Dynamically Time-Capped Possibilistic Testing of SubClassOf Axioms Against RDF Data to Enrich Schemas

Andrea G. B. Tettamanzi
Univ. Nice Sophia Antipolis,
I3S, UMR 7271
Sophia Antipolis, France
andrea.tettamanzi@unice.fr

Catherine Faron Zucker
Univ. Nice Sophia Antipolis,
I3S, UMR 7271
Sophia Antipolis, France
faron@unice.fr

Fabien Gandon
INRIA Sophia Antipolis –
Méditerranée
Sophia Antipolis, France
fabien.gandon@inria.fr

ABSTRACT

Axiom scoring is a critical task both for the automatic enrichment/learning and for the automatic validation of knowledge bases and ontologies. We designed and developed an axiom scoring heuristic based on possibility theory, which aims at overcoming some limitations of scoring heuristics based on statistical inference and taking into account the open-world assumption of the linked data on the Web. Since computing the possibilistic score can be computationally quite heavy for some candidate axioms, we propose a method based on time capping to alleviate the computation of the heuristic without giving up the precision of the scores. We evaluate our proposal by applying it to the problem of testing `SubClassOf` axioms against the DBpedia RDF dataset.

Keywords

ontology learning, open-world assumption, possibility theory

1. INTRODUCTION

It is common practice, in the semantic Web, to put a strong emphasis on the construction or reuse of ontologies based on a principled conceptual analysis of a domain of interest, as a prerequisite for the organization of the Linked Open Data (LOD), much like a database schema must be designed before a database can be populated. While this approach is quite successful when applied to specific domains, it does not scale well to more general settings; it is aprioristic and dogmatic; it does not lend itself to a collaborative effort; etc. That is why an alternative, bottom-up, approach to ontology and knowledge base creation better suits many scenarios: instead of postulating an *a priori* conceptualization of reality (i.e., an ontology) and requiring that our knowledge about facts complies with it, one can start from RDF facts and learn OWL 2 axioms.

Recent contributions towards the automatic creation of OWL 2 ontologies from large repositories of RDF facts include FOIL-like algorithms for learning concept definitions [4], statistical schema induction via association rule mining [6], and light-weight schema enrichment methods based on the DL-Learner framework [9, 1]. All these methods apply and extend techniques developed within inductive logic programming (ILP) [12]. For a recent survey of the wider field of ontology learning, see [11].

There exists also a need for evaluating and validating ontologies, be they the result of an analysis effort or of a semi-automatic learning method. This need is witnessed by general methodological investigations [7, 8] and surveys [16] and tools like OOPS! [13] for detecting pitfalls in ontologies. Ontology engineering methodologies, such as METHONTOLOGY [5], distinguish two validation activities, namely verification (through formal methods, syntax, logics, etc.) and validation (through usage). Whilst this latter is usually thought of as user studies, an automatic process of validation based on RDF data would provide a cheap alternative, whereby the existing linked data may be regarded as usage traces that can be used to test and improve the ontologies, much like log mining can be used to provide test cases for development in the replay approaches.

Alternatively, one may regard the ontology as a set of integrity constraints and check if the data satisfy them, using a tool like Pellet integrity constraint validator (ICV), which translates OWL ontologies into SPARQL queries to automatically validate RDF data [15]. A similar approach also underlies the idea of test-driven evaluation of linked data quality [10]. To this end, OWL ontologies are interpreted under the closed-world assumption and the weak unique name assumption.

Yet this validation process may be seen from a reverse point of view: instead of starting from the *a priori* assumption that a given ontology is correct and verifying whether the facts contained in an RDF base satisfy it, one may treat ontologies like hypotheses and develop a methodology to verify whether the RDF facts corroborate or falsify them. Ontology learning and validation are thus strictly related. They could even be seen as an agile and test-driven approach to ontology development, where the linked data is used as a giant test case library not only to validate the schema but even to suggest new developments.

Ontology learning and validation rely critically on (candidate) axiom scoring. In this paper, we will tackle the problem of testing a single, isolated axiom, which is the first step to solve the problem of validating an entire ontology. Furthermore, to focus our evaluation and examples, we will restrict our attention to subsumption axioms of the form `SubClassOf`$(C\ D)$.

