Abstract

This paper presents an innovative knowledge management approach for the validation and transfer of knowledge between semantic networks or ontologies and a target ontology represented in OWL format. This process has been designed for addressing the quality improvement of ontologies automatically created by learning techniques. The knowledge transfer process is a semi-automatic computer aided method to assist the domain expert to improve the target ontology.

To validate our proposal, we have used an automatically generated target ontology. We used knowledge transfer from the well-known BabelNet semantic graph and a manually generated ontology to improve the quality of the target ontology. Finally, to show the suitability of our proposal in the ontology fixing process, we compare the improved target ontology resulting from the application of the proposed validation and knowledge transfer techniques with its original version. We developed an example of our proposal in our OntologyFixer tool which is available on a GitHub repository (https://github.com/gabyluna/OntologyFixer/tree/ontofixer_v2).

Keywords

Ontology; Semantic Graph; Automatic Knowledge Exchange; Ontology Fixing; Ontology Quality
Introduction and motivation

Ontologies enable computers to structure and manage information according to semantic criteria in order to perform computational inference processes. Therefore, there is great interest in building new ontologies that provide semantic representations of different areas of knowledge and eventually make it possible to solve new problems. To this end, in recent years a large number of ontologies have been developed and made available to the scientific community through different repositories. As examples we highlight NCBO BioPortal*, which allows downloading a large number of biomedical ontologies, LOV4IoT† which shares more than 800 ontologies that model different aspects of Internet of Things (IoT), the Finnish Ontology Library Service ONKI‡, Swoogle§ or the Protégé ontology library**. This fact proves that there is a great interest in the creation of ontologies to accurately model the semantics associated with specific problems.

Gruber defines an ontology as “A formal explicit specification of a shared conceptualization for a domain of interest”1. Gruber’s definition has been widely adopted in the web semantics community due to its simplicity and comprehensiveness. The formal and explicit specification of an ontology enables large-scale and systematic knowledge management, as well as computer-based processing and reasoning. In 2004, the World Wide Web Consortium (W3C) proposed the Web Ontology Language (OWL)†† as the key to the Semantic Web technologies. The OWL standard is a core component of the Semantic Web Protocol Stack for formal knowledge representation and reasoning. It is supported by a wide variety of open tools and frameworks and therefore, it was adopted as the reference ontology representation in our study. In detail, we used OWL DL dialect and take advantage of the knowledge represented and inferred in the ontology, expressed through individuals, classes, object properties and datatype properties defined in OWL DL.

Ontology construction is a cumbersome process that involves domain experts and knowledge engineers. Hence, ontology learning techniques have been introduced to enable the automatic construction of ontologies from documents written in natural language2,3. Combining these techniques with texts obtained from reliable sources such as scientific papers, we can obtain ontologies more easily than building them manually from scratch. However, current ontology learning tools have many limitations and the ontologies created by this method have to be fixed at a later stage. In particular, we highlight the identification of unhelpful terms that introduce junk knowledge in the ontology or the inexistence of terms in source documents (texts) that should be present (represented as nodes) in the target ontology, resulting in the lack of specific knowledge representation in the domain. Our contribution specially addresses the later issue. Several recent

* Available at http://bioportal.bioontology.org/
† Available at http://lov4iot.appspot.com
‡ Available at https://seco.cs.aalto.fi/services/onki/
§ Available at http://swoogle.umbc.edu/
** Available at https://protegewiki.stanford.edu/wiki/Protege_Ontology_Library#OWL_ontologies
†† See http://www.w3.org/2004/OWL/
works have highlighted this problem and have introduced methodologies and tools that help to partially automate the ontology fixing process\(^4\).\(^5\).

