Methods and Metrics for Knowledge Base Engineering and Integration

Giorgos Stoilos[1], David Geleta[1], Szymon Wartak[1], Sheldon Hall[1], Mohammad Khodadadi[1], Yizheng Zhao[2], Ghadah Alghamdi[2], and Renate A. Schmidt[2]

[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 the SNOMED CT, NCI, MedDRA, MeSH, ICD-10 ontologies, and many more are 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 the ontologies may not strictly follow the semantics of subClassOf as originally intended for faceted search or use as thesauri. Second, even when they do follow strict ontological guidelines, different ontologies may conceptualise the same domain in radically different ways. Analysing and understanding these sources before integrating them is highly beneficial. Third, monitoring and understanding how the structure of the Knowledge Base changes (evolves) after the integration is also crucial since changes to 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 achieving this large-scale integration task. Our work was motivated by the need for a medical Knowledge Base to be used to support digital healthcare services developed at Babylon Health. We present results on the metrics used 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 such as the

SNOMED CT[3], NCI [4], MedDRA[4], MeSH[5] ontologies, and many more, while BioPortal [10] 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*) techniques [11].

However, identifying correspondences between ontologies 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 ontologies 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 subset relationship. Examples of such sources in the biomedical domain are 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 the NCI ontology proteins are declared to be disjoint from anatomical structures whereas in the FMA ontology proteins are subclasses of anatomical structures. In this case a naive integration can lead to many undesired logical consequences such as 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, there is a dire need to develop methods that will help us analyse, monitor, and evaluate the content of (large) Knowledge Bases [8, 18, 9].

A significant effort in creating a large medical KB recently started in Babylon Health[6] using ontology matching and integration [14]. This medical KB is intended to support various services within Babylon like text annotation, understanding, 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 (including consistency and coherence) while others by services that depend on the KB. For example, text annotation and Named Entity Disambiguation ser-

[3] `https://www.snomed.org/`
[4] `https://www.meddra.org/`
[5] `https://meshb.nlm.nih.gov/`
[6] `https://www.babylonhealth.com/`

vices 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 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 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 statistics about using the pipeline to integrate many popular medical ontologies like SNOMED CT, 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 aligned[7] and, 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 CT international version and by its use some discrepancies were found in the Australian extension.

2 Ontologies and Ontology Matching

For brevity, 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 $\langle A \ \mathsf{rdfs{:}subClassOf} \ B \rangle$ 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 distinct IRI prefixes $p_1 \neq p_2$, $p_1{:}C$ and $p_2{:}C$ denote distinct classes. For an ontology $\mathcal{O}$ we use $\mathsf{Sig}(\mathcal{O})$ to denote the set of class names that appear in $\mathcal{O}$. Given an ontology $\mathcal{O}$ we assume that each class C in $\mathcal{O}$ has at least one triple of the form $\langle C \ \mathsf{skos{:}prefLabel} \ v \rangle$ and zero or more triples of the form $\langle C \ \mathsf{skos{:}altLabel} \ v_i \rangle$. For a given class C function $\mathsf{pref}(C)$ returns the string value v in the triple $\langle C \ \mathsf{skos{:}prefLabel} \ v \rangle$. An ontology is called *coherent* if every $C \in \mathsf{Sig}(\mathcal{O}) \setminus \{\bot\}$ 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 we assume 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

[7] `http://www.cs.ox.ac.uk/isg/projects/SEALS/oaei/2017/results/`

between $\mathcal{O}_1$ and $\mathcal{O}_2$ is a 4-tuple of the form $\langle C, D, \rho, n\rangle$, where $C \in \mathsf{Sig}(\mathcal{O}_1)$ $D \in \mathsf{Sig}(\mathcal{O}_2)$, $\rho \in \{\equiv, \sqsupseteq, \sqsubseteq\}$ 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 $C\ \rho\ D$ where the confidence degree is attached as an 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

In this section we briefly present the ontology matching and integration pipeline we designed for building a large KB [14] mostly through illustrative examples.

