Logical Comparison over RDF Resources in Bio-Informatics

S. Colucci\textsuperscript{a,}\textsuperscript{*}, F.M. Donini\textsuperscript{b}, E. Di Sciascio\textsuperscript{a}

\textsuperscript{a}Politecnico di Bari, Via Orabona 4, 70125 Bari, Italy
\textsuperscript{b}Università della Tuscia, Via S. Maria in Gradi 4, 01100 Viterbo, Italy

Abstract

Comparison of resources is a frequent task in different bio-informatics applications, including drug-target interaction, drug repositioning and mechanism of action understanding, among others. This paper proposes a general method for the logical comparison of resources modeled in Resource Description Framework and shows its distinguishing features with reference to the comparison of drugs. In particular, the method returns a description of the commonalities between resources, rather than a numerical value estimating their similarity and/or relatedness. The approach is domain-independent and may be flexibly adapted to heterogeneous use cases, according to a process for setting parameters which is completely explicit. The paper also presents an experiment using the dataset Bioportal as knowledge source; the experiment is fully reproducible, thanks to the elicitation of criteria and values for parameter customization.

1. Introduction and Motivation

The need for comparing resources\textsuperscript{1} is shared by different applications in bio-informatics. Usually, the output of such a comparison process is directly a number, that can be used either (i) as an absolute measure of similarity, or (ii) as a relative measure when a set of resources \(b_1,\ldots,b_n\) must be ranked with respect to how similar they are to a given resource \(a\). As an example, among methods for predicting drug-target\textsuperscript{2} interactions, those based on the evaluation of similarity among drugs (and targets) are recognized as the best performing ones\textsuperscript{3}. The work by Ding \textit{et al.} classifies similarity measures for drugs according to three different dimensions of comparison: chemical structure, side-effects, and gene-expression (i.e., the similarity is computed from the response of gene expression to drugs). Such a distinction reflects the structure of the data sources available to describe drugs, whichcover—in general—different features. Concerning target similarity, proposed measures are classified into the following three types: sequence-based similarity, similarity based on the protein-protein interaction (PPI) network and Gene Ontology (GO) semantic similarity.

Also in drug repositioning\textsuperscript{3} methods based on the evaluation of similarity between resources have been proposed. Some work\textsuperscript{4} uses the similarities between the prescribed drugs for a specific disease to infer repositioning candidates. The information on drug targets, drug interactions, substructures and side effects is extracted from DrugBank\textsuperscript{5} and used to generate a so-called drug-similarity network. A node in the drug-similarity network may represent the drug itself, a target, another drug interacting with the one originating the network, a substructure, or a side effect. Targets, drug interactions, substructures and side effects are included in the network only when shared by two or more drugs.

Zhang \textit{et al.}\textsuperscript{4} propose a drug repositioning method that uses a drug similarity network, a disease similarity network, and known drug-disease associations to explore the potential associations among other unlinked drugs and diseases. The measure of similarity between drugs is computed by combining three measures deriving from the comparison of three types of drug information, i.e., chemical structure, target protein, and side effect. Analogously, three types of disease information—

\textsuperscript{*}Principal Corresponding Author

\textit{Email addresses: simona.colucci@poliba.it} (S. Colucci),
\textit{donini@unitus.it} (F.M. Donini),
eugenio.disciacpio@poliba.it (E. Di Sciascio)

\textsuperscript{1}In this paper, by resource we mean an object of interest for the comparison, described according to a reference knowledge model. If the knowledge model is an RDF namespace, we usually speak about RDF resources

\textsuperscript{2}Each drug has to bind to a target molecule in order to exercise its effects. Most drug targets are proteins.

\textsuperscript{3}The search for potential additional uses for existing drugs.
phenotype, ontology, and disease gene—are considered in the computation of diseases similarity.

The work by Mathur et al. [5] addresses the measurement of disease similarity as an important biomedical task. A measure of similarity is proposed, computed by translating information between biomedical ontologies and quantifying similarity between terms in such ontologies.

Other studies [6] focus on the use of drug-similarity networks for understanding the mechanism of action of drugs. In particular, a network is built by connecting drugs with deletion strains on the basis of resistance or sensitivity relationships. Then, two drugs are considered similar if the number of relationships of resistance or sensitivity to the same deletion strains overcomes a given threshold.

In all applications recalled so far, similarity is expressed by a numerical value, computed according to different measures. Presenting the result of the comparison as a numerical value has some appealing characteristics: first of all it is concise, and secondly, it allows a ranking between several comparisons, thanks to the total order between numbers. In general, the purely numerical result works well when the comparison focuses on one specific facet of a resource, and does not treat it as a whole. This is inherent to this method: since it measures how much the two resources are similar, without saying why they are similar, when several characteristics are compared, or even all of them, numerical methods cannot explain how each facet contributes to the overall numerical result—apart from showing a mathematical formula, in which the knowledge used to weight each contribution is hidden inside tuned parameters. This causes two pairs of resources (drugs, targets, diseases and so on) with the same value of similarity to be considered equally similar, despite the inherent peculiarities that may emerge in the comparison process.

Yet, there are several applications which would benefit much more from a description of features shared by two resources, than from a numerical value estimating their similarity. As an example, consider the problem addressed by Kurtz et al. [7]: the retrieval of similar medical images described with semantic annotations. This task requires a high level of accuracy, and asks for more informative retrieval responses. Also, the design of ad-hoc search engines for retrieving biomedical-specific resources [8] testifies how such resources may need special handling and works as an example of an application requiring retrieval results as explicit as possible.

Generally speaking, a mechanism for comparison should:

- explain as much as possible the reason for returned similarity values;
- allow applications to choose resource features to be compared, possibly according to multiple data sources;
- treat each resource to be compared as a unique item, described according to heterogeneous facets.

In this paper, we apply a general method, whose theory has already been fully developed in a previous paper [9], which automatically compares the data about two resources and returns the features they share. The method only requires resources to be uniquely and unambiguously modeled in Resource Description Framework (RDF) [10], which is an almost implicit requirement, given the level of maturity reached by the Linked (Open) Data Initiative (LOD) [11]. In fact, we can assume the availability of a unique data source in which resources are all modeled in RDF: the so-called “Web of Data”. Even though this availability comes at no warranty for quality of data [12], a significant level of interoperability is ensured for data published according to the LOD initiative. We exploit the semantics of RDF to make information-rich the results of comparison: the method is able to return an RDF description of the features shared by the two resources.

We show the potential of our method with reference to the comparison of drugs. We choose BioPortal [13] as endpoint to access the data source in our example. BioPortal represents by itself a significant example of a cloud of Linked Data, and is therefore useful to show the benefits of our method.

We immediately clarify the boundaries of our analysis: (i) first of all, in this paper, we compare resources at the data level, not at the ontology level. That is, we limit our comparison to data in RDF describing the two resources, without considering properties that are not explicitly stated in RDF, but that could be inferred from the semantics of RDF-S or RDF-serialized OWL statements. The interested reader can find a more detailed comment in the next section; (ii) we take the RDF data as is, without evaluating their quality.
The paper is organized as follows. In the next section, we position our work w.r.t. the literature on resource similarity and available tools to drug comparison. In Section 3 we describe the general method for resource comparison and provide all background knowledge required to make the paper self-contained. In Section 4 we show how to flexibly adapt our method to the chosen use case: the comparison of drugs. Section 5 reports on results and lessons learned from the application of the method to drug comparison. Section 6 closes the paper and summarizes future work.