The main drawback of the ontology fixing methodologies is that the process is mainly driven by quality assessments, consistency errors and design defects. Therefore, there is no validation from a semantic point of view of the correctness and completeness of the knowledge contained in the ontology. However, some third-party ontologies may have some knowledge that could be used to perform some validations of the represented knowledge. We could leverage ontology matching/alignment processes\(^6\).\(^7\) to identify relevant knowledge from third-party ontologies and use it to fix an ontology. In addition, using similar schemes, relevant knowledge can be extracted from semantic graphs (ontology dictionaries) enabling them as a very useful tool for ontology quality validation and improvement processes. Our proposal includes a simple ontology matching scheme and shows how to take advantage of relevant knowledge discovered in different sources of information to fix semi-automatically generated ontologies. The fixing mechanism has been tested on an ontology used for experimentation in previous ontology fixing studies. Our experiments show the usefulness of the proposal and its possible application for semi-automatic construction of ontologies.

The remainder of this study is structured as follows: Section 2 summarises previous efforts and proposals to improve ontology fixing processes. Section 3 presents our proposal to leverage third-party knowledge sources for ontology fixing. Section 4 performs an evaluation of our proposal and its possible application for fixing the ontologies. Finally, Section 5 shows the main conclusions and outlines directions for future work.

**Background**

In order to correct, improve and enrich the content of ontologies created automatically or semi-automatically with the help of different tools, researchers have proposed over the years different methods and strategies to fix those errors that may occur in different aspects of ontologies such as its structure, functionality, maintainability, semantics, etc. We briefly describe below some studies that address (i) improving the quality of the ontologies, (ii) strategies for identifying possible design defects and (iii) tools/methodologies for fixing the ontologies.

Bandeira *et al.* introduced FOCA\(^8\), a methodology for evaluating ontologies. It is a three-step method that comprises the identification of ontology type, the application of a Goal/Question/Metric approach and the quality assessment. Thus, FOCA implements a role-based calculation of ontology quality according to its type; it includes a questionnaire to accomplish the evaluation and a statistical model that automatically computes the quality of the ontologies.

The OQuaRE\(^9\) framework\(^‡‡\) is a method to evaluate the quality of ontologies that adapt the SQuaRE standard (originally designed to assess the quality of software products) to the context of ontologies. OQuaRE uses different metrics to assess the quality of the ontologies with regard to different dimensions, including reliability, operability, maintainability, compatibility, transferability, and functional adequacy.

---

\(^‡‡\) Available at http://miuras.inf.um.es/oquarewiki/index.php5/Main_Page
FOEval\textsuperscript{10} introduces an evaluation model to choose the ontology that best fits the user requirements. The model allows users to select indicators and assign weights for each one selected among a wide variety of available quality indexes.

OntoClean\textsuperscript{11} is a methodology for the validation of the ontological adequacy and logical consistency of taxonomic relationships. It uses various concepts of philosophy, such as rigidity, identity criteria and unity, to provide modelling guidelines.

OntoQA\textsuperscript{12} is an approach that analyses ontology schemas and their instances (such as knowledge bases) and describes them using a well-defined set of metrics\textsuperscript{13} to evaluate the content of the ontologies. One of the features that highlight OntoQA is its flexible technique for classifying ontologies based on their content and relevance to a set of keywords, as well as user preferences.

Su and Gulla\textsuperscript{14} addressed semantic problems of ontologies and introduced a method to find semantic correspondences between ontology components in order to provide a semi-automatic mapping method and a prototype mapping system to support the ontology mapping process to improve semantic interoperability in heterogeneous systems.

Another study\textsuperscript{15} introduced a formal technique to control ontology evolution by domain experts and shows two cases of ontology evolution based on ontology templates that guarantees an ontology without inconsistencies.

OOPS!\textsuperscript{16} is a web application that provides mechanisms for the automatic detection of potential errors, called pitfalls, in order to help developers during the validation process. Each pitfall provides the following information: title, description, elements affected, and importance level to give the user enough information to identify problems and take appropriate actions.

Nicolas Troquard \textit{et al.}\textsuperscript{17} introduced an ontology repair methodology that preserves most of the original knowledge without additional costs in terms of computational complexity. On the other hand, researchers quickly realised that bringing the concept of smells/pitfalls to the domain of knowledge engineering could help in the process of improving the quality of ontologies. The work of Póveda-Villalón \textit{et al.}\textsuperscript{18} introduces and classifies a catalogue of 24 pitfalls that usually appear during the process of building ontologies. The classification of pitfalls is based on several works prepared by different authors who have identified common mistakes during the process of modelling ontologies.

