Predicting the Score of Atomic Candidate OWL Class Axioms

Ali Ballout, Andrea G B Tettamanzi, Célia da Costa Pereira

Predicting the Score of Atomic Candidate OWL Class Axioms

Ali Ballout
Université Côte d'Azur, I3S, Inria
Sophia Antipolis, France
ali.ballout@inria.fr

Andrea G. B. Tettamanzi
Université Côte d'Azur, I3S, Inria
Sophia Antipolis, France
andrea.tettamanzi@unice.fr

Célia da Costa Pereira
Université Côte d'Azur, I3S, CNRS
Sophia Antipolis, France
celia.pereira@unice.fr

***Abstract*—Candidate axiom scoring is the task of assessing the acceptability of a candidate axiom against the evidence provided by known facts or data. The ability to score candidate axioms reliably is required for automated schema or ontology induction, but it can also be valuable for ontology and/or knowledge graph validation. Accurate axiom scoring heuristics are often computationally expensive, which is an issue if you wish to use them in iterative search techniques like level-wise generate-and-test or evolutionary algorithms, which require scoring a large number of candidate axioms. We address the problem of developing a predictive model as a substitute for reasoning that predicts the possibility score of candidate class axioms and is quick enough to be employed in such situations. We use a semantic similarity measure taken from an ontology's subsumption structure for this purpose. We show that the approach provided in this work can accurately learn the possibility scores of candidate OWL class axioms and that it can do so for a variety of OWL class axioms.**

***Index Terms*—Ontology Learning, OWL Axioms, Concept Similarity**

I. Introduction and Related Work

An ontology is the explicit representation of components of a shared conceptualization [13]. In machines, an ontology is a vocabulary which is used by said machine in the representation of knowledge. An AI knowledge-based system would take an ontology as its universe of components and their relations and derive new implicit knowledge within that universe.

As for the components of an ontology, Z. Dragisic [9] defines them as follows:

- Concepts (classes) - the types of objects in a domain or area.
- Relations (properties, roles) - the relations between two concepts/classes.
- Instances (individuals) - instances of concepts/classes.
- Axioms - model sentences that are always correct in the domain. They're often utilized to represent information that cannot be formally specified by the other components [8].

Ontologies play a vital role in data and knowledge integration by making a common schema available [14]. Unfortunately, ontology construction is extremely expensive, in terms of both time and resources, and dependent on the availability of knowledge experts [14], [23]. The construction of an ontology for a certain field requires the contribution of a knowledge engineer and of an expert in that specific field. This dependency persists throughout the lifetime of an ontology as an expert is required to develop and expand it as new requirements arise. To overcome such a bottleneck in knowledge acquisition, the field of ontology learning was conceived. Ontology Learning [14], [16], [18] is the task consisting of the automatic generation of ontologies. Ontology learning includes a variety of techniques and those are grouped into:

- Linguistic techniques - Natural language processing mostly used in the preprocessing of data, and some learning tasks such as term extraction [2].
- Statistical techniques - such as data mining and information retrieval methods used to extract terms and associations between them [11], [12].
- Inductive logic programming (ILP) is a branch of machine learning that uses logic programming to generate hypotheses based on prior knowledge and a set of examples [10], [15], [20].

Each one of these technique groups is involved in one or more of the stages of ontology learning, those stages being preprocessing, term and concept extraction, relation extraction, concepts and relations hierarchies, axioms schemata and general axioms [2]. Linguistic techniques can have a role in almost all the stages, but as we mentioned they are mostly used for data preprocessing.

As for statistical techniques, they are methods that rely entirely on the statistics of the textual resources without any concern for the semantics. One such method is the possibilistic (statistics-based) heuristic detailed in [25], where the authors have developed a possibilistic framework for the Web ontology language (OWL 2) to test axioms against evidence expressed in the Resource Description Framework (RDF). The heuristic uses support, confirmations, and counterexamples to define possibility and necessity of an axiom and an acceptance/rejection index combining both of them. They test the developed theory on `SubClassOf` axiom testing against the DBpedia[1] database. The results of their experiments showed that the method was suitable for axiom induction and ontology learning [24]. The suggested heuristic has the drawback of being time consuming; this was addressed in a revision of the method that added a time cap for the querying process, of course at the cost of a little increase in error rate reaching 3.96% [26]. The number of axioms tested was 5050 and it took almost 342 CPU hours and a half with an average of 244 s per axiom **with time capping**. This is a significant improvement, considering that testing the same amount of axioms without a time cap would have taken approximately 2,027 CPU days. Another limitation is that lack of support means an inconclusive judgement, as

[1]https://www.dbpedia.org/resources/ontology/

the method queries for confirmations and counterexamples, and sometimes it might find none. The third drawback is that being a statistical approach it relies solely on instance data from the dataset, data that is prone to errors.

In comparison, ILP techniques are a sub-field of machine learning that follow exhaustive statistical or linguistic processing. One such example is [17], where the authors describe a method for the automated induction of fuzzy ontology axioms which follows the machine learning approach of ILP named `SoftFOIL`. One of `SoftFOIL`'s limits is a result of its sequential covering strategy. As it uses a greedy search to find rules, it does not guarantee to find the smallest or best set of rules that explain the training examples. Another is susceptibility to being trapped in a loop while searching for the best rule.