2. Paper Positioning

The literature on resource comparison is characterized by a huge number of proposals introducing metrics which compute the similarity and/or the relatedness of pairs of resources. The heterogeneity of such measures caused some attempts of unification under a common framework in past research [14][15].

The work by Petersen et al. [16] reviews a number of such measures, originally proposed in the domain of Natural Language Processing, and shows how to adapt some of them to the biomedical domain. Notably, similarity is managed as a special case of relatedness. The contribution of their work is threefold: i) it systematically classifies existing relatedness measures in three categories: path based, information content based and context vector based; ii) it raises the need to evaluate relatedness on the basis of the information embedded in a knowledge model (SNOMED-CT); iii) it provides a corpus, the Mayo Clinic Corpus of Clinical Notes, and a benchmark for term pairs similarity, thoroughly used as reference in the literature so far.

Path-based measures rely on the evaluation of the length of paths connecting resources to compare in a hierarchical model describing the domain of interest. Most reviewed measures adopt Wordnet taxonomy (with specific reference to nouns) as knowledge model and consider only paths corresponding to IS-A relationships. Petersen et al. highlight the main limitation of purely path based measures: the degree of semantic similarity implied by a single link is not consistent, because it does not take into account the information content embedded in each link.

The approach by Resnik et al. [17] addresses such issue by introducing a definition of Information Content of concepts in a hierarchy, which is a measure of the specificity of a concept, calculated based on the frequency of the occurrence of that concept in a large corpus of text. The similarity between two concepts is measured by the Information Content of their lowest common subsumer (lcs) in the hierarchy. Other approaches [18][19] propose different measures based on the information content of both concepts to compare (i.e., not only of their lcs).

A different approach is grounded on the representation of words to compare as context vectors [20]. In this case, the source of the information for the context vectors is a raw corpus of text, and not the paths found between concepts in an ontology.

Petersen et al. [16] show how such measures may be easily adapted to the biomedical domain by relying on the information content modeled in SNOMED-CT.

In particular, they propose a path-length measure for SNOMED-CT and an adaptation to SNOMED-CT of a path-based measure, specific for Wordnet, and proposed by Leacock and Chodorow [21]. In both cases, SNOMED-CT is used to compute the similarity between two concepts by counting the numbers of nodes on the shortest path between them in the is-a hierarchy. Notably, the authors themselves outline the need for measures both vocabulary-independent and going further IS-A relationships.

Moreover, when adapting to SNOMET-CT both measures based on information-content and on context-vectors based, the knowledge embedded in the ontology is not fully exploited. In fact, in the first class of measures, SNOMED-CT is used as the source of concepts, whose term frequency is counted in the Mayo Clinic Corpus of Clinical Notes. Also, context-vectors are derived from word vectors by counting the occurrence of SNOMED-CT concepts in the same corpus.

More recently, a different ontology-based measure has been proposed [22] which explores SNOMED-CT not only for searching the common ancestors but also non-shared super-concepts, as a degree of dissimilarity. The measure is defined as the ratio between the amount of non-shared knowledge and the sum of shared and non-shared knowledge. Again, the only relationship analyzed in the hierarchy is IS-A.

Apparently, none of the proposed measures really reads through the ontology and investigates possible semantic-based reasons for relatedness. Even when a concept hierarchy is exploited (with only reference to IS-A relationships), relatedness is measured in accordance to a numerical value. Yet, the need to better capture the knowledge implicitly or explicitly modeled in structured resources has been claimed [15].

On the contrary, our approach to comparison aims...
at exploiting a knowledge model to deduce a description of the features shared by the two resources. Such a description works as an explicit explanation of relatedness, which, to the best of our knowledge, none of the approaches proposed so far is able to return. Conversely, we could use such a description to define a numerical measure of similarity, even though this is out of the scope of this article.\footnote{A first attempt in this direction has been already made [23].}

This description is computed in terms of *Common Subsumer* (CS) of the resources to compare, through the process we detail hereby, which is flexible w.r.t. to the adopted dataset and completely reproducible. Notably, by construction, the CS may involve all relations used in the dataset, and not only those mapping IS-A relationships. Moreover, the approach we propose allows for choosing the dataset to adopt in a flexible fashion, even though it requires such a dataset to be written in RDF and accessible at a SPARQL endpoint. In other words, our approach is independent on the dataset from a theoretic point of view. Nevertheless, it embeds some special features for improving performance and fitness to the problem and the domain at hand. In particular, the whole set of triples in the dataset may be cropped to extract the portion of resource description useful for comparison. We also point out that the approach supports the adoption of multiple RDF dataset, which has been addressed as a key requirement in biomedical domain [24].

Common Subsumers were firstly proposed in Description Logics [25] to compute commonalities between conceptual descriptions. Research focused on computing so-called *Least Common Subsumers* (LCS) in various Description Logics, most of which can now be described as variants of OWL-EL (see the work of Zarrieß and Turhan [26] as one of the most recent ones). The problem of computing the LCS in ontologies with languages more expressive than OWL-EL is proved to be quite difficult [27], and to the best of our knowledge, no algorithm for computing LCS w.r.t. RDF-S semantics—to say nothing of OWL2—has been devised. As for the size of the LCS, it has been proved that it is worst-case exponential even for a small sublanguage of OWL-EL [28]. Our work here completely differentiates from the above research, since we deal only with explicit RDF data about a resource, and not with implicit properties that might be inferred when considering the semantics of RDF-S and (serialized) OWL statements. Our method computes a CS in quadratic time and space (see next section), and in this paper we prove that even its non-optimized implementation yields a manageable and meaningful CS in feasible execution times.

2.1. Tools for Visual Drug Comparison

The interest in drug comparison is also testified by the availability of some commercial and/or online tools devoted to the parallel visualization\footnote{The tools qualify themselves as “Visual” in their presentations. However, we note that maybe a more proper description for them would be ‘textual’.} of selected drug features.

The tool Lexicomp by Wolters Kluwer embeds a module\footnote{http://www.wolterskluwercdi.com/lexicomp-online/user-guide/tools-drug-comparisons/} for visual comparison of selected features of groups of drugs (2 to 4 items). The Drug Comparison module shows a table with drugs in columns and selected features in rows. Table cells include features explicitly stored in the database underlying the tool: no inference is made on such an information and common features are not highlighted for the user, who has to look through the table and manually infer the information she needs. As an example, by comparing “Fluconazole” to “Voriconazole”, the tool shows that the former has “Nausea, Abdominal Pain and Vomiting” as frequent side effects, while the latter has “Chills, Fever, Nausea, Skin Rash and Vomiting”. Lexicomp neither highlights that “Nausea and Vomiting” are common features, nor infers implicit side effects, like “stomach upset”.

The online tool Iodine\footnote{http://www.iodine.com/compare} provides an interface to compare drugs (up to 4 items), and returns a short abstract which combines all main drug descriptors, such as medical uses and side effects, without classifying them at all. The shared features are not highlighted and no inference is made on implicit commonalities.

On the contrary, our approach is able to return an explicit description of drugs commonalities, also inferring features implicitly embedded in the knowledge model used to describe drugs. In the rest of the paper, we show the potential of our approach to resource comparison, both w.r.t. research proposals and to available tools.