Example 1. Consider an ontology-based application that uses the SNOMED CT ontology $\mathcal{O}_{\mathsf{snmd}}$ as a KB. Although SNOMED CT is a large and well-engineered ontology some relevant medical information may be missing. For example, for the disease "Ewing Sarcoma" SNOMED CT only contains the axiom $\mathsf{snmd{:}EwingSarcoma} \sqsubseteq \mathsf{snmd{:}Sarcoma}$ and no relations to signs or symptoms. In contrast, the NCI ontology $\mathcal{O}_{\mathsf{nci}}$ contains the following axiom about this disease:

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

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

$$m_1 = \langle \mathsf{snmd{:}EwingSarcoma}, \mathsf{nci{:}EwingSarcoma}, \equiv\rangle$$
$$m_2 = \langle \mathsf{snmd{:}Fever}, \mathsf{nci{:}Fever}, \equiv\rangle$$

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

However, it is well known that due to differences in the structure of the two ontologies, the integration may introduce undesired consequences like unsatisfiable classes [7] or new subClassOf relations to the initial KB [5].

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

$$m_1 = \langle \mathsf{snmd{:}SoftTissueDisorder}, \mathsf{nci{:}SoftTissueDisorder}, \equiv\rangle$$
$$m_2 = \langle \mathsf{snmd{:}Epicondylitis}, \mathsf{nci{:}Epicondylitis}, \equiv\rangle$$

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

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

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

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

$\mathcal{O} = \mathsf{saturate}(\mathcal{O})$
Mappings $:= \emptyset$
for all matcher : Config.Align.Matchers **do**
  for all $\langle C, D, \rho, n\rangle \in \mathsf{matcher}(\mathcal{KB}, \mathcal{O})$ **do**
    Mappings := Mappings $\cup \{\langle C, D, \rho, n, \mathsf{matcher}\rangle\}$
  end for
end for
$\mathcal{M}_f := \emptyset$
$w = \oplus\{\mathsf{matcher}.w \mid \mathsf{matcher} \in \mathsf{Config.Align.Matchers}\}$
for all $\langle C, D, \rho, \_, \_\rangle \in$ Mappings such that no $\langle C, D, \rho, n\rangle$ exists in $\mathcal{M}_f$ **do**
  $n := \oplus\{n_i \times \mathsf{matcher}.w \mid \langle C, D, \rho, n_i, \mathsf{matcher}\rangle \in \mathcal{M}\}/w$
  if $n \geq \mathsf{Config.Align}.thr$ **then**
    $\mathcal{M}_f := \mathcal{M}_f \cup \{\langle C, D, \rho, n\rangle\}$
  end if
end for
$\langle \mathcal{O}', \mathcal{M}_f\rangle := \mathsf{postProcess}(\mathcal{KB}, \mathcal{O}, \mathcal{M}_f, \mathsf{Config})$
$\mathsf{Model}_{\mathcal{KB}} := \mathsf{analyse}(\mathcal{KB})$
$\mathsf{Model}_{\mathcal{KB}'} := \mathsf{analyse}(\mathcal{KB} \cup \mathcal{O}' \cup \mathcal{M}_f)$
$\mathsf{DiffModel} := \mathsf{diff}(\mathsf{Model}_{\mathcal{KB}}, \mathsf{Model}_{\mathcal{KB}'})$
$\mathsf{Report} := \mathsf{produceReport}(\mathsf{DiffModel}, \mathsf{Config.Expectations})$
return $\mathcal{KB} \cup \mathcal{O}' \cup \mathcal{M}_f$

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

To repair this issue the typical approach followed in literature removes 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 depth in [14] removes axioms from the new ontology. In this example, we can compute $\mathcal{KB}_2^{\mathsf{int}} := \mathcal{KB}_1^{\mathsf{int}} \setminus \{\mathsf{nci{:}Epicondylitis} \sqsubseteq \mathsf{nci{:}SoftTissueDisorder}\}$ and hence $\mathcal{KB}_2^{\mathsf{int}} \not\models \mathsf{snmd{:}Epicondylitis} \sqsubseteq \mathsf{snmd{:}SoftTissueDisorder}$ as desired. ♢

In addition, various services that are built on top of the KB may impose other types of requirements on the KB and integrating sources may negatively impact them.

Example 3. Assume that our SNOMED CT-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 CT ontology assigning meaning to the text. For example, the word "severe" would be associated to the SNOMED CT class Severe.