We also introduced a methodology to evaluate the quality of the ontologies (quantitatively and graphically) and to correct ontology inconsistencies by minimising design defects\textsuperscript{4}. The proposed methodology is based on the Deming cycle and is supported by quality standards that have proven to be effective in the software engineering domain and that have a high potential to be extended to the quality management of knowledge engineering.

With this in mind, we find that the methodologies and tools developed so far could benefit from ontology mismatch corrections by transferring knowledge from knowledge bases such as third-party ontologies designed by experts and ontological dictionaries strongly reinforced and fed from different sources of knowledge. To achieve this goal, we use an ontology alignment/matching scheme. Ontology matching is the process of discovering correspondences between elements (classes, objects, etc.) included in two or more ontologies. As can be observed in some recent literature reviews\textsuperscript{6,7,19}, ontology matching has been widely studied. Some recent works have introduced complex ontology alignment systems based on BERT\textsuperscript{20,21} and neural networks\textsuperscript{22}. 
Ontology alignment schemes have been successfully used to address different tasks such as ontology merging, query answering, data translation or semantic web navigation. We have also found that ontology alignment schemes could be successfully used to address ontology fixing tasks. Next section introduces our proposal for addressing knowledge transfer and leveraging knowledge exchanged to improve and fix ontologies.

**Semantic Transfer Process**

In this section we describe our strategy to improve and validate the semantics of automatically generated ontologies that may have some conceptual and relational shortcomings. The goal of our proposal is to validate and enhance the knowledge-base concepts (classes and individuals) using third-party ontologies or ontological dictionaries (such as Babelnet§§ or Wordnet***). Figure 1 graphically represents the semantic transferring strategy introduced in this work.

---

§§ Available at https://babelnet.org

*** Available at https://wordnet.princeton.edu/
Figure 1. Semantic transferring process

As shown in Figure 1, the strategy comprises four stages: (i) search for each class or individual concept presence in third-party ontologies or semantic dictionaries. (ii) compare the relations between concepts in the knowledge bases (ontologies or dictionaries) with those of the target ontology, (iii) when both relations match, proceed to transfer or adapt this knowledge to the target ontology and (iv) validate that the new ontology has improved the quality of the original ontology after the applied changes.

Stages i (concept comparison) and ii (relations comparison) correspond to a two-step ontology matching strategy that identifies relevant knowledge for ontology fixing. The first stage comprises a string-based element-level matcher that identifies possible alignments between nodes of ontologies. Moreover, the second stage implements a taxonomy-based structure-level matcher that is able to refine the results achieved in the first stage. The ontology matching process was designed to be simple and to allow the use of third-party ontologies and semantic graphs.

During the first stage, an iterative process indicated in Figure 1 is carried out with each of the nodes (classes or individuals) of the ontology using the OWL API†††. The name of the class or individual is used for running a search in other ontologies or ontological dictionaries to find knowledge that can be used for improving the target ontology. Figure 2 shows the concept_comparison method that compares the concepts of the knowledge sources with those of the target ontology.

```java
// Figure 2. Algorithm designed to execute the concept comparison (first stage)

function concept_comparison(targetOntology, ontologiesThirdParty, ontologicalDictionaries):
    listRecords
    nodesMatch <- new List;
    forEach (o ∈ ontologiesThirdParty):
        nodesMatch <- join(nodesMatch, concept_comparison_ontology(targetOntology, o, LMD))
    synsetsMatch <- new List;
    forEach (od ∈ ontologicalDictionaries):
        synsetsMatch <- join(synsetsMatch, concept_comparison_dict(targetOntology, od, LMD))
    listRecords <- join(nodesMatch, synsetsMatch)
    return listRecords
end_function

††† See http://owlapi.sourceforge.net/
```
misspellings, we use Levenstein distance\textsuperscript{23} to retrieve lexically close nodes. The algorithm only returns those nodes with a Levenshtein distance less or equal to a certain value (see line 01 in Figure 2 which refers to Levenshtein Maximum Distance parameter). For example, if we consider the node named “action” in the target ontology, the node named “actions” in a knowledge source will be returned for the next stage because the Levenshtein distance between both node names is less or equal to the value set for LMD (LMD=1).