A new emerging group of techniques, is a hybrid breed that takes advantage of combining classical ILP and statistical machine learning. To stay in the context of the previous examples, a model would be trained with axioms having their scores assigned by a statistical method such as [25], as well as using a similarity measure that results from the logical processing of the data. This is exactly what was done in [19], where the authors modified the support vector clustering algorithm to attempt to predict the possibilistic score of **OWL** axioms. They used the heuristic from [25] as the scorer, and used a model originally developed for inferring the membership function for fuzzy sets. As a similarity measure they used one specific for subsumption axioms based on semantic considerations and reminiscent of the Jaccard index. The predictor performance was poor in terms of root mean square error (RMSE) scoring 0.572 (Table 3 in [19]). In addition, the authors mentioned that they found a group of axioms that were *hard* to predict, and could not find a reason that explains why. This method was computationally far more efficient than [24], yet it was still reliant on the instances in the data set and querying them to construct the similarity measure, meaning that even though it is an improvement, it still falls victim to the same problems. The authors explicitly mention that a major weakness of their method is that training such a model consumes a significant amount of resources.

Our work addresses the shortcomings of the previous techniques that heavily rely on error-prone instance-dependent statistics. It is also able to predict the scores of multiple types of axioms and is not bound to simply `subsumption`. We propose a method that can be used as a building-block or an extension/plug-in to other existing statistical analysis or ILP options, such as DL-Learner [4], to allow faster execution while maintaining high scoring accuracy, while still having the ability to perform as a simpler stand-alone scorer. The method works by training a model on a set of atomic class axioms scored by an algorithm, in this case [25]. This enables the model to predict the score of any new atomic (consisting of a single concept on each side) candidate axiom. We experimented using multiple machine learning methods, and compared our work to the state of the art [19] that aims to achieve the same goal.

This paper is structured as follows: Sect. II provides some background about both axiom scoring and concept semantic similarity which are both prerequisites to training the models. As for Sect. III it lays out the methodology explaining how the axioms were extracted and scored, how the semantic measure we use was developed, and also how an axiom based vector space was modeled leading to the prediction of the scores. We detail our experiments including a comparison with the method presented in [19] in Sect. IV then present the results while listing our observations and findings. We end the paper with some notes and conclusions.

II. Background

A. Axiom Scoring

Axiom scoring can be done using statistical analysis, we mentioned in the previous section a heuristic [25] that does this using the theory of possibility. Possibility theory [27] is a mathematical theory of epistemic uncertainty. Its central notion is that of a possibility distribution which assigns to each elementary event a degree of possibility ranging from 0 (impossible, excluded) to 1 (completely possible, normal). A possibility distribution π induces a *possibility measure* Π, corresponding to the greatest of the possibilities associated to an event and the dual *necessity measure* N, equivalent to the impossibility of the negation of an event.

Here we provide a brief explanation of the theory behind the scoring. Given a candidate OWL 2 axiom ϕ, expressing a *hypothesis* about the relations holding among some entities of a domain, we wish to evaluate its credibility, in terms of possibility and necessity, based on the *evidence* available in the form of a set of facts contained in an RDF dataset $\mathcal{K}$.

The content of ϕ is defined as a (finite) set of *basic statements* ψ which are logical consequences of ϕ, i.e., $\phi \models \psi$. The open-world hypothesis (OWA) is fully taken into account. Therefore, given $\mathcal{K}$, for each such ψ, there are three cases:

1) $\mathcal{K} \models \psi$: in this case, we will call ψ a *confirmation* of ϕ;
2) $\mathcal{K} \models \neg\psi$: if so, we will call ψ a *counterexample* of ϕ;
3) $\mathcal{K} \not\models \psi$ and $\mathcal{K} \not\models \neg\psi$: in this case, ψ is neither a confirmation nor a counterexample of ϕ.

If we denote by u_ϕ the *support* of ϕ, which is the cardinality of its content, by u_ϕ^+ the number of confirmations of ϕ and by u_ϕ^- the number counterexamples of ϕ, the possibility and the necessity of candidate axiom ϕ may be defined as follows:

- if $u_\phi > 0$,

$$\Pi(\phi) = 1 - \sqrt{1 - \left(\frac{u_\phi - u_\phi^-}{u_\phi}\right)^2}; \quad (1)$$

$$N(\phi) = \begin{cases} \sqrt{1 - \left(\frac{u_\phi - u_\phi^+}{u_\phi}\right)^2}, & \text{if } u_\phi^- = 0, \\ 0, & \text{if } u_\phi^- > 0; \end{cases} \quad (2)$$

- if $u_\phi = 0$, $\Pi(\phi) = 1$ and $N(\phi) = 0$, i.e., we are in a state of maximum ignorance, given that no evidence is

available in the RDF dataset to assess the credibility of ϕ.

The possibility and necessity of an axiom can then be combined into a single handy acceptance/rejection index

$$\begin{aligned} \mathrm{ARI}(\phi) &= N(\phi) + \Pi(\phi) - 1 = N(\phi) - N(\neg\phi) \\ &= \Pi(\phi) - \Pi(\neg\phi) \in [-1, 1], \end{aligned} \tag{3}$$

because $N(\phi) = 1 - \Pi(\neg\phi)$ and $\Pi(\phi) = 1 - N(\neg\phi)$ (duality of possibility and necessity). A negative $\mathrm{ARI}(\phi)$ suggests rejection of ϕ ($\Pi(\phi) < 1$), whilst a positive $\mathrm{ARI}(\phi)$ suggests its acceptance ($N(\phi) > 0$), with a strength proportional to its absolute value. A value close to zero reflects ignorance about the status of ϕ.