Assume now that NCI is integrated with SNOMED CT. NCI contains class SevereAdverseEvent with an alternative (synonymous) label "severe". Consequently, after the integration there are two different classes that can be associated

with the word "severe". The choice of which class to use may have significant impact on the application and the interoperability of services. ♢

Our ontology integration approach is given in Algorithm 1. 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 such as HermiT [3] as well as additional custom saturation rules. As the KB is loaded to a triple-store for scalable SPARQL query answering this step is meant to improve the 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 $\mathcal{O}$, it aggregates them using a different weight for each matcher (matcher.w), and finally removes those mappings that fall below a certain threshold (Config.Align.thr). As mentioned previously, newly introduced subClassOf relations between symbols of the current KB are eliminated and this is performed in method postProcess. This method is quite involved and details can be found in [14].

In the current paper we focus on the final analysis step of our integration pipeline. After the matching and mapping post-processing, a set of analysers are applied. 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 by 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 the metrics we are using into 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 includes metrics about the actual content of the KB like assessing its completeness. Many of these 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 subcategories.

KB Coherence 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 $\mathcal{KB} \not\models A \sqsubseteq \bot$. 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 coherence checking algorithms are based on approximate methods and techniques inspired by the DL-Lite language [1]. More precisely, the function analyse internally computes the following set:

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

where $\models_{\mathsf{rdfs}}$ denotes the entailment relation 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
```

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 $\mathsf{diff}_\Sigma(\mathcal{O}, \mathcal{O}')$) is defined as the set of axioms α satisfying: (i) $\mathsf{Sig}(\alpha) \subseteq \Sigma$, (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 $\mathsf{diff}_\Sigma(\mathcal{KB}, \mathcal{KB} \cup \mathcal{O} \cup \mathcal{M}_f) = \emptyset$ where $\Sigma = \mathsf{Sig}(\mathcal{KB})$. Computing logical differences between expressive, let alone large KBs, is very challenging if possible at all. Nevertheless, a highly optimised system, called LDiff-FAME has been implemented within the scope of this project as an adaptation of our system FAME [20, 19]. These systems are based on the notion of uniform interpolation which is similar to logical difference but LDiff-FAME is highly optimised to scale over large ontologies. As is shown in the evaluation section LDiff-FAME was able to compute logical differences between SNOMED CT versions.

LDiff-FAME has not yet been applied to the full scale of our integrated KB. Instead we used 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 $\mathsf{Leg} \sqsubseteq$

$\exists$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 The above metrics are based on well-defined notions from logics. In addition to those, a number of graph-based metrics can be identified to further analyse ontologies and comprehend 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 measures 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. Instead of counting how many times a node has more than one parent normalised with the cardinality of the graph [2] we also quantify the joins of these forks. First we define the set of least-common-subsumers (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:

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

   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:

```
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 identify parts of the KB in which large structural changes occurred after the integration.

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

Label integrity/ambiguity As motivated by Example 3 we wish to avoid having distinct classes sharing labels. Obviously this is impossible in general as language is inherently ambiguous. For example, SNOMED CT 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" of these classes[8] to disambiguate, however, this is still a hard problem for text annotation services. For this reason we have developed methods to measure and eliminate duplication as much as possible. Let $\mathbb{C} \subseteq \mathsf{Sig}(\mathcal{KB})$ be a set of types in $\mathcal{KB}$. For every $C \in \mathbb{C}$ we define the following set:

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

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 $\mathsf{amb\text{-}lab}(\mathcal{KB}) = \Sigma_{C \in \mathbb{C}} \sharp \mathsf{amb\text{-}lab}(C, \mathcal{KB})$.

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 integration 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

[8] By "types" we mean top-level classes which are commonly used to group classes into categories, e.g., the type of Leg is BodyPart and that of Malaria is Disease; types are defined by the KB engineer.

literature [16, 9]. Quantifying completeness is in general impossible since to do so one would need to know what is all 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 $R \in \mathsf{Sig}(\mathcal{KB})$ let $\mathsf{dom}(R) := \{C \mid \langle R\ \mathsf{rdfs{:}domain}\ C\rangle \in \mathcal{KB}\}$ and $\mathsf{ran}(R) := \{D \mid \langle R\ \mathsf{rdfs{:}range}\ D\rangle \in \mathcal{KB}\}$. Our function analyse computes the following sets:

1. **Relation usage:**

$$\begin{aligned}\mathsf{usage}(R) := \sharp\{\langle A\ R\ B\rangle \in \mathcal{KB} \mid \mathcal{KB} \models \{A \sqsubseteq C, B \sqsubseteq D\},\\ C \in \mathsf{dom}(R), D \in \mathsf{dom}(R)\}\\ \mathsf{usage}(C, R) := \sharp\{\langle A\ R\ B\rangle \in \mathcal{KB} \mid \mathcal{KB} \models \{A \sqsubseteq C, B \sqsubseteq D\}, D \in \mathsf{ran}(R)\}\end{aligned}$$

