. Obviously this is impossible in general as language is inherently ambiguous. For example, SNOMED alone already contains several classes that have overlapping labels, e.g., two classes with label “foot”, one for the unit of measurement and one for the body part. One could use the “types” (categories) of these classes to disambiguate, however, this is still a hard problems for text annotation services. For this reason we have developed methods to measure and eliminate duplication on labels as much as possible. Let $\mathbb{C} \subseteq \text{Sig}(\mathcal{KB})$ be as set of “categories” in \mathcal{KB} —that is, classes that denote groups of other classes, e.g., $\mathbb{C} := \{\text{Disease}, \text{Drug}, \text{BodyPart}, \dots\}$. For every $C \in \mathbb{C}$ we define the following set:

$$\text{amb-lab}(C, \mathcal{KB}) := \{ \ell \mid A \neq B \text{ exist s.t. } \mathcal{KB} \models \{A \sqsubseteq C, B \sqsubseteq C\}, \text{ and } \{ \langle A \text{ skos:label } \ell \rangle, \langle B \text{ skos:label } \ell \rangle \} \subseteq \mathcal{KB} \}$$

The following SPARQL query computes the above set:

```
select ?l where {
  ?s1 rdfs:subClassOf :C ; skos:prefLabel|skos:altLabel ?l .
  ?s2 rdfs:subClassOf :C ; skos:prefLabel|skos:altLabel ?l .
  filter (?s1 != ?s2)
}
```

Moreover, we also define $\text{amb-lab}(\mathcal{KB}) = \Sigma_{C \in \mathbb{C}} \# \text{amb-lab}(C, \mathcal{KB})$.

As we will show in the evaluation section single ontologies may come with a significant amount of ambiguity due to the “loose” way ontology authors may use synonyms and alternative labels. To reduce ambiguity we have developed a set of heuristics which can be used in the pipeline. Some of these heuristics are the following:

- If ℓ appears as a preferred label in one class and as an alternative in the other then delete the latter.

- If ℓ appears in two classes one of which is a super-class of the other and the label in the sub-class is not its preferred label, then delete the label from the sub-class.
- If ℓ appears in two classes that have a common direct super-class (i.e., in two sibling classes), then delete the label from both of them and create an intermediate parent containing this label.

Clearly, these heuristics do not completely eliminate ambiguity as there are more combinations not covered by them.

4.2 KB Completeness Assessment

The completeness or coverage that a knowledge base provides to the underlying domain is another relevant notion for which metrics have been defined in the literature [16, 9]. Quantifying completeness is in general impossible since to do that one would need to know what is all the possible knowledge to which the content of the KB should be compared. Hence, at best we can analyse the content of the KB and assess by manual inspection and consulting domain experts what type of content the KB is missing.

For some relation $R \in \text{Sig}(\mathcal{KB})$ let $\text{dom}(R) := \{C \mid \langle R \text{ rdfs:domain } C \rangle \in \mathcal{KB}\}$ and $\text{ran}(R) := \{D \mid \langle R \text{ rdfs:range } D \rangle \in \mathcal{KB}\}$. Our function *analyse* computes the following sets:

1. Relation usage:

$$\text{usage}(R) := \#\{\langle A \ R \ B \rangle \in \mathcal{KB} \mid \mathcal{KB} \models \{A \sqsubseteq C, B \sqsubseteq D\}, \\ C \in \text{dom}(R), D \in \text{dom}(R)\}$$

$$\text{usage}(C, R) := \#\{\langle A \ R \ B \rangle \in \mathcal{KB} \mid \mathcal{KB} \models \{A \sqsubseteq C, B \sqsubseteq D\}, D \in \text{ran}(R)\}$$