3. The Method

In order to make the paper self-contained, in this section we summarize notions and previously published results that we use in Section 4. In the next subsection, we recall basic notions about RDF and set up our criteria for choosing a subset of triples relevant for comparing two
resources. Then in Subsection 3.2 we briefly summarize a recent proposal about logical comparison of resources in RDF [9]. The acquainted reader may skip this section.

3.1. Rooted RDF-graphs

URIs of RDF resources often refer to namespaces—available worldwide—which are abbreviated as a prefix in the resource URI. Prefixes are paired to namespaces in declarations which appear at the beginning of the serialization in Turtle [29] of RDF datasets. In our examples, we make use of the namespaces whose prefixes are listed in Figure 1 and of Turtle syntax.

RDF is based on triples \( t = \langle s, p, o \rangle \) (subject-predicate-object) [10] each triple expressing a fact. For example, the following triple \( t_1 \):

\[
\text{ndfrt}:N0000145918 \\
\text{ndfrt}:may\_prevent \\
\text{ndfrt}:N0000002278.
\]

in Bioportal expresses the fact that Aspirin (coded by the URI ndfrt:\text{N0000145918}) may prevent pain (coded by the URI ndfrt:\text{N0000002278}). Recall that the prefix name ndfrt is defined in Figure 1.

A set of triples is usually referred to as a graph, where subjects and objects are the labels of nodes, which are linked by arcs, labeled by predicates. Referring to the well-known representation of graphs as Relational structures (see for example [50] p.7 and [51] p.316), in general the triple \( \langle s, p, o \rangle \) expresses the fact \( p(s, o) \). For example, the triple \( t_1 \) above could be interpreted as the ground fact \textit{may\_prevent}(Aspirin, pain). This correspondence between a set of triples and a graph can be considered at the basis of numerical methods: for instance, Kernel methods [32] compare random walks on graphs and interpret the number obtained as a property on the meaning of the graphs—like a similarity between resources. However, considering a set of triples as a (usual) graph is incorrect, for at least the following two reasons.

First, since every element of a triple can appear in any position in another triple, the set of triples is more correctly interpreted in Higher-Order Logic, since a resource used in a predicate position in a triple can appear in the subject position in another triple. For example, referring to Bioportal, the predicate of the following triple

\[
\text{bridgmodel:DefinedProcedure.methodCode} \\
\text{skos:example} \\
\text{"veni puncture"}.
\]

is involved as subject in the next triple, taken from the namespace SKOS:

\[
\text{skos:example} \\
\text{rdfs:subPropertyOf} \\
\text{skos:note}.
\]

Such a subtlety is almost never considered in present algebraic methods for computing similarities—e.g., Kernel methods—which cannot use the label qualifying a value (the predicate in the triple) as a value itself.

The second reason is that RDF admits blank nodes, which are existential variables whose scope is the file they appear in. Blank nodes are different from Database null values, since they can be interpreted as any constant, even one not already occurring in the RDF file. They are useful when some known resources must be linked through some resource whose IRI is unknown. We remark that also blank nodes are problematic for numerical similarity methods [33], since blank nodes are simply considered as missing data. Remarkably, Bioportal eliminated blank nodes by skolemizing[11] them to fictitious IRIs.

From now on, we distinguish RDF-graphs from usual graphs, and we denote blank nodes in examples and definitions with the last letters of the alphabet: \( w, x, y, z \).

Colucci et al. [9] adapted the basic notions of Graph Theory to RDF-graphs, with some definitions we briefly recall here to make the paper self-contained. First, an RDF-path from \( r \) to \( s \) is a sequence of triples \( t_1, ..., t_n \) in which the subject of \( t_i \) is \( r \), either the predicate or the object of the triple \( t_i \) is \( s \), and for \( i = 1, ..., n - 1 \), either the predicate or the object of \( t_i \) is the subject of \( t_{i+1} \). A resource \( r \) is RDF-connected to a resource \( s \) if there exists an RDF-path from \( r \) to \( s \). Observe that paths (and connections) are always oriented, since triples are so. The length of such an RDF-path is \( n \), and the RDF-distance between two resources is the length of the shortest RDF-path between them. Also, the RDF-distance between a resource \( r \) and a triple \( t \) is the shortest RDF-distance between \( r \) and the subject of \( t \)—in particular, triples which \( r \) is the subject of, have zero-RDF-distance from \( r \) itself, as expected.

---

[10] Strictly speaking, a triple in RDF files must be written as \( s \ p \ o \).

[11] Given a formula \( \phi \) with existentially quantified variables, its skolemization \( S(\phi) \) is another formula in which every occurrence of a variable \( x \) has been replaced with a constant \( c_x \). It is well known that \( \phi \equiv \psi \) if and only if \( S(\phi) \equiv S(\psi) \), but \( \phi \) and \( S(\phi) \) are not equivalent formulas.
Finally, a resource $r$ is connected to a triple $t$ if $r$ is RDF-connected to the subject of $t$.

Given a resource $r$, a crucial choice of Semantic Web applications is which triples are pertinent for $r$. Large data repositories as Bioportal can count billions of triples, and obviously an application has to balance between completeness (i.e., considering as many triples as possible) and a timely response. However, using the RDF-distance as the only criteria would be a too naive choice. To explain the problem, suppose that an application processing $r$ limits to all and only the triples $r$ is the subject of (that is, triples whose RDF-distance from $r$ is 0). Such a choice would be both too large—since it includes (usually uninteresting) triples regarding an instantiation of this pattern in Bioportal is depicted in Figure 2 (as RDF-distance) from $r$, also meaningful. Frequently, applications do not make explicit such choice criteria, making their experiments non-reproducible.

Colucci et al. explicit their choice criteria as

1. data sources: which datasets (one or more) triples are drawn from for the purpose of the comparison;
2. RDF-distance: exclude triples which are “too far” from $r$—i.e., those whose RDF-distance from $r$ exceeds a given threshold;
3. stop-patterns\footnote{Note that RDF-S or OWL deduction would be of no help here. For RDF-S, there is no RDF-S Rule involving ndfrt:may_prevent and rdfs:subClassOf, while OWL cannot even give meaning to such triples, since the object of ndfrt:may_prevent is an individual, while the subject of rdfs:subClassOf should be a class.} exclude triples which fit a given pattern $\langle s p o \rangle$, where any number of variables are instantiated by an RDF resource (an IRI or a blank node or a literal);
4. connectedness: there must be an RDF-path from $r$ to the subject of each chosen triple.

Such a portion of triples, centered around $r$, is called a rooted RDF-graph (from now on r-graph), denoted by $\langle r, T_r \rangle$. For instance, in Bioportal, choosing an RDF-distance of 1, and using the stop-patterns described in Appendix A one would represent Heparin (ndfrt:N0000146860) through the rooted RDF-graph depicted in Figure 3\footnote{After stop-words in Information Retrieval search algorithms.}.