   The above measures are quite similar to the metric $\mathsf{freq}(R, C)$ defined in [9] but here we only count triples that fall within the "specified" use of the relation R—that is, within its 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 relation. We call this issue *drift* and measure it according to the following sets where $\langle A\ R\ B\rangle \in \mathcal{KB}$:

$$\begin{aligned}\mathsf{drift}_o(R) := \sharp\{\langle A\ R\ B\rangle \mid \mathcal{KB} \models A \sqsubseteq C \text{ for some } C \in \mathsf{dom}(R), \text{ and}\\ \mathcal{KB} \not\models B \sqsubseteq D \text{ for all } D \in \mathsf{ran}(R)\}\\ \mathsf{drift}_s(R) := \sharp\{\langle A\ R\ B\rangle \mid \mathcal{KB} \models B \sqsubseteq D \text{ for some } D \in \mathsf{ran}(R) \text{ and}\\ \mathcal{KB} \not\models A \sqsubseteq C \text{ for all } C \in \mathsf{dom}(R)\}\\ \mathsf{drift}_{so}(R) := \sharp\{\langle A\ R\ B\rangle \mid \mathcal{KB} \not\models A \sqsubseteq C, \mathcal{KB} \not\models B \sqsubseteq D \text{ for all}\\ C \in \mathsf{dom}(R) \text{ and } D \in \mathsf{ran}(R)\}\end{aligned}$$

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

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

   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 $\mathsf{extension}(\mathsf{owl{:}Thing}, \mathsf{skos{:}definition})$. 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.

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

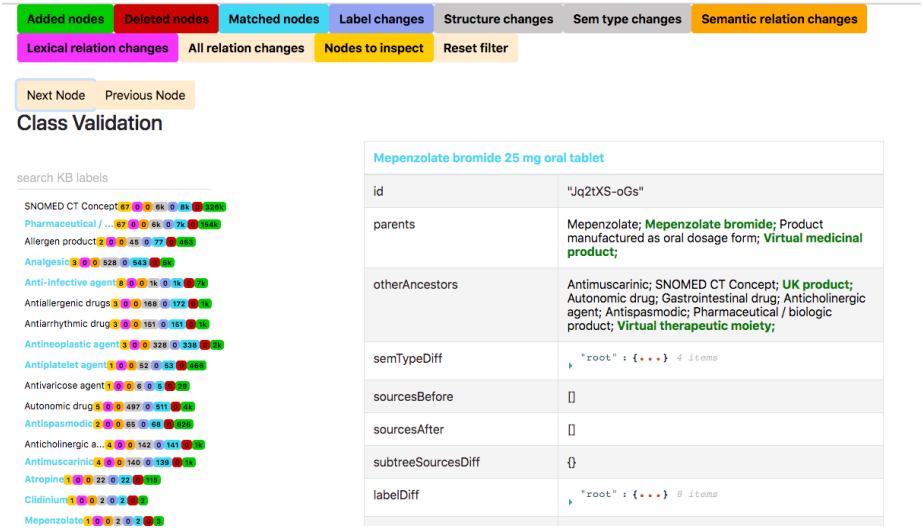

Fig. 1. KB and Class inspection

4.3 Metric Inspection and KB Verification

After all metrics are extracted, a diff operation is performed on several of them (line 19, Algorithm 1) in order to provide 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, coherence 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 which should not increase significantly. More formally, for $\mathcal{KB}, \mathcal{O}, \mathcal{M}_f$ the sets computed up to line 16 in Algorithm 1, we should have:

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

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

To further assist in inspecting the content of the KB a graphical tool has been developed at 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 resulting from 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

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

	SNOMED CT	NCI	MesSH	MedDRA	CTV3	ICD-10	Read2	FMA
$count(\mathsf{tang})$	118 120	12 529	7 950	8 248	10 092	0	0	0
$avg(\mathsf{tang})$	2.0	1.26	1.2	1.0	1.14	0	0	0
$avg(\mathsf{tang}_{\downarrow})$	27.0	1.9	0.8	0.3	0.33	0	0	0
$max(\mathsf{tang})$	32	11	8	1	9	0	0	0
$\mathsf{amb\text{-}lab}(\mathcal{KB})$	1 072	4 873	0	5	24 960	708	1 139	261