The above measures are quite similar to the metric $\text{freq}(R, C)$ defined in [9] but here we only count triples that fall within the “specified” use of the relation—that is, within their defined domain and range. However, in a large ontology integration project and due to the scale of the alignment problem, relations may start to “connect” classes whose types fall outside the domain and ranges of the relations. We call this issue *drift* and we measure it according to the following sets where $\langle A \ R \ B \rangle \in \mathcal{KB}$:

$$\text{drift}_o(R) := \#\{\langle A \ R \ B \rangle \mid \mathcal{KB} \models A \sqsubseteq C \text{ for some } C \in \text{dom}(R), \text{ and} \\ \mathcal{KB} \not\models B \sqsubseteq D \text{ for every } D \in \text{ran}(R)\}$$

$$\text{drift}_s(R) := \#\{\langle A \ R \ B \rangle \mid \mathcal{KB} \models B \sqsubseteq D \text{ for some } D \in \text{ran}(R) \text{ and} \\ \mathcal{KB} \not\models A \sqsubseteq C \text{ for all } C \in \text{dom}(R)\}$$

$$\text{drift}_{so}(R) := \#\{\langle A \ R \ B \rangle \mid \mathcal{KB} \not\models A \sqsubseteq C, \mathcal{KB} \not\models B \sqsubseteq D \text{ for any} \\ C \in \text{dom}(R) \text{ and } D \in \text{ran}(R)\}$$

Next for a relation R with domain $C \in \text{dom}(R)$ we measure the percentage of classes of this type that have this relation associated to them:

$$\text{extension}(C, R) := \frac{\#\{A \mid A \sqsubseteq C \wedge \text{usage}(A, R) > 0\}}{\#\{A \mid A \sqsubseteq C\}}$$

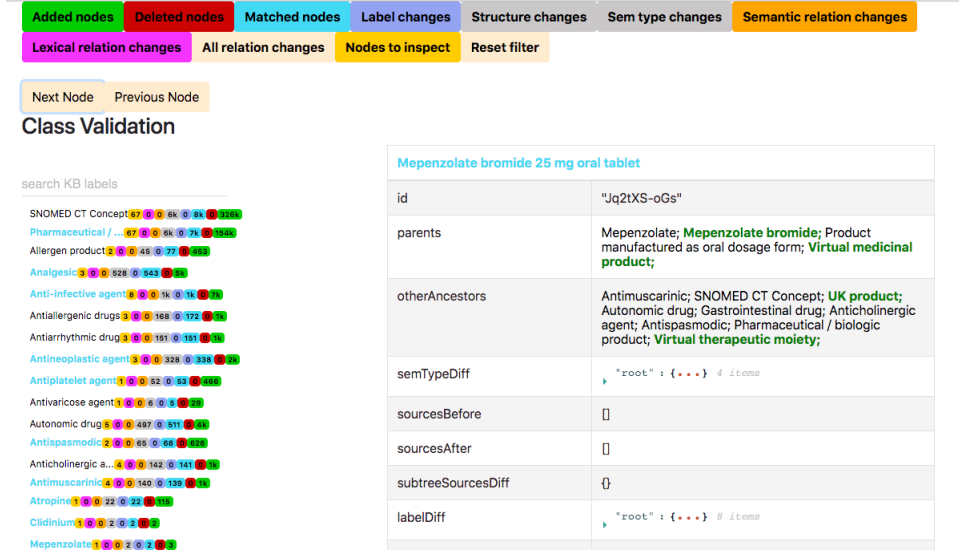


Fig. 1. KB and Class inspection

The above metric can also be used for data type properties like `skos:definition` to measure how many classes in the ontology have associated definitions. In this case we have `extension(owl:Thing, skos:definition)`. A similar metric in [9] is the normalised frequency.

2. **Class undefinedness:** For a class A we define its level of “undefinedness” as the set of relations that should be defined for this class according to its specified domains.

$$\text{undef}(A) := \{R \in \mathcal{KB} \mid \mathcal{KB} \models A \sqsubseteq C \text{ for some } C \in \text{dom}(R) \text{ and } \mathcal{KB} \not\models \langle A \ R \ B \rangle \text{ for any } B\}$$

4.3 Metric Inspection and KB Verification

After all metrics are extracted, a diff operation is performed on several of them (line 19) in order to provide with an intuition about the delta in the content and integrity of the KB. In addition to the above metrics a diff is also performed on every class of the KB to compute its change of labels, properties, and ancestors. Many of these diffs are compared against pre-set “expectations” the violation of which can raise errors or warnings depending on how critical a metric is. For example, coherency is a strong requirement hence if after integration the new KB is not coherent an error is raised and the set of unsatisfiable concepts are put in the report. Other expectations can be that the number of triples for some subset of properties should increase. For example, if the new ontology is rich in “has symptom” relations then we expect that “completeness” with respect to this relation increases. Another example is the length of `subClassOf` paths

Table 1. Statistics about the KB after each integration/enrichment iteration.