Moreover, we also search for the name of each node of the target ontology in external ontological dictionaries. For this, the search will be limited to the grammatical use of the word (node name) as a noun (POS, part of speech). The result of the search will return a list of synsets. However, due to the polysemy of natural language, some synsets may not match the semantic meaning of the target ontology node. The result of executing the first stage is a collection of records that could be used to improve the ontology which consist of third-party ontology nodes and synsets gathered from ontological dictionaries.

During the second stage (relations comparison/disambiguation), we identify and discard records containing invalid (out of scope) knowledge that were collected in the first stage. We consider a record to have invalid knowledge when it fits lexically with the original ontology node but they are semantically distinct, i.e. they have different meanings. Therefore, the identification of invalid knowledge involves the application of techniques inspired by Word Sense Disambiguation (WSD) in texts\textsuperscript{24}. However, unlike WSD, the identification of invalid knowledge does not involve the analysis of text fragments. Instead, the algorithm must compare the group of nodes connected to the original concept in the target ontology with the group of nodes connected to the same concept in the third-party ontology (or the ontological dictionary) used as a source of knowledge that is connected to the retrieved record for the concept. If there is no correspondence between these two groups of nodes, then the knowledge included in the retrieved record can be ignored. Exceptionally, when the target ontology has a very specific subject, the disambiguation process can be relaxed and check only if the nodes connected to each record retrieved in the source of knowledge (semantic graph / ontology) are present in the target ontology (regardless of how these nodes are connected in the target ontology). The algorithm used for this comparison is shown in Figure 3.
The function `is_valid` (Figure 3) determines whether a record retrieved during the previous stage contains valid knowledge. The function computes the ratio of relations of the original node to neighbour nodes in the target ontology that are also present in the retrieved record for the knowledge source. To do so, we use the `get_nodenames_connected_with(Node n, Ontology|SemanticGraph g)` function that returns a set of names with the nodes connected to node n in the ontology or semantic graph g and `get_ontology_node_set(Ontology o)` function that computes the set of nodes included in an ontology o. In case of using misspelt ontologies, the intersection function (line 13) can be replaced by specific functions using string similarity functions (such as Levenshtein distance). The condition to relax the semantic disambiguation process (as explained in the previous paragraph) is modelled using the `VERY_SPECIFIC_TARGET_ONTOLOGY` variable (see lines 2 and 15). Finally, a matching threshold parameter (MT) has been introduced to determine the minimum semantic match (lines 1 and 22).

The third stage uses the valid knowledge included in the records retrieved in the second stage to improve the structure and the semantic consistency of the target ontology. We exploit the knowledge extracted from the connections between each retrieved record and other nodes in the same knowledge source. In the case of semantic graphs that include a wide range of relations between nodes (e.g. meronyms, holonyms), only subclasses, superclasses and instances of this type of knowledge source are transferred. Then, the integration of knowledge allows us to verify that the relations retrieved for each specific node fully match those included in the target ontology. We have to transfer the knowledge about relations that do not exist in the target ontology, as well as transfer connected nodes if they do not exist, and fix the existing relations in the case they do not have the same type/direction. When a knowledge transfer is possible, i.e. the relations retrieved do not fully match with the target ontology, we can also detect a symptom of concept mismatch that
can be corrected using quick fix strategies. Additionally, we can transfer the names of the nodes in case that they do not fully match those of the knowledge source (eventually using a quick-fix strategy). By including this knowledge in the ontology, we are greatly modifying its structure and content, so its quality may be compromised or improved. To determine whether the changes can be applied, the ontology must be validated by an expert user (using quick-fix) or according to the ontology quality validation process described in the next (fourth) stage.