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 at Babylon Health. Several medical sources have been considered. As mentioned, determining whether two sources can be integrated "smoothly", monitoring the whole process at such a large scale, and fine tuning the parameters of the pipeline is not trivial. Initially, we used the tangledness and label ambiguity metrics to analyse the following popular medical data sources: SNOMED CT, NCI, MeDRA, MeSH, ICD-10, Read2, CTV3 (a successor of Read2), and FMA. The results are depicted in Table 1 where $count(\mathsf{tang}) = \sharp\{A \mid \mathsf{tang}(A) > 0\}$.

As can be seen SNOMED CT 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 was followed. 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 results we selected SNOMED CT as our seed ontology and used Algorithm 1 to build a medical KB. On top of SNOMED CT 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

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

	SNOMED CT	+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	∅	∅	∅	∅
Ambiguity	1 072	5 768	9 207	9 811
Ambiguity-Repaired	180	1 266	1 892	2 078

step are depicted in Table 2. As can be seen our postProcess method ensures that no new subClassOf axioms are introduced between the symbols of the seed ontology (LDiff row) something that does not happen when using other ontology matching frameworks (see also 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 helped 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 an implementation 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.

We have also considered the Canadian and Australian extensions of SNOMED CT, denoted by $\mathcal{O}^{ca}_{\mathsf{snmd}}$ and $\mathcal{O}^{au}_{\mathsf{snmd}}$, respectively. These extensions add more labels, country specific concepts, and provide additional local variations and customisations relevant to healthcare communities of these countries. Ideally we should have $\mathsf{diff}_{\Sigma}(\mathcal{O}_{\mathsf{snmd}}, \mathcal{O}^{*}_{\mathsf{snmd}}) = \emptyset$ for $* \in \{au, ca\}$ and $\Sigma = \mathsf{Sig}(\mathcal{O}_{\mathsf{snmd}})$. 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 CT and any new IRI is a new classes not in SNOMED CT.

We used LDiff-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 CT with additional labels and classes without affecting its structure; this is not the case for the Australian extension in which case

there are 67 strongest inferred axioms in the above set. One example is a class subsumption $A \sqsubseteq B \in \mathcal{O}_{\mathsf{snmd}}$ which in $\mathcal{O}^{au}_{\mathsf{snmd}}$ is $B \sqsubseteq A$. How to properly deal with Australian SNOMED CT remains part of future work. SNOMED International was made aware of these cases.

6 Conclusions

We have presented a 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 CT, NCI, FMA as well as labels from the CHV vocabulary. To the best of our knowledge, no ontology integration at this scale has been performed in the past; all previous matching efforts have used much smaller versions of them and the computed mappings were never actually used to merge the ontologies.

To help us decide which sources to use and how to configure our integration pipeline a set of analysers were implemented. They were 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 have presented our results in applying several of these metrics on well-known medical data sources that are currently under investigation at Babylon. However, our results showed interesting properties about them like ambiguity of their labels and multi-hierarchical nature of their structure. The metrics and verification mechanism assisted in fixing several errors in the pipeline, assessing the success of the integration and fine tuning it. Finally, the LDiff-FAME system helped determine that the Australian extension of SNOMED CT 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) 53, 245–269 (2014)
4. Golbeck, J., Fragoso, G., Hartel, F.W., Hendler, J.A., Oberthaler, J., Parsia, B.: The national cancer institute's thésaurus 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., Walther, 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., Atemezing, G., Poveda-Villalón, M., Vatant, B.: Linked open vocabularies (LOV): A gateway to reusable semantic vocabularies on the web. Semantic Web 8(3), 437–452 (2017)
16. Vrandecic, D.: Ontology evaluation. In: Handbook on Ontologies, pp. 293–313. Springer (2009)
17. Zaveri, A., Kontokostas, D., Sherif, M.A., Bühmann, 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 $\mathcal{ALCOIH}\mu^+(\nabla, \sqcap)$-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)
20. Zhao, Y., Schmidt, R.A.: FAME: An automated tool for semantic forgetting in expressive description logics. In: Proceedings of the 9th International Joint Conference On Automated Reasoning (IJCAR). Lecture Notes in Artificial Intelligence, vol. 10900, pp. 19–27. Springer (2018)