K-CAP 2015 October 07–10, 2015, Palisades, NY, USA

DOI: http://dx.doi.org/10.1145/2815833.2815835

The most popular scoring heuristics proposed in the literature are based on statistical inference. We argue that such a probability-based framework is not satisfactory. We propose an axiom scoring heuristic based on a formalization in possibility theory of the notions of logical content of a theory and of falsification, inspired by Karl Popper's approach to epistemology, and working with an open-world assumption.

The first results [17] indicated that applying a possibilistic approach to the task of testing candidate axioms for ontology learning yields very promising results and hints that the same approach could be beneficial to ontology and knowledge base validation as well. However, the proposed heuristic is much heavier, from a computational point of view, than the probabilistic scores it aims to complement. Fortunately, there is evidence (see [17] and Section 5 below) that the time it takes to test an axiom tends to be inversely proportional to its score. This suggests that time-capping the test might be an acceptable additional heuristic to decide whether to accept or reject a candidate axiom, for an axiom which takes too long to test will likely end up having a very negative score. In this paper, we follow this suggestion and investigate the effectiveness of time-capped possibilistic testing of OWL axioms against the facts contained in an RDF repository. Our research question is, therefore: "Can time capping alleviate the computation of the proposed possibilistic axiom scoring heuristic without giving up the precision of the scores?" This paper is organized as follows: Section 2 presents the principles of axiom testing. Section 3 proposes an axiom scoring heuristic based on possibility theory. A framework for axiom scoring based on such heuristic is then presented in Section 4 and evaluated on subsumption axioms in Section 5. Section 6 draws some conclusions and directions for future work.

2. PRINCIPLES OF AXIOM TESTING

Testing an axiom against an RDF dataset can be done by checking whether the formulas entailed by it are confirmed by the facts contained in the RDF dataset.[1]

2.1 OWL 2 Semantics

We refer to the direct model-theoretic semantics of OWL 2 as defined in [2].[2] An interpretation $\mathcal{I}$ for a datatype map D and a vocabulary V over D is defined by an interpretation domain $\Delta^{\mathcal{I}} = \Delta_I \cup \Delta_D$ (Δ_I is the *object domain* and Δ_D the *data domain*), and a valuation function $\cdot^{\mathcal{I}}$ with seven restrictions: $\cdot^{C}$ mapping class expressions to subsets of Δ_I, $\cdot^{OP}$ mapping object properties to subsets of $\Delta_I \times \Delta_I$, $\cdot^{DP}$ mapping data properties to subsets of $\Delta_I \times \Delta_D$, $\cdot^{I}$ mapping individuals to elements of Δ_I, $\cdot^{DT}$ mapping datatypes to subsets of Δ_D, $\cdot^{LT}$ mapping literals to elements of the set of data values $(DT)^{DT}$ of D and $\cdot^{FT}$ mapping facets to subsets of $(DT)^{DT}$.

2.2 Content, Support, Confirmation, and Counterexample of an Axiom

Let ϕ be a candidate axiom; we denote by u_ϕ the support of ϕ, i.e., the cardinality of the set of formulas entailed by ϕ which will be tested against the facts contained in the RDF dataset. We shall define this notion of support with respect to an RDF dataset more precisely.

We define the *content* of an axiom ϕ, $content(\phi)$, as the finite set of formulas, which can be tested against an RDF dataset $\mathcal{K}$, constructed from the set-theoretic formulas expressing the semantics of ϕ by grounding them, i.e., by

- omitting all $\forall$ quantifiers,
- substituting all universally quantified variable symbols x denoting an individual of $\Delta^{\mathcal{I}}$ by every resource r or literal l occurring in $\mathcal{K}$,[3]
- substituting all symbols $C^{\mathcal{I}}$ denoting subsets of $\Delta^{\mathcal{I}}$ by their corresponding class name or datatype name C, and
- substituting all symbols $R^{\mathcal{I}}$ denoting subsets of $\Delta_I \times \Delta_I$ or $\Delta_I \times \Delta_D$ by their corresponding object or data property name R.