In the fourth stage, a validation of the ontology quality is performed using the OntologyFixer web tool introduced recently. This web tool includes several modules for fixing errors and ontology quality assessment using quality metrics based on the OQuaRE framework. Correctness and completeness metrics often require more intensive human intervention. In order to reduce user effort and intervention, our evaluation is performed by calculating nine content/structure metrics: RROnto (Relationship Richness), INROnto (Relationships per class), ANNOnto (Annotation Richness), CROnto (Class Richness), NOMOnto (Number of Properties), RFCOnto (Response for a Class), CBOOnto (Coupling Between Objects), LCOMOnto (Lack of Cohesion in Methods), and RCOnto (Average Population / Average Distribution of Instances Across all Classes). These metrics can be calculated automatically and allow the evaluation of different aspects of ontologies, such as their structure, functional adequacy, compatibility, transferability, operability and maintainability. They are represented in a radar chart where the vertices of the polygon are the measures of the metrics and the total area is an indicator of overall quality. The value of the area must be compared with that of the original ontology in order to evaluate the improvement of the ontology quality and, therefore, to support the recommendation of whether or not to apply the changes. This validation can eventually be replaced or complemented by the assistance of an expert user.

The next section includes several examples of application of the automatic knowledge exchange methodology introduced.

A case study

This section provides a detailed description of the experiments performed to demonstrate the utility of the introduced methodology. First subsection describes the target ontology used to evaluate the designed quality improvement strategy. Second subsection presents the results obtained from applying the proposed semantic transfer strategy to the target ontology. Finally, third subsection shows a list of lessons learned from the process of comparing and transferring semantic knowledge to the ontology.

Experimental data

In the previous study, an ontology learning process was carried out to build an ontology from 62 specific papers in the domain of Multi-objective optimization. To this end, different tools were combined to implement the whole process (i.e. information extraction and automatic ontology generation). This ontology (which was automatically generated and contains a broad range of errors, pitfalls and design defects) was shared in Zenodo platform, to be used to run experimental evaluations of quality metrics and ontology-fixing processes. This ontology was successfully used for experimental purposes in the above-mentioned study.

Although there are several repositories from which it is possible to download a large number of ontologies with an acceptable quality for different purposes, to the best of our knowledge, there are no other examples of ontologies that allow validating strategies for fixing and/or assessing the
quality of ontologies. Therefore, considering the nature of the current work, we found that the previously mentioned ontology will be an excellent case study for the evaluation of our proposal.

**Experimental data**

In this subsection we present some issues detected using the strategy outlined in the previous section with Babelnet and an ontology created by experts\textsuperscript{26,27} as sources of knowledge.

The configuration parameters were experimentally optimised to allow the knowledge transferring from a list of nodes (“optimization”, “aggregation”, “connection”, “decision making” of Babelnet and “preference model”, “desirability function”, “researcher”, “moo”, “metaheuristic”, “optimisation”, “PreferenceIntegration”, “PreferenceInformationFromDm” of the third-party ontology). In order to obtain those nodes that are orthographically similar to the target ontology and that can be used to extract knowledge, we set LMD=1 for the execution of the experiments. To select this value, we compared the “preference model” node of the target ontology with the “PreferenceModel” node of the third-party ontology during the experimental run. Those nodes matched with LMD=1, so we take this configuration value as valid for the first stage. On the other hand, the MT parameter (second stage) was set to 0.15. This is a suitable value to validate that the records achieved in the first stage match nodes of the target ontology. Finally, since the target ontology only incorporates semantic information about MOEA, the value of \texttt{VERY\_SPECIFIC\_TARGET\_ONTOLOGY} variable is set to true to relax the conditions for the disambiguation process (stage two).

As an example, we have chosen the class "optimization" to analyse the knowledge transfer process from the Babelnet semantic graph. When comparing with the ontological dictionary all the synsets related to the selected class must be returned. In this case, six synsets of type NOUN were returned by the Babelnet API (bn:10424683n, bn:01564896n, bn:02021921n, bn:00059223n, bn:00059223n, bn:03309733n). If more than one synset is obtained as a result, the strategy proposes to carry out a disambiguation process to select the best synset related to the sense of the target ontology. In Table 1 we can see the results of the selection process.
In the second stage of the introduced methodology, we compare the relations of the retrieved senses (synsets) with the original node named “optimization”. In particular, we count how many nodes that are connected with the retrieved records are also connected to this node in the target ontology. Finally, we select only the synset bn:01564896n as valid since it has the highest match, whose value is greater than MT. Once the synset bn:01564896n is marked as valid, we retrieve all of its relationships using Babelnet API and validate if they exist in the target ontology. Table 2 shows the list of relations found.