B. Ontological Semantic Similarity

Semantic similarity is a notion used to define a distance between terms or concepts based on meaning or semantics. It includes in its calculation only relations of the type **IS-A** [3]. It is often confused with semantic relatedness, for example a *train* and *train tracks* are functionally complementary, where as a *train* and an *airplane* are functionally similar. The latter is an instance of semantic similarity where the relatedness of both terms is based on the defining features they share where both are vehicles [3], [6]. Most semantic similarity measures that rely on a structured ontology, are based on path lengths between concepts as well as depth of concept nodes in an **IS-A** hierarchy. As for information-based measures they use information content (**IC**) of concept nodes derived from the ontology hierarchy structure and corpus statistics [1].

The similarity measure we utilize in our work is the one detailed in [5], [6], under the subsection titled *Ontological Approximation*. The idea is to calculate the ontological distance between two concepts by using the subsumption path length. Following is the general definition:

$$\begin{gathered} \forall (t_1, t_2) \in H^2, \\ D_H(t_1, t_2) = min_t(l_H(\langle t_1, t\rangle) + l_H(\langle t_2, t\rangle)) \\ = min_t \left(\sum_{\{x \in <t_1,t>, x \neq t_1\}} 1/2^{d_H(x)} + \sum_{\{x \in <t_2,t>, x \neq t_2\}} 1/2^{d_H(x)} \right) \end{gathered} \tag{4}$$

Formula 4 translates to: for all type pairs t_1 and t_2 in an inheritance hierarchy *H*, the *ontological distance* between t_1 and t_2 in the inheritance hierarchy *H* is the minimum of the sum of the lengths of the subsumption paths between each of them and a common super type. And the length of the subsumption path between a type t_1 and its direct super type t is equal to $1/2^{d_H(t)}$ with $d_H(t)$ being the depth of t in *H*.

The authors of [6] implemented this measure as part of a larger ontology-based search engine tool named `Corese` [7][2]. This is the tool that has been used as means of extracting the semantic similarity between concepts in our method.

We propose an extension to this similarity to present similarities between axioms, this process is detailed in Sect. III-B.

[2]https://project.inria.fr/corese/

C. Instance Semantic Similarity

This similarity is used in the state of the art method detailed in [19]. The support vector clustering method the authors used requires a kernel function which was assumed to return the similarity between two axioms.

Similar to the ontological similarity, it is based on the semantics of axioms and not on their syntax. The measure S is defined with values in $[0, 1]$, satisfying the following desirable properties: for all axioms ϕ and ψ,

1) $0 \leq S(\phi, \psi) \leq 1$;
2) $S(\phi, \psi) = 1$ if and only if $\phi \equiv \psi$;
3) $S(\phi, \psi) = S(\psi, \phi)$.

The similarity is defined based on a fuzzy implication operator $\Longrightarrow$: we can say two axioms ϕ and ψ are similar to the extent that $\phi \Rightarrow \psi$ and $\psi \Rightarrow \phi$.

A fuzzy definition, based on the Herbrand semantics of the axioms, might be the following:

$$\Longrightarrow (\phi, \psi) = \frac{\|\{\mathcal{I} : \mathcal{I} \models \neg\phi \vee \psi\}\|}{\|\Omega\|} = \frac{\|[\neg\phi] \cup [\psi]\|}{\|\Omega\|} = \frac{\|\overline{[\phi]} \cup [\psi]\|}{\|\Omega\|}, \tag{5}$$

where $[\phi]$ denotes the set of the models of ϕ.[3]

One problem is that instead of exactly computing the numerator, which would require to count the models and counter models of both axioms being compared, a not so convincing rough approximation of it was used. This approximation was obtained by replacing models and counter models with instances occurring in the RDF dataset that confirm or contradict the two axioms. This renders the similarity bound to and limited by instance data, which means it is prone to errors as well as unusable in case instance data is scarce or non existent.

The similarity addresses and works with subsumption axioms only, of the form $C \sqsubseteq D$, where C and D are two OWL class expressions, and their negation $C \not\sqsubseteq D$. Given an individual a occurring in an RDF dataset, we may say that

- a confirms axiom $C \sqsubseteq D$ (contradicts $C \not\sqsubseteq D$) iff $C(a) \wedge D(a)$;
- a contradicts axiom $C \sqsubseteq D$ (confirms $C \not\sqsubseteq D$) iff $C(a) \wedge \neg D(a)$.

Instead of counting the models of an axiom, the individuals that confirm it are counted; instead of counting its counter models, the individuals that contradict it. Given four OWL class expressions A, B, C, and D, the similarity which is also the degree to which axiom $A \sqsubseteq B$ implies axiom $C \sqsubseteq D$ can be computed as

$$S(A \sqsubseteq B, C \sqsubseteq D) = \frac{\|[A] \cap [B] \cup [C] \cap [D]\|}{\|[A] \cup [C]\|}. \tag{6}$$

In order to predict the ARI of subsumption axioms in this case, similarities between positive and negated subsumption axioms need to be computed.

The similarity between two candidate OWL axioms of the form $A \sqsubseteq B$ and $C \sqsubseteq D$, can be computed using SPARQL counting queries. For instance, the denominator $\|[A] \cup [C]\|$ may be computed by

```
SELECT (count(DISTINCT ?x) AS ?n)
WHERE { { ?x a A . } UNION { ?x a C . } },
```
(7)

[3]A model of ϕ is an interpretation that satisfies ϕ and a counter-model is an interpretation that does not satisfy ϕ.

whereas the numerators (which are four when comparing two axioms and their negations) may be computed by SPARQL queries of the form

$$\begin{array}{l}\texttt{SELECT (count(DISTINCT ?x) AS ?n)}\\ \texttt{WHERE}\ \{\ \{\ Q([A])\ .\ Q([B])\ .\ \}\\ \texttt{UNION}\{\ \{\ Q([C])\ .\ Q([D])\ .\ \}\ \},\end{array} \tag{8}$$

where

$$\begin{aligned} Q([X]) &= \texttt{?x a}\ X,\\ Q([\overline{X}]) &= \texttt{FILTER NOT EXISTS ?x a}\ X. \end{aligned}$$

This is a slow process, and for every candidate axiom you would need to calculate the similarity for it and its negation. Since the queries used are counting queries similar to those used in the scoring heuristic this means it also suffers from the same limitations such as heavy computation cost and instance dependency.