For example, let us consider the test of candidate axiom

$$\phi = \texttt{SubClassOf(dbo:LaunchPad dbo:Infrastructure)},$$

or `dbo:LaunchPad` $\sqsubseteq$ `dbo:Infrastructure` in Description Logics (DL) syntax, against the DBpedia dataset. The semantics of ϕ is

$$\texttt{dbo:LaunchPad}^{\mathcal{I}} \subseteq \texttt{dbo:Infrastructure}^{\mathcal{I}},$$

which can also be written as

$$\forall x \in \Delta^{\mathcal{I}}, x \in \texttt{dbo:LaunchPad}^{\mathcal{I}} \Rightarrow x \in \texttt{dbo:Infrastructure}^{\mathcal{I}}.$$

We may thus express $content(\phi)$ as

$$\{\ \texttt{dbo:LaunchPad}(r) \Rightarrow \texttt{dbo:Infrastructure}(r) : r \text{ is a resource occurring in DBPedia}\ \}.$$

By construction, for all $\psi \in content(\phi)$, $\phi \models \psi$. Indeed, let $\mathcal{I}$ be a model of ϕ; by definition, $\mathcal{I}$ is also a model of the formula which expresses the semantics of ϕ and *a fortiori*, also of all its groundings; since ψ is a grounding of the formula which expresses the semantics of ϕ, $\mathcal{I}$ is a model of ψ.

Now, given a formula $\psi \in content(\phi)$ and an RDF dataset $\mathcal{K}$, there are three cases:

1. $\mathcal{K} \models \psi$: in this case, we will call ψ a *confirmation* of ϕ;
2. $\mathcal{K} \models \neg\psi$: in this case, 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.

The definition of $content(\phi)$ may be refined by adopting Scheffler and Goodman's principle of *selective confirmation* [14], which characterizes a confirmation as a fact not simply confirming a candidate axiom, but, further, favoring the axiom rather than its contrary. For instance, the occurence of a black raven *selectively confirms* the axiom `Raven` $\sqsubseteq$ `Black` because it both confirms it and fails to confirm its negation, namely that there exist ravens that are not black. On the contrary, the observation of a green apple

[1] Note that calling linked data search engines like Sindice could virtually extend the dataset to the whole LOD cloud.

[2] `http://www.w3.org/TR/2012/REC-owl2-direct-semantics-20121211/`, Section 2.2 Interpretations

[3] This may be construed as $r^{\mathcal{I}} = x$ or $l^{\mathcal{I}} = x$

does not contradict `Raven` $\sqsubseteq$ `Black`, but it does not disconfirm `Raven` $\not\sqsubseteq$ `Black` either, i.e., it does not selectively confirm `Raven` $\sqsubseteq$ `Black`.

The definition of $content(\phi)$ may be further refined, in order to restrict it just to those ψ which can be counterexamples of ϕ, thus leaving out all those ψ which would be trivial confirmations of ϕ. That is like saying that, to test a hypothesis, we have to try, as hard as we can, to refute it.

For example, in the case of a `SubClassOf`$(C\ D)$ axiom, all ψ involving the existence of a resource r for which $\mathcal{K} \not\models C(r)$ will either be confirmations (if $\mathcal{K} \models D(r)$) or they will fall into Case 3 otherwise. Therefore, such ψ will not be interesting and should be left out of $content($`SubClassOf`$(C\ D))$.

Applying this principle greatly reduces $content(\phi)$ and, therefore, the number of ψ that will have to be checked.

We can now define the support of ϕ as the cardinality of $content(\phi)$:

$$u_\phi = \|content(\phi)\|. \tag{1}$$

Since an RDF dataset is finite, u_ϕ is also finite.

We denote by u_ϕ^+ the number of formulas $\psi \in content(\phi)$ which are entailed by the RDF dataset (confirmations); and by u_ϕ^- the number of such formulas whose negation $\neg\psi$ is entailed by the RDF dataset (counterexamples). Notice that it is possible that, for some $\psi \in content(\phi)$, the RDF dataset entails neither ψ nor $\neg\psi$ (Case 3 above). Therefore,

$$u_\phi^+ + u_\phi^- \leq u_\phi. \tag{2}$$

For example, when testing

$$\phi = \texttt{dbo:LaunchPad} \sqsubseteq \texttt{dbo:Infrastructure}$$

against the DBpedia dataset, we found that $u_\phi = 85$, $u_\phi^+ = 83$, i.e., there are 83 confirmations of ϕ in the dataset; and $u_\phi^- = 1$, i.e., there is 1 couterexample in the dataset, namely