The reader may find in Appendix B the complete serialization in Turtle of the r-graph shown in Figure 2. Here, we describe only the two paths highlighted in red dashed ellipses in Figure 2 (for the sake of example):

a) at RDF-distance 0, this path connects Heparin to the resources Thromboembolism (ndfrt:N0000002934) and Venous Thrombosis (ndfrt:N0000004074) through the property ndfrt:may_prevent; at RDF-distance 1 from Heparin, it connects both Thromboembolism and Venous Thrombosis to the resource Thrombosis (ndfrt:N0000002936) through the property rdfs:subClassOf.

b) this path connects Heparin to the resource Antithrombin Activators (ndfrt:N0000009960) through the property

\begin{figure}
\centering
\includegraphics[width=\textwidth]{prefixes}
\caption{RDF declaration of prefixes (and the namespaces they refer to) used in this paper.}
\end{figure}

\begin{itemize}
\item @prefix skos: <http://www.w3.org/2004/02/skos/core#> .
\item @prefix ndfrt: <http://purl.bioontology.org/ontology/NDFRT/> .
\item @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
\item @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
\item @prefix owl: <http://www.w3.org/2002/07/owl#> .
\item @prefix vrank: <http://purl.org/voc/vrank#>.
\item @prefix owl: <http://www.w3.org/2002/07/owl#>.
\item @prefix skos: <http://www.w3.org/2004/02/skos/core#>.
\item @prefix ndfrt: <http://purl.bioontology.org/ontology/NDFRT/>.
\item @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
\end{itemize}
ndfrt:hasMechanismOfAction and Antithrombin Activators to the resource Enzyme Activators (ndfrt:N0000000231) through the property rdfs:subClassOf.

We stress the fact that the first three choice criteria (dataset, distance and stop-patterns) can be parameterized for the particular application at hand. This is because we are proposing here a general service for comparing resources, not a single application.

We observe also that more generally, our framework could check boolean combinations of stop-patterns, although in this paper we are not going to make use of this feature. Moreover, we may combine triples coming from different datasets when needed—for example, an RDF-path starting from Bioportal with a triple using a predicate $p$ may continue with a triple from a SNOMED dataset where $p$ is qualified by other triples. The choice about which datasets to combine triples from is parametric in our framework.

RDF is equipped with a model-theoretic semantics \[10\], which is straightforward but for the fact that to accommodate Higher-Order facts, every resource $r$ is interpreted both as an individual—its actual IRI if $r$ is not a blank node—and mapped to a predicate over resources, which are again interpreted in this way. Thanks to such a semantics, a notion of deduction between sets of triples is established, which is well-defined although simple. For instance, in Bioportal (see Fig-
ure 3), given the facts that identifier #RID23302 is a rdf:type #nerve_metaclass, and that the latter is a rdfs:subClassOf #neuraxis_metaclass, the fact that identifier #RID23302 is also a rdf:type #neuraxis_metaclass is a correct deduction.

3.2. Common Subsumers in RDF

Our general method for comparing resources \( r, s \) starts from two r-graphs \( \langle r, T_r \rangle, \langle s, T_s \rangle \), whose triples \( T_r, T_s \) have been extracted from a data repository with the criteria explained in the previous section. We define an auxiliary function \( \tau(u, v) \) over pairs of RDF terms such that \( \tau(u, v) = u = v \) if \( u = v \), and otherwise \( \tau(u, v) \) is a new blank node which is one-one with the pair \( u, v \). Terms appearing in pairs may be IRIs, literals or blank nodes. The Least Common Subsumer (LCS) of \( \langle r, T_r \rangle, \langle s, T_s \rangle \), \( \langle s, T_s \rangle \) is another r-graph \( \langle x, T_x \rangle \) such that:

1. \( x = \tau(r, s) \)—so, \( x \) is a blank node unless \( r \) and \( s \) coincide;
2. \( x \) is RDF-connected to every triple in \( T_x \), and
3. every triple in \( T_x \) is of the form \( \ll < y_i, w_i > \tau(w_j, r_j), \tau(z_r, z_j) \gg \), where both \( \ll y_i, w_i, z_r, z_j \gg \in T_r \) and \( \ll y_i, w_i, z_r, z_j \gg \in T_s \).

Colucci et al. prove that such an LCS is unique (up to blank nodes renaming), is idempotent, commutative and associative. Associativity—that is, \( LCS(r, LCS(s, t)) = LCS(LCS(r, s), t) \) if we leave triples implicit—is of particular importance, since it says that when there are three or more resources, their LCS is the same, no matter which pair of resources one starts the comparison from. Observe that the LCS is another r-graph, rooted in a node that represents an abstraction of \( u, v \). Hence one can compare this r-graph too, iterating the process. Contrast this characteristic with numerical methods, that yield always a number that cannot be further compared with another resource.

Moreover, Colucci et al. prove that the LCS computed by means of the function \( \tau \) above coincides with the strongest common logical consequence of both \( \langle r, T_r \rangle \) and \( \langle s, T_s \rangle \). Intuitively, this says that the LCS is the most specific set of properties \( r \) and \( s \) share—no irredundant triple can be added to \( T_s \) without losing deducibility from either \( \langle r, T_r \rangle \), or \( \langle s, T_s \rangle \), or both.

Colucci et al. prove also that, given two r-graphs \( \langle a, T_a \rangle \) and \( \langle b, T_b \rangle \), a representation of their Least Common Subsumer under Simple Entailment has size limited by \( |T_a| \cdot |T_b| \) and can be computed in time \( O(|T_a| \cdot |T_b|) \). This causes any strategy for improving performance to be aimed at the reduction of \( |T_a| \) and \( |T_b| \).

Notably, for real applications, the LCS may contain too many triples which, although logically implied by both \( \langle r, T_r \rangle \) and \( \langle s, T_s \rangle \), provide little information. For example, a triple \( \ll y_i, w_i, z_r, z_j \gg \), saying that resource \( a \) belongs to an unknown class \( y \), is of little information—although true—since every resource is of some type in RDF. We name these triples uninformative triples, and we eliminate them from the comparison result. What we obtain is a—no more Least—Common Subsumer, containing only the most informative triples which are deducible from both r-graphs.

4. Drug comparison

We address the comparison of drugs as a use case for our method to compare RDF resources. Recall (see Section 3.2) that the method allows users to customize the following parameters:

1. the datasets triples are extracted from;
2. the maximum RDF-distance of selected triples from r-graph root;
3. a list of stop-patterns.

As for the first parameter, we chose the dataset Bioportal for our experiments, since it collects in RDF most of the knowledge formalized so far about biological and medical facts. Thus, it represents a wide source of information to describe drugs. In order for such information to be significant to our purpose (comparing drugs), we
need to set more criteria for selecting triples from Bio-
portal, according to the second and the third parameter
above.

For a matter of presentation of results, we here refer
to an RDF-distance equal to 1, without loss of general-
ity.

We analyzed the triples returned according to the first
two parameters and identified a set of patterns we con-
sider irrelevant w.r.t. the objective of comparing drugs.
Thus, we set them as stop-patterns, in order to exclude
them from the r-graphs of resources to compare. The
complete set of stop-patterns used in our experiments is
given in Appendix A. We here only describe the criteria
at the basis of their selection.

In particular, we exclude triples for one of the following
reasons:

- some properties are not useful for the comparison,
because two different drugs may never share their
values. Such properties include, among others,
textual descriptions (e.g., ndfrt:STATUS), labels
(e.g., skos:prefLabel), and identifiers of drugs
(e.g., ndfrt:VUID).