III. Methodology

In an **OWL** ontology containing an inheritance hierarchy of concepts formed by the subsumption axiom *rdfs:subClassOf*, our aim is to predict an acceptability score for a candidate atomic class axiom by learning a set of previously scored axioms of the same type, the score used is the one detailed in II-A. A model is built for each type of axiom to be predicted, this means that the method is repeated for the number of axiom types being dealt with. To measure the similarity between (candidate) axioms, we construct a similarity measure by extending the *ontological distance* discussed in II-B, which is defined among concepts, not axioms. To this end, we consider the following necessary steps:

1) **Axiom extraction and scoring**: This step constitutes the creation of the set of scored axioms of a certain type to be learned. We either use a ready scored set such as we did for our comparison, or we score a new generated set.
2) **Semantic measure construction and assignment**: This step involves the retrieval of concepts used in our set of axioms, and their ontological distance from the ontology. Followed by extending that similarity to those axioms. This was done by calculating a single value that represents the similarity between each pair of axioms, by applying a function such as *Average* to the ontological distances of concepts in those axioms.
3) **Axiom base vector space modeling**: This step focuses on using axiom similarity measures as weights, each axiom can be represented as a vector in an axiom based vector space.
4) **Score prediction**: This step is dedicated to training a Machine Learning model with the data set (vector space model in addition to the scores) and predicting the acceptability score of new candidate axioms.

We start by collecting the set of scored axioms we plan to train and test our method with. After that, we query the ontology to retrieve the ontological distances between all concepts. Then similarity measures between axioms will be derived from those distances and assigned as weights to represent the axioms as vectors in an axiom-based vector space. Machine learning models are used in the end to learn the set of scored axioms with their similarity weights and predict the acceptability score of a candidate axiom. We will consider subClassOf axioms for the comparison with [19] since that is the only type of axiom they can address, and their dataset which we use for said comparison is made of subClassOf axioms. We will also use disjointWith axioms for our experiment to show that we can apply our method to all atomic owl class axiom types, as well as highlight that no leakage or bias is present from utilizing the subclass of hierarchy.

A. Axiom Extraction and Scoring

In this paper, we consider two approaches. First generating a number of atomic class candidate axioms randomly. The generated candidates are filtered for duplicates. We then make sure non of the candidates already exist in the ontology explicitly or implicitly, using the search engine and its reasoner, then we score them.

The other, which we prefer and use for controlled tests is to query existing axioms. To make sure we have positive scores we query the ontology for existing axioms and score them. The same can be said for negatively scored axioms, we can query the counter type and score it as if it were the first. For example subClassOf and disjointWith are counter types so if $disjointWith(C_1C_2)$ has a positive score, $subClassOf(C_1C_2)$ will have a negative one. This is bound by how many axioms exist to be queried (after inferring implicit axioms).

Query 1 is used to extract existing axioms of both types and labeling them to be scored after with the heuristic. The search engine used is Corese [7] and the ontology is Dbpedia. After the extraction of the axioms, we select an equal amount of both classes.

```
SELECT ?class1 ?class2 ?label WHERE {
{ ?class1 a owl:Class . ?class2 a owl:Class . ?class1
    rdfs:subClassOf ?class2
filter (!isBlank(?class1) && !isBlank(?class2))
filter (?class1 != ?class2)
bind(1.0 as ?label) }
Union{ ?class1 a owl:Class . ?class2 a owl:Class .
    ?class1 owl:disjointWith ?class2
filter (!isBlank(?class1) && !isBlank(?class2))
filter (?class1 != ?class2)
bind(0.0 as ?label) }}
```

Query 1: Axiom extraction

The second step is to score, this is done by simply inputting the extracted axioms as a text file using the possibilistic heuristic [25], and receiving an output file containing the scores (ARI), it should be noted that the process is very slow thus the need for a method such as ours.

For disjointWith axioms, possibility only is considered since necessity is meaningless in the case of this axiom and incalculable. So the score is between 0 and 1.

$Concepts$	C_0	C_1	$\dots$	C_n
C_0	1	$S_{0,1}$	$\dots$	$S_{0,n}$
C_1	$S_{1,0}$	1	$\dots$	$S_{1,n}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
C_{n-1}	$S_{n-1,0}$	$S_{n-1,1}$	$\dots$	$S_{n-1,n}$
C_n	$S_{n,0}$	$S_{n,1}$	$\dots$	1

Matrix 1: Concept similarity matrix

$Scores$	$Axioms$	A_0	A_1	$\dots$	A_m
$score_0$	A_0	1	$S_{0,1}$	$\dots$	$S_{0,m}$
$score_1$	A_1	$S_{1,0}$	1	$\dots$	$S_{1,m}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$score_{m-1}$	A_{m-1}	$S_{m-1,0}$	$S_{m-1,1}$	$\dots$	$S_{m-1,m}$
$score_m$	A_m	$S_{m,0}$	$S_{m,1}$	$\dots$	1

Matrix 2: Axiom similarity matrix with labels

B. Semantic Measure Construction and Assignment

To be able to assign similarity measures between axioms, we need to retrieve the ontological distances between all classes. Using *Corese* in which the ontological distance metric is implemented, this translates into a function added to the *SPARQL* query. Query 2 retrieves three columns, the first two contain the combination of all classes with the third containing the ontological distance denoted by similarity. Blank nodes are ignored.

```
select * (kg:similarity(?class1, ?class2) as
    ?similarity) where {
?class1 a owl:Class . ?class2 a owl:Class
filter (!isBlank(?class1) && !isBlank(?class2)) }
```

Query 2: Class ontological distance retrieval

After retrieving the table of similarities it is pivoted to construct a symmetric $n \times n$ matrix where the first column and the first row are the classes and the cells are the similarities between them with a diagonal of only *1's* since as we mentioned the similarity between a class *C* and itself is *1*. The shape of the concept similarity matrix is illustrated in Matrix 1.