### Table 2. Relations retrieved from Babelnet semantic graph for knowledge exchange

<table>
<thead>
<tr>
<th>Relation</th>
<th>Babelnet</th>
</tr>
</thead>
<tbody>
<tr>
<td>is-a</td>
<td>optimisation, mathematical optimization, process</td>
</tr>
<tr>
<td>has-kind</td>
<td>glowworm swarm optimization, interprocedural optimization, entry form</td>
</tr>
<tr>
<td></td>
<td>optimization, partition alignment, compiler optimization</td>
</tr>
<tr>
<td>has-instance</td>
<td>loop optimization, loop unrolling</td>
</tr>
</tbody>
</table>

Several relations listed in Table 2 were not found in the target ontology, so they were added to it and subsequently checked whether the application of these changes improve the quality of the
ontology. Figure 4 provides a comparison of the relations of the “optimization” node in the target ontology before and after executing the knowledge transfer process.

![Comparison knowledge integration: a) Original Relations, b) Ontological dictionary relations](image)

**Figure 4.** Comparison knowledge integration: a) Original Relations, b) Ontological dictionary relations

As shown in Figure 4a, only two relations (optimization subclassOf: “Thing” and optimization subclassOf: “action”) were connected with the “optimization” node. On the other hand, in Figure 4b some relations are transferred from the ontological dictionary to the target ontology. We can clearly observe that the process provided a better structure and knowledge to the target ontology. This is because both the extracted relations, properties and individuals provide missing information that has been validated by experts.

The same process performed for the “optimization” class was carried out for all other nodes. As a result, 551 classes and 120 individuals were automatically transferred from Babelnet to the target ontology. Table 3 shows some examples of the knowledge transferred from the Babelnet ontological dictionary.
Table 3. Examples of items transferred from Babelnet

<table>
<thead>
<tr>
<th>Nodes (classes)</th>
<th>Nodes (individuals)</th>
</tr>
</thead>
<tbody>
<tr>
<td>support system</td>
<td>conditional relation</td>
</tr>
<tr>
<td>subClassOf Thing</td>
<td>individualOf logical relation</td>
</tr>
<tr>
<td>search engine</td>
<td>reflexiveness individualOf logical relation</td>
</tr>
<tr>
<td>subClassOf software</td>
<td>single-valued function individualOf</td>
</tr>
<tr>
<td>conjunction</td>
<td>mathematical relation</td>
</tr>
<tr>
<td>subClassOf connection</td>
<td>logical relation</td>
</tr>
<tr>
<td>operation research</td>
<td>NATO double-track decision individualOf</td>
</tr>
<tr>
<td>subClassOf research</td>
<td>decision making</td>
</tr>
<tr>
<td>object-oriented programming</td>
<td>semantic search individualOf search engine</td>
</tr>
<tr>
<td>subClassOf Thing</td>
<td>desktop search individualOf</td>
</tr>
<tr>
<td>simulation software</td>
<td>search engine</td>
</tr>
<tr>
<td>subClassOf application</td>
<td>individualOf search engine</td>
</tr>
<tr>
<td>optimality criterion</td>
<td>residual sum of squares individualOf</td>
</tr>
<tr>
<td>subClassOf criterion</td>
<td>optimality criterion</td>
</tr>
<tr>
<td>modeling</td>
<td>robotics simulator individualOf simulation software</td>
</tr>
<tr>
<td>subClassOf representation</td>
<td>technology CAD individualOf simulation software</td>
</tr>
<tr>
<td>simulation</td>
<td>high-throughput computing individualOf</td>
</tr>
<tr>
<td>subClassOf modeling</td>
<td>computing</td>
</tr>
<tr>
<td>cosine function</td>
<td>sine individualOf cosine function</td>
</tr>
<tr>
<td>subClassOf transcendental function</td>
<td></td>
</tr>
<tr>
<td>confidence interval</td>
<td></td>
</tr>
<tr>
<td>subClassOf statical method</td>
<td></td>
</tr>
<tr>
<td>In addition, the knowledge transfer process was executed using the third-party ontology as source of knowledge. As a result, 74 classes and 324 individuals were automatically transferred from the third-party ontology to the target ontology. Table 4 includes some examples of the knowledge transferred.</td>
<td></td>
</tr>
</tbody>
</table>
Table 4: Examples of items transferred from our third-party ontology