	SNOMED	NCI	MesSH	MedDRA	ctv3	ICD-10	Read2	FMA
$count(\mathbf{tang})$	118 120	12 529	7 950	8 248	10 092	0	0	0
$avg(\mathbf{tang})$	2.0	1.26	1.2	1.0	1.14	0	0	0
$avg(\mathbf{tang}_{\downarrow})$	27.0	1.9	0.8	0.3	0.33	0	0	0
$max(\mathbf{tang})$	32	11	8	1	9	0	0	0
$amb\text{-}lab(\mathcal{KB})$	1 072	4 873	0	5	24 960	708	1 139	261

which should not increase significantly. More formally, for $\mathcal{KB}, \mathcal{O}, \mathcal{M}_f$ the sets computed up to line 16, we should have:

$$\text{depth}(\mathcal{KB} \cup \mathcal{O} \cup \mathcal{M}_f) \leq \text{depth}(\mathcal{KB}) + w \times \text{depth}(\mathcal{O})$$

for some $w \in (0, 1]$.

To further assist in inspecting the content of the KB a graphical tool has been developed in Babylon in which the diffs can be visualised; Figure 1 depicts a screen-shot. Users can navigate the KB hierarchy and click on concepts to view their diff with respect to the previous KB version. New information that is due to the integration is highlighted with different colours (e.g., green for new ancestors). A weighted formula on the diff of each class is also computed in order to assess the most “changed” classes in the KB and select a subset of them for doctor-based verification [17]. Fine tuning the formula and developing a doctor-based verification methodology is currently under further investigation.

5 Evaluation

Algorithm 1 has been fully implemented in a highly modular and configurable pipeline and used to build a large medical KB in Babylon Health. Several medical sources have been considered. As mentioned, determining whether two sources can be integrated “smoothly”, monitoring the whole process at such large scale, and fine tuning the parameters of the pipeline is far from trivial. Initially, we used the tangledness and label ambiguity metrics to analyse the following popular medical data sources: SNOMED, NCI, MeDRA, MeSH, ICD-10, Read2, CTV3 (a successor of READ2), and FMA; the results are depicted in Table 1 where $count(\mathbf{tang}) = \#\{A \mid \mathbf{tang}(A) > 0\}$.

As can be seen SNOMED is a highly multi-hierarchical ontology followed by NCI and to some extent also MeSH although out of the almost 8K classes that had a fork-rejoin in MeSH the rejoin point was `owl:Thing`. In MedDRA *all* forks have `owl:Thing` as a least-common-subsumer while in ICD-10 and Read2 there are no forks, a confirmation that all these sources are classification systems. Interestingly, also the FMA ontology does not contain any forks. This ontology models the human anatomy and perhaps is reasonable to assume that some body part is not classified as two different things at the same time. Note that ctv3 is a successor of Read2 and apparently a more ontological approach is followed.

Table 2. Statistics about the KB after each integration/enrichment iteration.

	SNOMED	+NCI	+CHV	+FMA
Classes	340 995	429 241	429 241	524 837
Properties	93	124	124	219
SubClassOf Axioms	511 656	617 542	617 542	713 313
ObjPropAssertions	526 146	664 742	664 742	962 190
DataPropAssertions	543 416	946 801	1 043 874	1 211 459
LDiff	\emptyset	\emptyset	\emptyset	\emptyset
Ambiguity	1 072	5 768	9 207	9 811
Ambiguity-Repaired	180	1 266	1 892	2 078

Regarding ambiguity we note that NCI exhibits a high degree of ambiguity. After closer inspection we concluded that the alternative labels in NCI do not represent synonyms but are used in a loose way to indicate similar terms (see also Example 3).