From this matrix, we can derive the similarity between axioms. The goal is to end up with a similar square symmetric matrix of the shape $m \times m$, m being the number of axioms, that has axioms instead of concepts as both first row and column, and the cells would be the similarities between a pair of axioms. Exactly as in the case of the concept similarity matrix, the diagonal of this matrix will contain only *1's* as the similarity between an axiom *A* and itself is *1*.

In order to construct this axiom similarity matrix M_A depicted in Matrix 2 alongside the labels, we use Algorithm 1. The algorithm iterates over the set of labeled axioms T_A which we extracted in Section III-A, comparing each axiom A_i to all other axioms A_j in T_A having j increment from i to length of $T_A - 1$ after each iteration of i. This is so we avoid redundant calculations. While comparing axioms A_i and A_j, we first deal with the concept on the left side of each axiom, so the left

Algorithm 1 Constructing the matrix of axiom similarities

Require: All concepts included in axioms in T_A be present in concept similarity matrix M_C
Ensure: $0 \leq S \leq 1$ ▷ S is the similarity between 2 axioms
  $M_C \leftarrow Concept\ similarity\ matrix$
  $T_A \leftarrow Set\ of\ labeled\ axioms$
  $M_A \leftarrow Axiom\ similarity\ matrix\ to\ be\ filled$
  for $i = 0 \rightarrow Length(T_A) - 1$ **do**
    for $j = i \rightarrow Length(T_A) - 1$ **do**
      $S_L \leftarrow M_C[A_i[L], A_j[L]]$
      $S_R \leftarrow M_C[A_i[R], A_j[R]]$
      $S^1 \leftarrow Avg(S_L, S_R)$ ▷ Experiments were done with (min, Avg)
      if Axiom Type is Disjointness or Equivalence **then**
        $S_L \leftarrow M_C[A_i[L], A_j[R]]$
        $S_R \leftarrow M_C[A_i[R], A_j[L]]$
        $S^2 \leftarrow Avg(S_L, S_R)$
      else
        $S^2 \leftarrow 0$
      end if
      $S \leftarrow Max(S^1, S^2)$
      $M_A.Add(A_i, A_j, S)$
    end for
  end for
  Note: For Symmetric axioms, we have to compare both ways which is what we do inside the if statement for Disjointness and Equivalence axioms.

concept of A_i denoted by $A_i[L]$ and that of A_j denoted by $A_j[L]$. To retrieve S_L the similarity between those concepts from M_C, we search in the first row of the concept similarity matrix M_C for concept $A_i[L]$, and in the first column of M_C for concept $A_j[L]$. $M_C[A_i[L], A_j[L]]$, the intersection between the row where $A_i[L]$ was found, and the column where $A_j[L]$ was found represents the left side similarity between axioms A_i and A_j. The same process is repeated for the right side, and then the axiom similarity S is calculated as a function between S_L and S_R. In this work we applied two functions and they were *minimum* and *average*. After calculating S in each iteration of j it is added to M_A in the cell where the row is the index of A_i and column the index of A_j.

C. Axiom Base Vector Space Modeling

We define a vector-space model to represent axioms as vectors. The number of dimensions d of our vector space is equal to the number of axioms we have in the labeled set T_A. Each axiom can be represented as a vector V in this d-dimensional space. Now that we have the similarities between axioms in our ontology, looking at the axiom-similarity matrix it would be intuitive to consider each row of the matrix as a vector V in a vector space where the dimensions d are the columns which are the axioms themselves having as weights the similarities S. Thus Algorithm 1 is utilized whenever an axiom is generated or a new candidate axiom is suggested and will encode said axiom into a vector V in this vector space.

D. Score Prediction

After constructing our vector space and representing axioms as vectors, we consider the set of scored axioms as a set of vectors. It is now possible to apply regression machine learning methods on the vector space representation of an axiom base. We used a range of methods throughout our experiments such as random forests, Support vector regressor, and neural networks. Trees and random forests were mostly used to check that there was no bias during the prediction,

since they allow us to interpret our predictive model easily and visually analyse the decisions.

To avoid information leakage since the matrix is symmetric, considering the size of the matrix is $n \times n$, all models would be trained using an $m \times m$ sub-matrix of the axiom similarity matrix with the labels, and then tested on a $z \times m$ sub-matrix. This means an axiom vector V will have m dimensions (features) which are the ones used to train the model, regardless what the number of dimensions (features) in the full matrix is. In other words the symmetric part of the matrix equivalent to z (the columns of n outside the range of m) is discarded. The model's goal is to predict the score of a candidate axiom, which is represented as a vector V in our vector space having m dimensions (features), which are the axiom's similarities with the axioms of the same type used to train the model. The score is a number between -1 and 1 for subClassOf and equivalence, and between 0 and 1 for disjointWith. where 1 represents the highest acceptability. Any scorer/scoring algorithm can be used, we decided on [25] to be able to compare our results with [19].