<table>
<thead>
<tr>
<th>Node (classes)</th>
<th>Node (individuals)</th>
</tr>
</thead>
<tbody>
<tr>
<td>preferenceModel subclassOf thing</td>
<td>basic ASF individualOf achievement scalarizing function</td>
</tr>
<tr>
<td>achievement scalarizing function subclassOf preference model</td>
<td>composite scalarizing functions individualOf achievement scalarizing function</td>
</tr>
<tr>
<td>decision rules subclassOf preference model</td>
<td>GUESS-ASF individualOf achievement scalarizing function</td>
</tr>
<tr>
<td>fuzzy logic subclassOf preference model</td>
<td>STOM-ASF individualOf achievement scalarizing function</td>
</tr>
<tr>
<td>knee point subclassOf preference model</td>
<td>Tchebycheff ASF individualOf achievement scalarizing function</td>
</tr>
<tr>
<td>light beam search subclassOf preference model</td>
<td>weighted euclidean distance ASF individualOf achievement scalarizing function</td>
</tr>
<tr>
<td>objective relative importance subclassOf PreferenceModel</td>
<td>Wierzbicki ASF individualOf achievement scalarizing function</td>
</tr>
<tr>
<td>outranking relation subclassOf preference model</td>
<td>DRSA individualOf decision rules</td>
</tr>
<tr>
<td>polyhedral cone based subclassOf preference model</td>
<td>fuzzy integral individualOf fuzzy logic</td>
</tr>
<tr>
<td>ranking subclassOf PreferenceModel</td>
<td>fuzzy measure individualOf fuzzy logic</td>
</tr>
<tr>
<td>set preference relation subclassOf preference model</td>
<td></td>
</tr>
<tr>
<td>utility function subclassOf preference model</td>
<td></td>
</tr>
</tbody>
</table>

As mentioned in Section 3, the fourth stage of the proposed methodology is dedicated to validation of the quality of the ontology with the help of evaluation metrics. To do so, we have used the Ontology fixer tool\(^5\) that displays a radar chart using several metrics and computes the area of the polygon formed when the dots (quality metric values) are connected between them. The larger the
area, the higher the quality of the ontology. Figure 5 shows the radar charts before and after knowledge transfer, both with Babelnet and the third-party ontology.

![Radar charts](image)

**Figure 5.** Quality metrics results: a) Initial target ontology, b) Result Babelnet knowledge integration, c) Result third-party and ontology knowledge integration

As evidenced in Figure 5b, the quality of the ontology resulting from integrating only the knowledge of the ontological dictionary improved 6% (6 points) compared to the initial ontology. On the other hand, Figure 5c shows that the quality improves to a score of 31.81 when the knowledge of third-party ontology is also incorporated. CBOnto was the metric that shows the greatest improvement in knowledge transfer from the ontological dictionary, which implies an enhancement in the coupling of the ontology. That is, a more complete and specialised ontology was obtained using the methodology proposed in the paper, which means that the knowledge exchange process successfully filled the existing class-based concept gaps. Furthermore, there was also an increase in the CROnto and RCONto metrics related to the number of individuals per class when transferring knowledge from the third-party ontology. Unlike when we used Babelnet as the
source of knowledge, the amount of concrete knowledge (number of individuals) has increased significantly. Finally, we highlight that a quality assessment is not performed with respect to the ontology competency question metrics (CBOnto is a maintainability metric, CROnto is a functional adequacy metric and RCOnto is a knowledge base metric). If we had used metrics capable of evaluating this dimension, the quality of the target ontology would achieve a better-quality score.