- some triples suffer from modeling issues in Bioporo-
tal. As an example, consider the triple \( t_1 \) in Section
3.1. In Bioportal, it is also present the triple \( t_2 \):

\[
\text{ndfrt:N0000002278} \quad \text{ndfrt:may_be_prevented_by} \quad \text{ndfrt:N0000145918}
\]

(which says that “pain may be prevented by As-
pirin”). Given that \( t_1 \) and \( t_2 \) convey the same in-
formation, we identified triples whose property is
\text{ndfrt:may_be_prevented_by} as stop patterns, in
order to exclude redundant triples like \( t_2 \).

- some triples match patterns too generic to be sig-
ificant in a comparison: the fact that two
 drugs share such patterns is irrelevant w.r.t. to
finding their similarities. Examples of such patterns are:
\( p=\text{umls:hasSTY} \) and \( o=\text{umls:sty/T047} \) (stating that a given resource
has semantic type “Disease or Syndrome”) or
\( p=\text{rdf:type} \) and \( o=\text{owl:Class} \).

The r-graphs of the two resources to compare do not
include triples matching any of the above described
patterns.

Recall (Section 3.2) that the LCS of a pair of re-
sources may include triples which are not informative
w.r.t. the objective of finding shared features. Thus, the
computation of (not-least) Common Subsumers exclud-
ing such triples is more useful to our aim. The full list of
uninformative triples used in the computation is avail-
able in Appendix A.

We here show the application of our method
to the comparison of two common drugs: Hep-
arin (URI \text{ndfrt:N0000146860}) and Ardeparin (URI
\text{ndfrt:N0000022083}). Both drugs have to be modeled
as r-graph, setting the stop-patterns as described above.
The r-graph of Heparin is depicted in Figure 2.

If the triples described in Appendix A are set
as uninformative, a CS of the pair (Heparin, Arde-
parin) is the one shown in Figure 4 and fully re-
ported in Turtle serialization in Appendix B. We
here just describe, for the sake of example, the
path highlighted in triangle \( a \) and zoomed in Fig-
ure 5. This path encloses important common fea-
tures. First, both drugs are connected to the resources
Thromboembolism (ndfrt:N0000002934) and Ve-
nous Thrombosis (ndfrt:N0000004074) through the
property \( \text{ndfrt:may_prevent} \); also, both Throm-
boembolism and Venous Thrombosis are connected
to the resource Thrombosis (ndfrt:N0000002936)
through the property \text{rdfs:subClassOf}. Second, Hep-
arin and Ardeparin are connected through the property
\( \text{ndfrt:may_prevent} \) to two resources that, although
different (their CS is a blank node), are both classified
as Thrombosis.

5. Evaluation

In this section, we show the results of a thorough ex-
perimentation of our approach with a two-fold aim. On
the one hand, we demonstrate the feasibility of the pro-
posed approach in terms of execution times and its in-
dependence of the input dataset. To this aim, we com-
pute the Common Subsumers of 300 pairs of resources
randomly selected from two different datasets hosted by
org/ontology/SNOMEDCT/) and NDFRT (http://
purl.bioontology.org/ontology/NDFRT/). We
report on such experiments in Section 5.1.

On the other hand, we show the informative potential
of our approach by comparing our results to the ones
returned by a numerical method (selected among the the
wide set of available ones, without loss of generality) in
terms of explanation. The comparison is discussed in
Section 5.2.

Lessons learned from the experimentation are gath-
ered in Section 5.3.
Figure 4: The shown r-graph, rooted in the blank node rounded by the red ellipse, represents a CS of drugs Heparin and Ardeparin. Predicate labels are omitted for a matter of readability (only the predicate rdfs:subClassOf is recognizable, because it is represented by a blue arrow with an empty triangle as head). Path highlighted in triangle a is zoomed in Figure 5. We observe that the tool we used for visualization, RDF Gravity, denotes resources according to the following formatting rules: both grey rectangles and violet triangles (those containing an “A” and an “I”, respectively) are used for anonymous resources; both violet and blue rectangles (containing © and “URI” respectively) denote RDF resources.

5.1. Feasibility and Independence of the Dataset

We here show the computation of the CS of 300 randomly selected pairs of drugs modeled in Bioportal: 150 are extracted from SNOMED and 150 from NDFRT. The report refers to a program implementing an algorithm for computing a Common Subsumer, presented in a previous work [9]. All tests have been executed on an Intel Xeon server, equipped with a 3.00 GHz processor and 8 GB RAM.

We report in Table 1 the average results related to the 150 executions for each dataset.

The first column in Table 1 shows the dataset used for random extraction of resources a and b to compare. The second column reports d, that is the maximum RDF-distance from a and b used to select triples in $T_a$ and $T_b$. The third column shows t, the average execution time of

| Dataset | d  | t    | $|T_a|$ | $|T_b|$ | $|T_{cs}|$ |
|---------|----|------|-------|-------|--------|
| SNOMED  | 0  | 2172,21 | 9,53  | 9,43  | 4,94   |
|         | 1  | 35216,27 | 48,65 | 50,35 | 83,09  |
| NDFRT   | 0  | 2042,70 | 9,30  | 9,57  | 2,09   |
|         | 1  | 42687,63 | 63,30 | 80,01 | 21,28  |

Table 1: Average execution times (in milliseconds) and sizes of input and result sets for the computation of 300 pairs of resources randomly selected from SNOMED (150 pairs) and NDFRT (150 pairs).
Figure 5: Zoomed image of the path highlighted in triangle a in Figure 4. Patterns in the r-graph show that both drugs: i) may prevent Thromboembolism (ndfrt:N0000002934) and Venous Thrombosis (ndfrt:N0000004074) and that both Thromboembolism and Venous Thrombosis are classified as Thrombosis (ndfrt:N0000002936); ii) may prevent some disease which can be classified as Thrombosis.

the complete process computing $CS(a, b)$. The remaining columns show the average values of $|T_a|$, $|T_b|$, $|T_{cs}|$: the number of triples in each of the three sets. We report online, at http://193.204.59.20/rdfcs/JBi/Experiments.zip, the values of $t$, $|T_a|$, $|T_b|$, $|T_{cs}|$ and the serialization in Turtle of $T_{cs}$ for each of the 300 analyzed pairs.

As the reader may check (see column $|T_{cs}|$ in Table 1), the result sets returned for the pairs extracted from SNOMED are larger. Nevertheless, most of the triples included in the sets $T_{cs}$ are unreadable for humans. In fact, most of the triples shared by the pairs of drugs modeled in SNOMED hosted by Bioportal match the pattern:

```
  s snomed:SUBSETMEMBER l .
```

where $s$ is either an IRI or a blank node and $l$ is a literal (an alphanumeric code) not further explained in BioPortal. Being members of a same subset is, of course, an interesting feature, but the way SNOMED (hosted by Bioportal) models such a membership results hard to read for humans.

For this reason, we choose to show the full potential of our approach and to provide more details about results, with reference to some pairs of drugs modeled in NDFRT (hosted by Bioportal), that provides a more human-readable characterization of resources.

In particular, we report in Table 2 some information about the computation of the Common Subsumers ($CS(a, b)$) of 5 pairs $(a, b)$ of resources denoting drugs out of the 150 randomly selected from NDFRT.

The first two columns in Table 2 show the URIs of the two resources to compare, $a$ and $b$. The third column reports $d$, that is the maximum RDF-distance from $a$ and $b$ used to select triples in $T_a$ and $T_b$.