$$\texttt{dbo:LaunchPad(:USA)} \Rightarrow \texttt{dbo:Infrastructure(:USA)},$$

since

$$\begin{aligned} \text{DBpedia} &\models \texttt{dbo:LaunchPad(:USA)}, \\ \text{DBpedia} &\models \neg\texttt{dbo:Infrastructure(:USA)}. \end{aligned}$$

and one formula in $content(\phi)$ is neither a confirmation nor a counterexample, namely

$$\texttt{dbo:LaunchPad(:Cape\_Canaveral)} \Rightarrow \texttt{dbo:Infrastructure(:Cape\_Canaveral)},$$

because

$$\begin{aligned} \text{DBpedia} &\models \texttt{dbo:LaunchPad(:Cape\_Canaveral)}, \\ \text{DBpedia} &\not\models \texttt{dbo:Infrastructure(:Cape\_Canaveral)}, \\ \text{DBpedia} &\not\models \neg\texttt{dbo:Infrastructure(:Cape\_Canaveral)}. \end{aligned}$$

Further interesting properties of u_ϕ, u_ϕ^+, and u_ϕ^- are the following:

1. $u_\phi^+ = u_{\neg\phi}^-$ (the confirmations of ϕ are counterexamples of $\neg\phi$);
2. $u_\phi^- = u_{\neg\phi}^+$ (the counterexamples of ϕ are confirmations of $\neg\phi$);
3. $u_\phi = u_{\neg\phi}$ (ϕ and $\neg\phi$ have the same support).

3. A POSSIBILISTIC CANDIDATE AXIOM SCORING

We present an axiom scoring heuristic which captures the basic intuition behind the process of axiom discovery based on possibility theory: assigning to a candidate axiom a degree of possibility equal to 1 just means that this axiom is possible, plausible, i.e. is not contradicted by facts in the knowledge base. This is much weaker than assigning a probability equal to 1, meaning that the candidate axiom certainly *is* an axiom.

3.1 Possibility Theory

Possibility theory [18] is a mathematical theory of epistemic uncertainty. Given a finite universe of discourse Ω, whose elements $\omega \in \Omega$ may be regarded as events, values of a variable, possible worlds, or states of affairs, a possibility distribution is a mapping $\pi : \Omega \to [0, 1]$, which assigns to each ω a degree of possibility ranging from 0 (impossible, excluded) to 1 (completely possible, normal). A possibility distribution π for which there exists a completely possible state of affairs ($\exists\omega \in \Omega : \pi(\omega) = 1$) is said to be *normalized*.

There is a similarity between possibility distribution and probability density. However, it must be stressed that $\pi(\omega) = 1$ just means that ω is a plausible (normal) situation and therefore should not be excluded. A degree of possibility can then be viewed as an upper bound of a degree of probability. See [3] for a discussion about the relationships between fuzzy sets, possibility, and probability degrees. Possibility theory is suitable to represent incomplete knowledge while probability is adapted to represent random and observed phenomena.

A possibility distribution π induces a *possibility measure* and its dual *necessity measure*, denoted by Π and N respectively. Both measures apply to a set $A \subseteq \Omega$ (or to a formula ϕ, by way of the set of its models, $A = \{\omega : \omega \models \phi\}$), and are defined as follows:

$$\Pi(A) = \max_{\omega \in A} \pi(\omega); \tag{3}$$

$$N(A) = 1 - \Pi(\bar{A}) = \min_{\omega \in \bar{A}} \{1 - \pi(\omega)\}. \tag{4}$$

Here are a few properties of possibility and necessity measures induced by a normalized possibility distribution on a finite universe of discourse Ω:

1. $\Pi(\emptyset) = N(\emptyset) = 0, \quad \Pi(\Omega) = N(\Omega) = 1$;
2. $\forall A \subseteq \Omega$, $\Pi(A) = 1 - N(\bar{A})$ (duality);
3. $\forall A \subseteq \Omega$, $N(A) > 0$ implies $\Pi(A) = 1$, and $\Pi(A) < 1$ implies $N(A) = 0$.

In case of complete ignorance on A, $\Pi(A) = \Pi(\bar{A}) = 1$.