Indeed, the possibility of adding other external sources of knowledge, in addition to those used in this case study, could lead to an improvement in the ontology quality metrics. Moreover, the process could be run interactively so that experts could specifically select the most relevant knowledge for transfer. This would result in an even more robust ontology. The following subsection summarises the advantages of our contribution, analysing the benefits and computational requirements of our approach.

**Advantages of our proposal**

The usage of ontology learning process is useful to reduce the efforts in creating ontologies to model knowledge domains from textual sources. However, ontologies generated through these algorithms often contain some limitations such as missing knowledge that could not be successfully extracted from textual sources and therefore needs to be fixed. The ontology correction processes developed previously, only considered information related to structural, functional, adaptability, maintainability or operability quality parameters. However, the correction of semantic aspects was not possible since no semantic validation mechanisms had been introduced. In this work, we introduce a knowledge exchange methodology that takes advantage of available knowledge from different sources such as ontology dictionaries and third-party OWL ontologies for fixing purposes. The mechanism takes advantage of a simple ontology matching scheme which is able to identify relevant knowledge from third-party sources (ontologies or semantic graphs) to fix automatically generated ontologies.

Since the knowledge sources used in this methodology have a structured nature (e.g. Babelnet API), knowledge collection, validation and exchange can be done efficiently and using low computational resources. As long as the methodology can be applied semi-automatically, it can effectively reduce the workload of experts in the ontology fixing and quality improvement processes.

**Conclusions and future work**

In this study we introduce a strategy for the validation and transfer of knowledge from ontologies created by third parties (ontological repositories) and ontological dictionaries (e.g. Babelnet, Wordnet) to semi-automatically or automatically generated ontologies that may contain semantic problems affecting their quality and structure. The algorithm comprises an ontology matching process that only retrieves information that could be valuable to enrich the content, improve the quality of the ontology and/or validate that the ontology content is appropriate within the domain.

With the support of the OntologyFixer tool developed in a previous research work, we can assess the quality of the target ontology by comparing radar charts that combine different metrics. The area of the radar chart displays the overall quality and allows the user to decide whether the transferred knowledge has a positive or negative impact on the target ontology. Likewise, by using OntologyFixer we can further improve the quality of the ontology by applying the different quick fixes to address possible remaining issues.
Experimental results have demonstrated the suitability of the methodology for addressing some issues of an automatically generated ontology. The knowledge identification and selection process automatically discovered a collection of Babelnet synsets and third-party ontology nodes (records) that can be transferred to the target ontology. This shows the great potential of using exchange of semantic resources to improve the quality of a target ontology. So, a large number of third-party ontologies shared in public repositories can be used as sources of knowledge for improving other ontologies.

Finally, the knowledge transfer process is computationally simple. This is because (i) it is based on a computationally simple ontology matching method, (ii) the knowledge sources considered in this methodology have a structured nature (graph) and, finally, (iii) the graph structure of knowledge sources matches (in the case of ontologies) or is very similar (in the case of semantic networks) to the target knowledge bases. The simplicity of this method clearly contrasts with the alternative of having an expert available to manually fix the ontologies.

This work may be extended to include new sources of knowledge that can be used for knowledge transfer. In particular, we believe that knowledge engineering may continue to benefit from methods and tools based on software engineering. In this sense, we believe that there is a possibility to adopt methods and technologies used in software engineering (e.g. SQL schema designs of relational databases, UML/XMI\textsuperscript{‡‡} domain diagrams, etc.) for knowledge transfer processes and/or improvement of the quality of ontologies. Furthermore, the use of semantic graphs allows additional simple validations such as ensuring that the different words or n-grams that make up a synset do not appear as names of different classes or individuals. Finally, the use of complex ontology matching schemes and the comparison of their performance should also be addressed as future work.

\textsuperscript{‡‡} Available at https://www.omg.org/spec/DD/1.0/
References

Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 91–100.