The remaining columns show, for each pair of resources $a$ and $b$, and for an RDF-distance $d$, the information below:

- $t$: the execution time of the complete process computing $CS(a, b)$;
- $|T_a|$, $|T_b|$, $|T_{cs}|$: the number of triples in each of the three sets;
- $t_{T_a}$ and $t_{T_b}$: the time for computing $T_a$ and $T_b$, respectively;
- $\sigma$: the sum of $t_{T_a}$ and $t_{T_b}$.

By comparing the fourth and the last column, the reader may notice that $t$ is almost equal to $\sigma$: the execution time is almost all devoted to the computation of the sets of triples $T_a$ and $T_b$. Once such sets have been computed, the algorithm runs in less than 1 second, thanks to the research effort spent in reducing the size of $T_a$ and $T_b$ through the identification of the stop-patterns listed in Appendix A. We recall that such a reduction preserves relevant information, as explained in Section 4.

From the observations above, it emerges that any further attempt to improve performance should address the computation of the r-graphs of resources to compare. We immediately notice that the time for computing $T_a$...
and $T_b$ (addressed by $\sigma$ in Table 2) is high also because of reasons independent on our implementation. In particular, $\sigma$ includes an overhead time for connection to the SPARQL endpoint and the time for retrieval, which are completely controlled by the SPARQL service.

In fact, it is easy to notice that $t_r$ is much higher than $t_{c,s}$ in almost all examples (all of them if $d = 0$): once the connection has been established, the determination of triples to include in the two sets $T_a$ and $T_b$ works in comparable times. When $d = 1$, the size of the sets to compute affects the execution time much more than the overhead, which becomes negligible.

In order to demonstrate the existence of the overhead time above and roughly quantifying it, we perform the following experiment: we swap resources $a$ and $b$ in the computation of the CS of the same pairs shown in Table 2. In Table 3, we show, for each pair of resources $a$ and $b$, and for both $d = 0$ and $d = 1$, the execution time values described below:

| Resource $a$ | Resource $b$ | $d$ | $t$ | $|T_a|$ | $|T_b|$ | $|T_{c,s}|$ | $t_{c,s}$ | $t_r$ | $\sigma = t_r + t_{c,s}$ |
|--------------|--------------|-----|-----|-------|-------|-------|--------|-------|-----------------|
| ndftr:N0000146860 | ndftr:N0000146860 | 0   | 2910| 21   | 12   | 7     | 2016   | 856   | 2872            |
| ndftr:N0000146860 | ndftr:N0000146860 | 1   | 65619| 57  | 34   | 33    | 41512  | 24010 | 65522           |
| ndftr:N0000146860 | ndftr:N0000022083 | 0   | 3002| 21   | 14   | 9     | 2016   | 945   | 2961            |
| ndftr:N0000146860 | ndftr:N0000022083 | 1   | 69075| 57  | 36   | 48    | 41142  | 27850 | 68992           |
| ndftr:N0000146791 | ndftr:N0000146860 | 0   | 2818| 20   | 21   | 2     | 1899   | 877   | 2776            |
| ndftr:N0000146791 | ndftr:N0000146860 | 1   | 88290| 55  | 57   | 56    | 39954  | 48187 | 88141           |
| ndftr:N0000022054 | ndftr:N0000145931 | 0   | 2849| 25   | 25   | 23    | 1964   | 841   | 2805            |
| ndftr:N0000022054 | ndftr:N0000145931 | 1   | 95622| 58  | 58   | 88    | 49241  | 46279 | 95520           |
| ndftr:N0000022054 | ndftr:N0000147503 | 0   | 2922| 25   | 28   | 14    | 2029   | 840   | 2869            |
| ndftr:N0000022054 | ndftr:N0000147503 | 1   | 99401| 58  | 65   | 64    | 47640  | 51614 | 99254           |

Table 2: Execution times (in milliseconds) and sizes of result sets for the computation of the CSs (for two different values of $d$) for five pairs of drugs.

and $T_b$ (addressed by $\sigma$ in Table 2) is high also because of reasons independent on our implementation. In particular, $\sigma$ includes an overhead time for connection to the SPARQL endpoint and the time for retrieval, which are completely controlled by the SPARQL service.

In fact, it is easy to notice that $t_r$ is much higher than $t_{c,s}$ in almost all examples (all of them if $d = 0$): once the connection has been established, the determination of triples to include in the two sets $T_a$ and $T_b$ works in comparable times. When $d = 1$, the size of the sets to compute affects the execution time much more than the overhead, which becomes negligible.

In order to demonstrate the existence of the overhead time above and roughly quantifying it, we perform the following experiment: we swap resources $a$ and $b$ in the computation of the CS of the same pairs shown in Table 2. In Table 3, we show, for each pair of resources $a$ and $b$, and for both $d = 0$ and $d = 1$, the execution time values described below:

1. $t_{c,s}(CS(a, b))$: the time for computing $T_a$ when $a$ is the first argument in CS computation ($t_{c,s}$ in Table 2);
2. $t_{c,s}(CS(b, a))$: the time for computing $T_a$ when $a$ is the second argument in CS computation;
3. $t_{c,s}(CS(a, b))$: the time for computing $T_b$ when $b$ is the second argument in CS computation ($t_{c,s}$ in Table 2);
4. $t_{c,s}(CS(b, a))$: the time for computing $T_b$ when $a$ is the first argument in CS computation.

In rows referring to $d = 0$, the overhead for computing $T_a$ (respectively, $T_b$) may be roughly quantified as the difference between values in columns 4 and 5 (respectively, 7 and 6). As hinted before, in rows referring to $d = 1$ this overhead is negligible w.r.t. to execution time required to build the two sets, whose size is much bigger than for $d = 0$ (see Table 2 for sizes).

5.2. Comparison to numerical methods

The main distinguishing feature of our approach to the comparison of RDF resources is that it provides a description of the commonalities rather than a measure of similarity. Intuitively, the informative content embedded in the computed CS allows for defining a numerical measure evaluating “how much” two resources are similar on the basis of their shared commonalities. A first attempt in this direction has been already made [23], but the definition and the evaluation of such a measure is out of the scope of this paper and will be investigated in future work.

Without delving into the development of functions, we just compare the size of CSs sets: a bigger $T_{c,s}$ denotes—in general—a pair with more commonalities. For this reason, we compared in Table 4, for the ten pairs analyzed in Section 5.1, the values of $|T_{c,s}|$ and of the “Euclidean Distance” $\sum_d |T_{c,s}|^2$ computed as detailed below. Intuitively, distance is meant to be inversely proportional to similarity.

The Euclidean Distance has been computed by implementing a workflow using the Linked Open Data extension (LODExtension) [33] of the Machine Learning tool RapidMiner [56]. In particular, an RDF graph is generated for each resource in the pair, with user-specified graph depth. Then, a kernel method [37] counts the different walks in the subgraphs (up to the provided graph depth) around the root and returns a so-called “ExampleSet”: a set of kernel-generated features describing the resources of interest. The generation process is set to make use of inference on explicit knowledge. The
Euclidean Distance is computed between items of the ExampleSet.

Even from the small set of examples in Table 4, the reader may notice that the values of $|T_{cs}|$ and Euclidean Distance are not inversely proportional, as one may expect. In other words, a ranking based on the size of the CS would not coincide with a ranking based on Euclidean Distance, as a matter of fact.