3.2 Possibility and Necessity of an Axiom

The basic principle for establishing the possibility of a formula ϕ should be that the absence of counterexamples to ϕ in the RDF repository means $\Pi(\phi) = 1$, i.e., that ϕ is completely possible.

A hypothesis should be regarded as all the more *necessary* as it is explicitly supported by facts and not contradicted by any fact; and all the more *possible* as it is not contradicted by facts. In other words, given hypothesis ϕ, $\Pi(\phi) = 1$ if no counterexamples are found; as the number of counterexamples increases, $\Pi(\phi) \to 0$ strictly monotonically; $N(\phi) = 0$

if no confirmations are found; as the number of confirmations increases and no counterexamples are found, $N(\phi) \to 1$ strictly monotonically. Notice that a confirmation of ϕ is a counterexample of $\neg\phi$ and that a counterexample of ϕ is a confirmation of $\neg\phi$.

A definition of Π and N which captures the above intuitions, but by no means the only possible one, is, for $u_\phi > 0$,

$$\Pi(\phi) = 1 - \sqrt{1 - \left(\frac{u_\phi - u_\phi^-}{u_\phi}\right)^2}; \quad (5)$$

$$N(\phi) = \begin{cases} \sqrt{1 - \left(\frac{u_\phi - u_\phi^+}{u_\phi}\right)^2}, & \text{if } u_\phi^- = 0, \\ 0, & \text{if } u_\phi^- > 0. \end{cases} \quad (6)$$

3.3 Axiom Scoring

We combine the possibility and necessity of an axiom to define a single handy acceptance/rejection index (ARI) as follows:

$$\mathrm{ARI}(\phi) = N(\phi) - N(\neg\phi) = N(\phi) + \Pi(\phi) - 1 \in [-1, 1]. \quad (7)$$

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.

Although this ARI is useful for the purpose of analyzing the results of our experiments and to visualize the distribution of the tested axiom with respect to a single axis, one should always bear in mind that an axiom is scored by the proposed heuristic in terms of two bipolar figures of merit, whose meanings, though related, are very different:

- $\Pi(\phi)$ expresses the degree to which ϕ may be considered "normal", in the sense of "not exceptional, not surprising", or not contradicted by actual observations;
- $N(\phi)$, on the other hand, expresses the degree to which ϕ is certain, granted by positive evidence and corroborated by actual observations.

4. A FRAMEWORK FOR CANDIDATE AXIOM TESTING

A general algorithm for testing all the possible OWL 2 axioms in a given RDF store is beyond the scope of this paper. Here, we will restrict our attention to `Class` and `ObjectComplementOf` class expressions and to `SubClassOf` axioms. Scoring these axioms with their ARI requires to compute the interpretation of `Class` and `ObjectComplementOf` class expressions.

4.1 Computational Definition of `Class` and `ObjectComplementOf` Class Expressions

We define a mapping $Q(E, \texttt{?x})$ from OWL 2 class expressions to SPARQL graph patterns, where E is an OWL 2 class expression, and `?x` is a variable, such that the query `SELECT DISTINCT ?x WHERE { `$Q(E, \texttt{?x})$` }` returns all the individuals which are instances of E. We denote this set by $[Q(E, \texttt{?x})]$:

$$[Q(E, \texttt{?x})] = \{v : (?x, v) \in \mathit{ResultSet}(\texttt{SELECT DISTINCT ?x WHERE}\{Q(E, \texttt{?x})\}\}. \quad (8)$$

For a `Class` class expression A (i.e., an atomic concept in DL),

$$Q(A, \texttt{?x}) = \{\texttt{?x a } A\}, \quad (9)$$

where A is a valid IRI.

For an `ObjectComplementOf` class expression, things are slightly more complicated, since RDF does not support negation. The model-theoretic semantics of OWL class expressions of the form `ObjectComplementOf(`C`)` ($\neg C$ in DL syntax), where C denotes a class, is $\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$. However, to learn axioms from an RDF dataset, the open-world hypothesis must be made: the absence of supporting evidence does not necessarily contradict an axiom, moreover an axiom might hold even in the face of a few counterexamples. Therefore, as proposed in [17], we define $Q(\neg C, \texttt{?x})$ as follows, to approximate an open-world semantics:

$$Q(\neg C, \texttt{?x}) = \{\ \texttt{?x a ?dc .} \ \texttt{FILTER NOT EXISTS} \{\ \texttt{?z a ?dc .}\ Q(C, \texttt{?z})\ \}\ \}, \quad (10)$$

where `?z` is a variable that does not occur anywhere else in the query.