IV. Experiments and Results

For the experiments[4], the workstation that was used had the following hardware configuration:

- Dual CPUs: 2 × Intel(R) Xeon(R) CPU E5-2689 0 @ 2.60GHz base and 3.30 all core boost. With 8 cores and 16 threads per CPU for a total of 16 cores and 32 threads.
- A total of 32 GB of RAM memory with frequency 1600 MHZ distributed as 8 × 4 GB sticks with 4 sticks assigned to each CPU.
- 1 TB of NVME SSD storage with read and write speeds of up to 2000 MB per second.

A. Dataset Preparation

For our testing and experiments, and to comply with both the scorer and the experiment of [19], the ontology used is Dbpedia. This also allows us to show how our method would perform in a real-world case.

We used two datasets for our experiments, one for axiom type subClassOf and it is the one used in [19], and another for axiom type disjointWith which is a generated set of atomic disjointWith candidate axioms, as described in Sect. III-A, scored using [25].

For the axiom set used in [19], they have 722 subClassOf axioms and their negations, making it 1444 axioms. The negations are specific for that similarity and for the support vector clustering method that was applied in [19]. These negations serve no meaning if it is possible to predict the scores of the original 722 axioms. For this reason, and since we had the possibility of an axiom and its negation, we can use that to calculate the ARI using 3. Then the dataset we work this will be the 722 axioms with their ARI score, since we will not be using the same machine learning method nor similarity for our proposed solution.

We did not include tests of equivalentClass axiom types since the process of creating the axiom similarity matrix of that axiom type is the same as disjointWith. They are both symmetrical axioms and equivalentClass axioms are basically a two way subClassOf axiom. For that reason we decided to include our experiment on the type disjointWith.

For the set of axioms of type disjointWith, we load the Dbpedia OWL file into Corese. The OWL file contains no individual data which means faster processing in the search engine, corese reasoner was applied to it to deduce the maximum number of axioms. Positively scored disjointness axioms were obtained from the results of [22].[5] Negatively scored disjointWith axioms were existing subClassOf axioms, explained in Section III-A. In case the used ontology is already populated with explicit axioms, this would be done in an attempt to obtain a set of axioms that the scorer will not provide a score close to 0 i.e., a score that implies ignorance for lack of instance data. This allows us to learn a better model that makes better predictions and surpasses the limitations a statistical scorer can face. But in our case this would result in a small set of 1120 scored axioms. One of the issues the authors of [19] faced is that the dataset they were working with was too small and they point out that this had prevented them from running certain experiments. For this case we combined both approaches expressed in Section III-A. Randomly generating atomic candidates and checking that they do not already exist, we managed to score an additional 2748 axioms for a total of 3868. This will allow better testing to see how the method performs in real-world cases and on a different axiom type.

We will denote by T_A the axiom set being used. We will not be comparing the time needed to score the set of axioms since they both use the same scorer. Heuristic [25] takes a list of candidate axioms as input and provides as output a list of scored candidate axioms. It is a slow but precise heuristic that depends on instance data and counting queries.

After preparing the set of axioms, we have to produce the concept similarity Matrix **(CSM)** 1, a process we detailed in Section II-B. The processing time for creating the **(CSM)** is dependent on the number of concepts. In our tested dataset, the number of unique concepts was 762 and creating its **(CSM)** took 2.0362 seconds.

The next step is constructing the axiom similarity matrix **(ASM)**, and encoding each of the axioms as vectors V in our vector space. This step is detailed in the second half of Section III-B and in Section III-C. The algorithm applied to our axiom data set T_A is Algorithm 1, this is the same algorithm used to encode candidate axioms into the vector space, it is the equivalent of running queries 7 and 8 for the method proposed in [19] to retrieve the similarity. In contrast the algorithm we use is not based on instance counting and is much faster in comparison. Table I details the time required to complete the axiom similarity matrix. As observed the time needed to complete the task of building the *ASM* significantly

[4]All the data and code needed to replicate the experiments available at https://anonymous.4open.science/r/axiom-score-prediction-BC16

[5]https://bitbucket.org/RDFMiner/classdisjointnessaxioms/src/master/Results/ClassDisjointnessAxioms/

Similarity Method	Type	Set size	ASM	Candidate processing	NN	Random Forest	SVR	Modified SVC	Explained variance
Instance based	subClassOf	1444	649,440	1,799	0.35299	0.30707	**0.26721**	0.572	0.52652
Ontological based	subClassOf	722	**13.7275**	**0.01901**	**0.31442**	**0.30231**	0.33972	NA	**0.88859**
Ontological based	disjointWith	3868	129.9637	0.03359	0.23325	0.21754	0.23771	NA	0.73462

TABLE I: Time cost in seconds, performance scores in RMSE per model, and the explained variance score of the best performing model per experiment.

increases as the number of axioms processed increases, but we also know that it depends on the size of the *CSM* as well from the time needed to encode a single axiom into a vector. As the number of concepts increases the *CSM* being searched increases as it has the shape $n_{concepts} \times n_{concepts}$ and the number of cells n^2. We would note that the test scenario (3868 disjointWith axioms) presented in Table I is extreme, especially the case of Dbpedia, as the number of axioms needed for training a model does not need to be as large to achieve peak accuracy/results. It is possible with a dataset size of subClassOf experiment (722).

While constructing the ASM using the instance similarity method, we had to calculate the denominators of all axiom pairs as in Equation 6. We ran into an issue of lack of instance data, this means a 0 in the denominator which is obviously a huge problem. The authors of [19] address this by denoting any similarity between a pair of axiom where the denominator is 0 (lack of instances) with a 0 as well. This highlights the weakness of instance based similarity methods. We do not believe that simply saying the similarity between a pair of axioms is 0 because they do not share individual data.