When dealing with numerical measures, in front of this kind of misalignment, one can only look at numbers, and, possibly, tune the measure to improve correspondence to human judgment. On the contrary, our approach to comparison provides an explicit explanation of shared features, that no numerical method may return. As an example, let us consider Row 8 and Row 9 in Table 4. In both cases, even though the value of $|T_{cs}|$ is rather big (56 and 64, respectively), the two pairs are considered rather distant (31,843 and 24,413, respectively), if compared with other pairs in the table. Thanks to the logical nature of our method, we looked at the content of $T_{cs}$, $T_a$ and $T_b$ for the pairs in Row 8 and Row 9 and discovered that the items in these pairs have several different features, other than the—though numerous—shared ones. The reason for their distance values is, therefore, in their differences.

At the present stage, our approach does not compute a difference between RDF resources, but it is the only one able to explicitly exhibit commonalities. Nevertheless, the definition and the computation of a difference between RDF resources is part of our future work, because we believe this is a complementary information crucial for resource comparison.

### 5.3. Lessons learned

The evaluation proposed so far leads to several interesting conclusion, we briefly sketch below.

<table>
<thead>
<tr>
<th>Resource a</th>
<th>Resource b</th>
<th>d</th>
<th>$t_{T_a}(CS(a, b))$</th>
<th>$t_{T_b}(CS(b, a))$</th>
<th>$t_{T_c}(CS(a, b))$</th>
<th>$t_{T_c}(CS(b, a))$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ndfrt:N0000146860</td>
<td>ndfrt:N0000148552</td>
<td>0</td>
<td>2016</td>
<td>837</td>
<td>856</td>
<td>2040</td>
</tr>
<tr>
<td>ndfrt:N0000146860</td>
<td>ndfrt:N0000148552</td>
<td>1</td>
<td>41512</td>
<td>38994</td>
<td>24010</td>
<td>24862</td>
</tr>
<tr>
<td>ndfrt:N0000146860</td>
<td>ndfrt:N0000222083</td>
<td>0</td>
<td>2016</td>
<td>807</td>
<td>945</td>
<td>2007</td>
</tr>
<tr>
<td>ndfrt:N0000146860</td>
<td>ndfrt:N0000222083</td>
<td>1</td>
<td>41142</td>
<td>39859</td>
<td>27850</td>
<td>28574</td>
</tr>
<tr>
<td>ndfrt:N0000146791</td>
<td>ndfrt:N0000146860</td>
<td>0</td>
<td>1899</td>
<td>873</td>
<td>877</td>
<td>2000</td>
</tr>
<tr>
<td>ndfrt:N0000146791</td>
<td>ndfrt:N0000146860</td>
<td>1</td>
<td>39954</td>
<td>36774</td>
<td>48187</td>
<td>40915</td>
</tr>
<tr>
<td>ndfrt:N0000022054</td>
<td>ndfrt:N0000145931</td>
<td>0</td>
<td>1964</td>
<td>911</td>
<td>841</td>
<td>2040</td>
</tr>
<tr>
<td>ndfrt:N0000022054</td>
<td>ndfrt:N0000145931</td>
<td>1</td>
<td>49241</td>
<td>49439</td>
<td>46279</td>
<td>47450</td>
</tr>
<tr>
<td>ndfrt:N0000022054</td>
<td>ndfrt:N0000147503</td>
<td>0</td>
<td>2029</td>
<td>876</td>
<td>840</td>
<td>2006</td>
</tr>
<tr>
<td>ndfrt:N0000022054</td>
<td>ndfrt:N0000147503</td>
<td>1</td>
<td>47640</td>
<td>47338</td>
<td>51614</td>
<td>57097</td>
</tr>
</tbody>
</table>

Table 3: Execution times (in milliseconds) for computing r-graphs of $a$ and $b$ in the processes for computing $CS(a, b)$ and $CS(b, a)$. 

The analysis in Section 5.1 demonstrates that most of the time to compute the CS of two resources $a$ and $b$ is due to the extraction of triples to embed in $T_a$ and $T_b$. We stress that reported times refer to a fully on-line extraction process. By reverting to partially off-line solutions, like the preliminary caching of all triples of interest for an application, the performance of CS computation could improve by dropping the above-mentioned overhead time.

Nevertheless, by giving up a fully on-line solution, the computed CS would be obsolete and could loose part of the available informative content, especially if the employed dataset is frequently updated. In other words, a trade-off between feasible response times and up-to-date informative content exists.

The analysis in Section 5.2 shows the value added by a logical explanation of commonalities to the problem of evaluating the similarity of resources in RDF. The discussion shows that, even for applications interested only in a numerical measure of similarity, a logical explanation of the shared features can help in understanding the reasons for returned values and consequently tuning the measure.

Furthermore, we remark that the described experiments are completely reproducible, because we provided in Appendix A the full list of stop-patterns and uninformative triples used to customize our method for drug comparison.

The identification of stop-patterns and uninformative triples follows a deep analysis of Bioportal, which allows us for flexibly discarding only triples irrelevant w.r.t. comparison. As a result, the managed sets $T_a$, $T_b$, and $T_{cs}$ are up to ten times smaller than the corresponding sets derived without filtering stop-patterns (for $d = 1$). The implementation of uninformative triples strategy further reduces the size of $T_{cs}$. Yet, the returned CS preserves all the informative content useful for com-
Table 4: Size of $|T_{ts}|$ and Euclidean Distance (defined in RapidMiner), for a pair of resources $(a,b)$.

| Row | Resource a          | Resource b          | $d$ | $|T_{ts}|$ | Euclidean Distance |
|-----|---------------------|---------------------|-----|----------|-------------------|
| 1   | ndfrt:N0000146791   | ndfrt:N0000146860  | 0   | 2        | 16.31             |
| 2   | ndfrt:N0000146860   | ndfrt:N0000148552  | 0   | 7        | 12.41             |
| 3   | ndfrt:N0000146860   | ndfrt:N0000022083  | 0   | 9        | 11.916            |
| 4   | ndfrt:N0000022054   | ndfrt:N0000147503  | 0   | 14       | 11.225            |
| 5   | ndfrt:N0000022054   | ndfrt:N0000145931  | 0   | 23       | 4                 |
| 6   | ndfrt:N0000146860   | ndfrt:N0000148552  | 1   | 33       | 24                |
| 7   | ndfrt:N0000146860   | ndfrt:N0000022083  | 1   | 48       | 21.726            |
| 8   | ndfrt:N0000146791   | ndfrt:N0000146860  | 1   | 56       | 31.843            |
| 9   | ndfrt:N0000022054   | ndfrt:N0000147503  | 1   | 64       | 24.413            |
| 10  | ndfrt:N0000022054   | ndfrt:N0000145931  | 1   | 88       | 9.592             |

6. Conclusion and Future Work

In this paper we applied a general method for comparing the properties of two drugs whose data are available in Bioporal as RDF triples. This case study showed several distinguishing features of our method:

- it is domain-independent and may be easily customized to the domain of interest; in fact, our approach may be flexibly adapted to the problem at hand, by just setting domain-dependent parameters, in a modular fashion.

- it returns an explicit description of the features shared by the two resources, in terms of RDF triples; this distinguishes our approach from the rest of the literature, completely devoted to the proposal of numerical measures of similarity, to the best of our knowledge.

- it makes completely explicit the criteria for selection of triples relevant for each application; this makes our approach overcome most of the proposals for applications using RDF datasets as data sources. In fact, the size of most available datasets makes unfeasible applications that manage all the triples stored and asks for the selection of a subset which is relevant to the problem at hand. Unfortunately, as far as we know, the criteria adopted for such a selection are always undeclared to the reader.