Based on the above we selected SNOMED as our seed ontology and used Algorithm 1 to build a medical KB. On top of SNOMED we have so far integrated NCI (which contains 130K classes and 143K subClassOf axioms), CHV (which contains 57K classes and 0 subClassOf axioms) and FMA (which contains 104K classes and 255K subClassOf axioms). CHV is a flat list of layman terms of medical concepts from which we only integrated label (synonym) information; hence CHV was also not included in the previous analysis. Due to the high ambiguity in NCI its alternative labels were given lower weight in the matching process. Statistics about the KBs that we created after each integration step are depicted in Table 2. As can be seen our `postProcess` method completely eliminated structural changes something that does not happen when using other ontology matching frameworks (see Evaluation in [14]). Moreover, our heuristics do reduce ambiguity significantly and a doctor-based evaluation showed that they are about 95% correct.

In addition to the above statistics we have also applied our completeness assessment metrics after each integration step to monitor how the content changes. These metrics have helped us in at least the following cases:

- NCI is rich in `skos:definition` hence our expectations were that the number of such triples would significantly increase in the KB. The first integration of NCI was rejected since there was no increase due to a bug.
- Integration of NCI introduced a range drift on property `partOf` whose range is `BodyParts`. This was because the NCI “part of” property was declared to be a sub-property of this property while the range of the NCI relation is either `BodyPart` or `Substance`. Consequently, the sub-property axiom was removed.
- As expected the integration of CHV introduced many alternative labels to existing classes in the KB while that of FMA many `partOf` relations between body parts. Moreover, the integration of NCI introduced many `hasFinding` relations between diseases and symptoms.

Finally, we have also considered the Canadian and Australian extensions of SNOMED international, denoted by $\mathcal{O}_{\text{snmd}}^{ca}$ and $\mathcal{O}_{\text{snmd}}^{au}$, respectively. These extend SNOMED with additional labels, country specific concepts, and provide additional local variations and customisations relevant to healthcare communities of these countries. Ideally we should have $\text{diff}_{\Sigma}(\mathcal{O}_{\text{snmd}}, \mathcal{O}_{\text{snmd}} \cup \mathcal{O}_{\text{snmd}}^*) = \emptyset$ for $* \in \{au, ca\}$ and $\Sigma = \text{Sig}(\mathcal{O}_{\text{snmd}})$. If this is the case then these can be “safely” integrated into the existing KB. Note that no alignment is required as the country extensions reuse the IRIs of SNOMED and any new IRI is a new classes not in SNOMED.

We used FAME to compute the above sets obtaining the following results: for $* = ca$ the above set is indeed empty and the Canadian extension simply enriches SNOMED international with additional labels and classes without affecting its structure; unfortunately, this is not the case for the Australian extension in which the above set contains almost 43K axioms. Surprisingly enough there is a case of a class subsumption $A \sqsubseteq B \in \mathcal{O}_{\text{snmd}}$ which in $\mathcal{O}_{\text{snmd}}^{au}$ is as $B \sqsubseteq A$. How to properly deal with Australian SNOMED remains part of future work.

6 Conclusions

We present our framework for large KB construction and engineering which is based on ontology integration. The framework has been used to build a large medical Knowledge Graph by integrating the popular and well-known ontologies SNOMED, NCI, FMA as well as labels from the CHV vocabulary. To the best of our knowledge, no ontology integration at that scale has been performed in the past; all matching efforts have used much smaller versions of them and the computed mappings were never actually used to merge them.

To help us decide which sources to use and how to configure our pipeline a set of analysers were implemented. There are also used at the end of each pipeline run in order to evaluate the integration process and assess the quality of the final enriched KB. To further assist in the verification process a graphical user interface was also built.

We present our results in applying several of these metrics on well-known medical data sources that are currently under investigation in Babylon. Our results show interesting properties about them like ambiguity of their labels and their structure like their multi-hierarchical nature. Our metrics and verification mechanism assisted in fixing several errors in the pipeline, assessing the success of the integration and fine tuning it. Finally, the FAME system helped us determine that the Australian extension of SNOMED is not a conservative extension.