B. Training and Testing

After obtaining our datasets as described in the previous section, we applied different regression methods as mentioned in Section III-D to check how the similarity measure performs.

The methods we tested include, but are not limited to, Random Forests, Neural Networks and Support Vector Regression. Experiments were with both Average and Minimum functions to get the similarity S between axioms, from S_L and S_R (as explained in Section III-B), we will denote by ASF the axiom similarity function here on out. All this means that the number of experiments performed was large so we will only report on some scenarios.

We used RMSE (root mean squared error) as the score, to be consistent with [19] during the comparison. We applied hyper-parameter tuning using grid search as well as five fold cross validation to infer better models. Table I shows the best results (in terms of RMSE) for each experiment using each model.

While replicating the experiment of [19], we decided to test other methods in addition to theirs. We will use the authors' best result for the comparison with the original model. We will be using the original 722 for our proposed method, since our similarity does not require negated axioms (no need for the extra 722 negated axioms).

C. Results and Observations

Comparing ASM creation and candidate axiom encoding time costs in Table I, for both our proposed method and the one in [19], our ontological based similarity seems almost instantaneous. For the instance based similarity it is a problem during dataset preparation as it needs seven and a half days to prepare a data set of 722 axioms (and their negations), as well as candidate axiom encoding/processing where it takes half an hour to process every candidate axiom we want to predict a score to, whereas our ontological based method requires about fourteen seconds to process and prepare the exact same dataset, and less than 0.1 s to encode a new axiom, making it much more preferable with regards to time cost.

During the experiments we were unable to compare both methods in anything other than subClassOf axioms. This is because the method detailed in [19] can only handle subClassOf axioms making it very limited and constrained, whereas our proposed method can address all atomic class axiom types, all that is needed is training one model for each set. Considering the timing needed to prepare the training data (ASM), as seen in Table I, this is easily doable and very efficient.

We observe in Table I that the time cost for creating the disjointWith (129 seconds for 3868 axioms) experiment's ASM was almost ten times that of the subClassOf (13 seconds for 722 axioms), even though the number of axioms is not ten times as much, only about five. This is normal since disjointWith axioms are symmetrical, and as shown in Algorithm 1 and explained in Section III-B, the calculation is doubled for every axiom to check forwards and backwards the similarity between the pair of axioms.

It is very important to note that the instance-based similarity method performed better when used with a machine learning method different from the modified support vector clustering method used in [19]. It was able to achieve less than half the RMSE score of what was stated in [19]. This suggests that the modified support vector clustering method is unnecessarily complicated and not the most appropriate method to deal with this problem, which turns out to be relatively easy once a good similarity measure manages to transform it to a suitable representation. The ontological based method outperformed the instance based method in Neural Network and Random forests models. For the same data set, the subClassOf set, the two methods seem to perform closely in terms of RMSE, while the ontological distance method seems to be achieving very good scores in the larger disjointWith set. This made us look more in-depth to what was happening. Turning our attention to the support vector regression model, we can see in Table I that for the same dataset (subClassOf) the instance based measure performed considerably better than the ontological based method, but was still outperformed in the larger (disjointWith) set as far as performance goes. So we investigated the reason behind this specific performance for this specific dataset in this model, and found the best

explanation in the explained variance score. We found that for the instance based method, according to the explained variance score of **0.56252** for its best performing model the support vector regressor, this performance is specific to this dataset. This means there exists a bias in the training set and it is expected that any new candidate will suffer from a higher error rate than what is experienced in the training stage. This is explained by the fact that both similarity method and dataset were handcrafted and picked to work with a support vector clustering method modified to work as a regressor. On the other hand for the ontological based method the explained variance score throughout all our experiments ranged between 0.71 up to 0.88 meaning this method will produce better results when predicting scores for new candidate axioms.

All results shown in Table I were obtained using the Average ASF, as it outperformed the Minimum ASF in all our runs.

V. Conclusion

We have proposed a method with the aim of learning predictors for the acceptability of an atomic candidate *OWL* axiom of any type. The method relies on a semantic similarity measure derived from the ontological distance between concepts in a subsumption hierarchy. Extensive tests that covered multiple parameters and settings were carried out to investigate the effectiveness and potential of the method compared with the state of the art.

The results obtained strongly support the effectiveness of the proposed method in predicting the scores of the considered OWL axiom types with a consistently low error rate. This allows us to confidently say that our proposed method can be used in combination with ILP or statistical methods, such as Dl-learner [4] or a Grammatical evolution approach such as [21], as a building block or extension/plugin to allow faster execution while maintaining accuracy.

We attribute the high accuracy and extremely good performance of our method to the close relation between the similarity measure and the model-theoretic semantics of axioms. This merits further investigation of links between similarity/distance measures and semantics, and the effect on performance of machine learning models in logical reasoning tasks.

Based on our findings, some research paths emerge, including:

- Developing the method to predict the scores of complex candidate axioms.
- Extending the method to property axioms since they also possess an **IS-A** hierarchy, which is the base of our similarity measure.
- Considering an ensemble approach that incorporates active learning based of the agreement or disagreement of included models based on a threshold such as the variance.