On the contrary, we explicitly provide our criteria and show how to customize our general method to the problem of comparing biomedical resources, with specific reference to drugs. This makes our experiments completely reproducible and provides a list of patterns not relevant for the comparison, which we call stop-patterns. The list of stop-patterns may be used and extended by other researchers interested in comparing RDF resources.

Remarkably, our method works by on-line querying Bioporal, that hosts the datasets we use as data source. Thus, it relies on data which are always up-to-date at no modeling cost for the application. The other side of the coin is a relatively high response time, which may be reduced if solutions working partially off-line are chosen.

Part of our future work will be devoted to investigation on methods for a human-readable presentation of results. Our approach returns, in fact, a description of features shared by the two resources, in terms of RDF triples. We believe that an automated process translating such a description in natural language may be useful for the adoption of our method in tools for resource comparison.

References


Appendix A. Stop Patterns and Uninformative Triples

In order to make our experiments reproducible, and to share with other researchers our evaluation about what information in Bioportal is not relevant for comparisons, in the following we report the full list of stop-patterns and uninformative triples set in our implementation.

**Stop-patterns** The complete set of stop-patterns is made up by all triples $\langle s \ p \ o \rangle$, such that one of the following conditions holds:

- $p = ndfrt:NDFRT_KIND$ and $o$ is a literal
- $y = ndfrt:contraindicated_drug$ and $s$ has an RDF-distance greater or equal to 1 w.r.t. to $r$
- $p = rdf:type$ and $o \in \{ owl:Class, rdf:Property, owl:ObjectProperty, owl:AnnotationProperty, owl:DatatypeProperty \}$
- $p = umls:hasSTY$ and $o = umls:sty/T047$.

**Uninformative triples** As recalled in Section 3.2, the LCS of a pair of resources modeled as r-graphs is an r-graph itself. We identified as uninformative triples all triples $\langle x \ y \ z \rangle$ in the LCS such that $z$ is a blank node with no successors and $y \in \{ rdf:type, dct:subject, foaf:isPrimaryTopicOf, rdfs:domain, rdfs:range, rdfs:seeAlso, owl:Thing, owl:equivalentClass, owl:equivalentProperty, rdfs:subClassOf, rdfs:subPropertyOf, skos:broaderr, vrank:hasRank, vrank:rankValue, vrank:rankValue, umls:hasSTY, ndfrt:may_treat, ndfrt:may_prevent, ndfrt:contraindicated_drug, ndfrt:has_mechanism_of_action, ndfrt:NDFRT_KIND, ndfrt:hasIngredient, snomed:has_active_ingredient, snomed:SUBSETMEMBER\}$.

Appendix B. Serialization in Turtle of Exemplified r-graphs

In the following, the r-graph describing Heparin, shown in Figure 2, is serialized according to Turtle notation:
In the following, the CS shown in Figure 4, rooted in the blank node _:r7ffd, is serialized according to Turtle notation:

_:r7ffd
umls:hasSTY umlssty:T118;
umls:hasSTY umlssty:T121;
numls:hasSTY umlssty:T123;
ndfrt:has_ingredient ndfrt:N0000006341;
ndfrt:has_physiologic_effect ndfrt:N0000008556;
rdfs:subClassOf ndfrt:N0000010590;
ndfrt:may_treat ndfrt:N0000002935;
ndfrt:may_treat ndfrt:N000000408;
ndfrt:may_treat ndfrt:N0000002085;
ndfrt:may_treat ndfrt:N000000722;
ndfrt:may_treat ndfrt:N000000858;
ndfrt:may_prevent ndfrt:N0000002934;
ndfrt:may_prevent ndfrt:N0000004074;
ndfrt:may_prevent ndfrt:N0000002541;
ndfrt:may_prevent ndfrt:N0000002936;
ndfrt:may_prevent ndfrt:N0000002937;
ndfrt:may_prevent ndfrt:N0000002467;
ndfrt:may_prevent ndfrt:N0000000722;
ndfrt:may_prevent ndfrt:N0000000858;
ndfrt:contraindicated_drug ndfrt:N0000000999;
ndfrt:contraindicated_drug ndfrt:N0000002932;
ndfrt:contraindicated_drug ndfrt:N0000004102;
ndfrt:has_mechanism_of_action ndfrt:N0000009963;
ndfrt:has_mechanism_of_action ndfrt:N0000009960 .

umlssty:T118 rdfs:subClassOf umlssty:T046;
umlssty:T121 rdfs:subClassOf umlssty:T120;
umlssty:T123 rdfs:subClassOf umlssty:T120.

ndfrt:N0000006341
umls:hasSTY umlssty:T118;
umls:hasSTY umlssty:T121;
umls:hasSTY umlssty:T123;
rdfs:subClassOf ndfrt:N0000007893. 

ndfrt:N0000008556
umls:hasSTY umlssty:T042;
rdfs:subClassOf ndfrt:N0000008362 .

ndfrt:N0000010590
umls:hasSTY umlssty:T185;
rdfs:subClassOf ndfrt:N0000010582 .

ndfrt:N0000002935
rdfs:subClassOf ndfrt:N0000002393 .

ndfrt:N0000000408
rdfs:subClassOf ndfrt:N0000000406 .

ndfrt:N0000002085 rdfs:subClassOf ndfrt:N0000003550 .

ndfrt:N0000000722 rdfs:subClassOf ndfrt:N0000004159 .

ndfrt:N0000000722 rdfs:subClassOf ndfrt:N0000000577 .

ndfrt:N0000002934 rdfs:subClassOf ndfrt:N0000002936 .
ndfrt:N0000000999 rdfs:subClassOf ndfrt:N0000001610 .

ndfrt:N0000002932 rdfs:subClassOf ndfrt:N0000000577 .

ndfrt:N00000004102
umls:hasSTY umlssty:T046;
rdfs:subClassOf ndfrt:N0000000730;
ndfrt:induced_by ndfrt:N0000146210;
ndfrt:induced_by ndfrt:N0000155927;
ndfrt:induced_by ndfrt:N0000155929;
ndfrt:induced_by ndfrt:N0000155926 .

ndfrt:N0000009963
umls:hasSTY umlssty:T044;
rdfs:subClassOf ndfrt:N0000000169 .

ndfrt:N0000009960
umls:hasSTY umlssty:T044;
rdfs:subClassOf ndfrt:N000000231 .

ndfrt:N0000002934 rdfs:subClassOf ndfrt:N0000002936 .
ndfrt:N0000000407
rdfs:subClassOf ndfrt:N0000001067.

ndfrt:N0000002541
rdfs:subClassOf ndfrt:N0000001067.

ndfrt:N0000000407
rdfs:subClassOf ndfrt:N0000002338.

ndfrt:N0000002467
rdfs:subClassOf ndfrt:N0000001067.

ndfrt:N00000002467
rdfs:subClassOf ndfrt:N0000001067.

ndfrt:N00000002467
rdfs:subClassOf ndfrt:N0000001067.

ndfrt:N00000002467
rdfs:subClassOf ndfrt:N0000001067.

ndfrt:N00000002467
rdfs:subClassOf ndfrt:N0000001067.

ndfrt:N00000002467
rdfs:subClassOf ndfrt:N0000001067.