References

1. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: *DL-lite: Tractable description logics for ontologies*. In: *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI)*. pp. 602–607 (2005)

2. Gangemi, A., Catenacci, C., Ciaramita, M., Lehmann, J.: Modelling ontology evaluation and validation. In: Proceedings of the 3rd European Semantic Conference, ESWC. pp. 140–154 (2006)
3. Glimm, B., Horrocks, I., Motik, B., Stoilos, G., Wang, Z.: Hermit: An owl 2 reasoner. Journal of Automated Reasoning (JAR), In Press 53, 245–269 (2014)
4. Golbeck, J., Frago, G., Hartel, F.W., Hendler, J.A., Oberthaler, J., Parsia, B.: The national cancer institute’s thesaurus and ontology. Journal of Web Semantics 1(1), 75–80 (2003)
5. Jiménez-Ruiz, E., Grau, B.C., Horrocks, I., Llavori, R.B.: Ontology integration using mappings: Towards getting the right logical consequences. In: Proceedings of the 6th European Semantic Web Conference, (ESWC). pp. 173–187 (2009)
6. Konev, B., Walthers, D., Wolter, F.: The logical difference problem for description logic terminologies. In: Proceedings of the 4th International Joint Conference on Automated Reasoning, IJCAR. pp. 259–274 (2008)
7. Meilicke, C., Stuckenschmidt, H.: An efficient method for computing alignment diagnoses. In: Proceedings of the 3rd International Conference on Web Reasoning and Rule Systems, RR. pp. 182–196 (2009)
8. Mendes, P.N., Mühleisen, H., Bizer, C.: Sieve: Linked data quality assessment and fusion. In: Proceedings of the 2012 Joint EDBT/ICDT Workshops. pp. 116–123 (2012)
9. Rashid, M., Torchiano, M., Rizzo, G., Mihindukulasooriya, N., Corcho, O.: Knowledge base quality assessment using temporal analysis. Submitted to the Semantic Journal Web, under review (2017)
10. Salvadores, M., Alexander, P.R., Musen, M.A., Noy, N.F.: Bioportal as a dataset of linked biomedical ontologies and terminologies in RDF. Semantic Web 4(3), 277–284 (2013)
11. Shvaiko, P., Euzenat, J.: Ontology matching: State of the art and future challenges. IEEE Transactions on Knowledge and Data Engineering 25(1), 158–176 (2013)
12. Solimando, A., Jiménez-Ruiz, E., Guerrini, G.: Detecting and correcting conservativity principle violations in ontology-to-ontology mappings. In: Proceedings of the 13th International Semantic Web Conference (ISWC). pp. 1–16 (2014)
13. Stoilos, G., Cuenca Grau, B., Motik, B., Horrocks, I.: Repairing ontologies for incomplete reasoners. In: Proceedings of the 10th International Semantic Web Conference (ISWC-11), Bonn, Germany (2011)
14. Stoilos, G., Geleta, D., Shamdasani, J., Khodadadi, M.: A novel approach and practical algorithms for ontology integration. In: Proceedings of the International Semantic Web Conference (ISWC) (2018)
15. Vandenbussche, P., Atemez, G., Poveda-Villalón, M., Vatan, B.: Linked open vocabularies (LOV): A gateway to reusable semantic vocabularies on the web. Semantic Web 8(3), 437–452 (2017)
16. Vrandečić, D.: Ontology evaluation. In: Handbook on Ontologies, pp. 293–313. Springer (2009)
17. Zaveri, A., Kontokostas, D., Sherif, M.A., Böhmann, L., Morsey, M., Auer, S., Lehmann, J.: User-driven quality evaluation of dbpedia. In: Proceedings of the 9th International Conference on Semantic Systems. pp. 97–104 (2013)
18. Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., Auer, S.: Quality assessment for linked data: A survey. Semantic Web 7(1), 63–93 (2016)
19. Zhao, Y., Schmidt, R.A.: Forgetting concept and role symbols in $\mathcal{ALCOI\mathcal{H}}_{\mu}^{+}(\nabla, \sqcap)$ -ontologies. In: Kambhampati, S. (ed.) Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI’16). pp. 1345–1352. AAAI Press/IJCAI (2016)