References

[1] Al-Mubaid, H., Nguyen, H.A.: A cluster-based approach for semantic similarity in the biomedical domain. IEEE Engineering in Medicine and Biology, pp. 2713–2717 (2006). https://doi.org/10.1109/IEMBS.2006.259235

[2] Asim, M.N., Wasim, M., Khan, M.U.G., Mahmood, W., Abbasi, H.M.: A survey of ontology learning techniques and applications. Database **2018**(2018), 1–24 (2018). https://doi.org/10.1093/database/bay101

[3] Ballatore, A., Bertolotto, M., Wilson, D.C.: An evaluative baseline for geo-semantic relatedness and similarity. GeoInformatica **18**(4), 747–767 (2014). https://doi.org/10.1007/s10707-013-0197-8

[4] Bühmann, L., Lehmann, J., Westphal, P.: Dl-learner—a framework for inductive learning on the semantic web. Journal of Web Semantics **39**, 15–24 (2016). https://doi.org/https://doi.org/10.1016/j.websem.2016.06.001

[5] Olivier Corby, Rose Dieng-Kuntz, Catherine Faron-Zucker, and Fabien Gandon. 2006. Searching the Semantic Web: Approximate Query Processing Based on Ontologies. IEEE Intelligent Systems 21, 1 (January 2006), 20–27. https://doi.org/https://doi.org/10.1109/MIS.2006.16

[6] Corby, O., Dieng-Kuntz, R., Faron Zucker, C. & Gandon, F. Ontology-based Approximate Query Processing for Searching the Semantic Web with Corese. (INRIA,2006), https://hal.inria.fr/inria-00070387

[7] Corby, O., Dieng-Kuntz, R., Faron-Zucker, C.: Querying the semantic web with corese search engine. ECAI 2004, pp. 705–709.

[8] Corcho, O., Fernández-López, M., Gómez-Pérez, A.: Ontological engineering: Principles, methods, tools and languages. Ontologies for Software Engineering and Software Technology pp. 1–48 (2006). https://doi.org/10.1007/3-540-34518-3_1

[9] Dragisic, Z.: Completion of Ontologies and Ontology Networks. Ph.D. thesis, Linköping University, Faculty of Science & Engineering (2017). https://doi.org/10.3384/diss.diva-139487

[10] Fanizzi, N., d'Amato, C., Esposito, F.: DL-FOIL concept learning in description logics. ILP 2008, pp. 107–121. https://doi.org/10.1007/978-3-540-85928-4_12

[11] Fleischhacker, D., Völker, J.: Inductive learning of disjointness axioms. OTM 2011, pp. 680–697. https://doi.org/10.1007/978-3-642-25106-1_20

[12] Fleischhacker, D., Völker, J., Stuckenschmidt, H.: Mining RDF data for property axioms. OTM 2012, pp. 718–735. https://doi.org/10.1007/978-3-642-33615-7_18

[13] Gruber R., T.: Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal of human-computer studies **43**(5-6), 907–928 (1995)

[14] Hadzic, M., Wongthongtham, P., Dillon, T., Chang, E.: Introduction to ontology learning. Studies in Computational Intelligence **219**, 37–60 (2009). https://doi.org/10.1007/978-3-642-01904-3_3

[15] Lehmann, J.: DL-Learner: Learning concepts in description logics. Journal of Machine Learning Research **10**, 2639–2642 (2009)

[16] Lehmann, J., Völker, J. (eds.): Perspectives on Ontology Learning, Studies on the Semantic Web, vol. 18. IOS Press, Amsterdam (2014). https://doi.org/10.3233/978-1-61499-379-7-i

[17] Lisi, F.A., Straccia, U.: A logic-based computational method for the automated induction of fuzzy ontology axioms. Fundamenta Informaticae **124**(4), 503–519 (2013). https://doi.org/10.3233/FI-2013-846

[18] Maedche, A., Staab, S.: Ontology learning for the semantic web. IEEE Intelligent Systems **16**(2), 72–79 (March 2001)

[19] Malchiodi, D., Tettamanzi, A.G.: Predicting the possibilistic score of OWL axioms through modified support vector clustering. SAC 2018, pp. 1984–1991. https://doi.org/10.1145/3167132.3167345

[20] Muggleton, S., De Raedt, L., Poole, D., Bratko, I., Flach, P., Inoue, K., Srinivasan, A.: ILP turns 20: Biography and future challenges. Machine Learning **86**, 3–23 (2012). https://doi.org/10.1007/s10994-011-5259-2

[21] Nguyen, T.H., Tettamanzi, A.G.B.: Learning Class Disjointness Axioms Using Grammatical Evolution. EuroGP 2019, pp. 278–294. https://doi.org/10.1007/978-3-030-16670-0_18,

[22] Nguyen, T.H., Tettamanzi, A.G.: An evolutionary approach to class disjointness axiom discovery. WI 2019. https://doi.org/10.1145/3350546.3352502

[23] Simperl, E., Bürger, T., Hangl, S., Wörgl, S., Popov, I.: ONTOCOM: A reliable cost estimation method for ontology development projects. Journal of Web Semantics **16**, 1–16 (2012). https://doi.org/10.1016/j.websem.2012.07.001

[24] Tettamanzi, A.G., Faron-zucker, C., Gandon, F.: Testing owl axioms against RDF facts: A possibilistic approach. EKAW 2014. https://doi.org/10.1007/978-3-319-13704-9_39

[25] Tettamanzi, A.G., Faron-Zucker, C., Gandon, F.: Possibilistic testing of OWL axioms against RDF data. International Journal of Approximate Reasoning (2017). https://doi.org/10.1016/j.ijar.2017.08.012

[26] Tettamanzi, A.G., Zucker, C.F., Gandon, F.: Dynamically time-capped possibilistic testing of SubClassOf axioms against RDF data to enrich schemas. K-CAP 2015. https://doi.org/10.1145/2815833.2815835

[27] Dubois, D. & Prade, H. Possibility Theory—An Approach to Computerized Processing of Uncertainty. (Plenum Press,1988)