For an atomic class expression A, this becomes

$$Q(\neg A, \texttt{?x}) = \{\ \texttt{?x a ?dc .} \ \texttt{FILTER NOT EXISTS} \{\ \texttt{?z a ?dc .}\ \texttt{?z a } A\ \}\ \}. \quad (11)$$

4.2 Computational Definitions of the Support and the ARI of `SubClassOf` Axioms

The semantics of `SubClassOf` axioms of the form $C \sqsubseteq D$ is $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, which may also be written $x \in C^{\mathcal{I}} \Rightarrow x \in D^{\mathcal{I}}$. Therefore, according to Equation 1 and following the principle of selective confirmation,

$$u_{C \sqsubseteq D} = \|\{D(a) : \mathcal{K} \models C(a)\}\|, \quad (12)$$

because, if $C(a)$ holds, then $C(a) \Rightarrow D(a) \equiv \neg C(a) \vee D(a) \equiv \bot \vee D(a) \equiv D(a)$.
As a result, a computational definition of $u_{C \sqsubseteq D}$ is the following SPARQL query:

```
SELECT (count(DISTINCT ?x) AS ?u)
WHERE {Q(C, ?x)}.
```
(13)

In order to compute the score of `SubClassOf` axioms, $ARI(C \sqsubseteq D)$, we must provide a computational definition of $u^+_{C \sqsubseteq D}$ and $u^-_{C \sqsubseteq D}$. We start with the following statements:

- confirmations are individuals i such that $i \in [Q(C, \texttt{?x})]$ and $i \in [Q(D, \texttt{?x})]$;
- counterexamples are individuals i such that $i \in [Q(C, \texttt{?x})]$ and $i \in [Q(\neg D, \texttt{?x})]$.

This may be translated into the following two SPARQL queries to compute $u^+_{C \sqsubseteq D}$ and $u^-_{C \sqsubseteq D}$ respectively:

```
SELECT (count(DISTINCT ?x) AS ?nConfirm)
WHERE { Q(C, ?x) Q(D, ?x) }
```
(14)

and

```
SELECT (count(DISTINCT ?x) AS ?nCounter)
WHERE { Q(C, ?x) Q(¬D, ?x) }.
```
(15)

Notice that an i such that $i \in [Q(C, \texttt{?x})]$ and $i \notin [Q(D, \texttt{?x})]$ does not contradict $C \sqsubseteq D$, because it might well be the

case that the assertion $D(i)$ is just missing. Likewise, an $i \in [Q(\neg D, \texttt{?x})]$ such that $i \in [Q(\neg C, \texttt{?x})]$ will not be treated as a confirmation, based on our choice to regard as evidence in favor of a hypothesis only selective confirmations.

Algorithm 1 is a plain implementation of the resulting scoring scheme.

Algorithm 1 Test a `SubClassOf` axiom (plain version, without time cap).

Input: ϕ, an axiom of the form `SubClassOf`$(C\ D)$;
Output: $\Pi(\phi)$, $N(\phi)$, a list of confirmations, and a list of counterexamples.

1: Compute u_ϕ using the query in Equation 13;
2: compute u_ϕ^+ using the query in Equation 14;
3: **if** $0 < u_\phi^+ \leq 100$ **then**
4: query a list of confirmations;
5: **if** $u_\phi^+ < u_\phi$ **then**
6: compute u_ϕ^- using the query in Equation 15;
7: **if** $0 < u_\phi^- \leq 100$ **then**
8: query a list of counterexamples;
9: **else**
10: $u_\phi^- \leftarrow 0$;
11: compute $\Pi(\phi)$ and $N(\phi)$ using Equations 5 and 6.

4.3 Scalable Axiom Scoring based on Time Prediction

As already discussed in [17], the proposed acceptance-rejection index is more accurate than the probabilistic score. For example, Figure 1 shows a comparison of the ARI to the probabilistic score proposed in [1] for 722 `SubClassOf` axioms involving atomic classes when tested against DBpedia, which are an extension of the results presented in [17]. The same work suggested to use an acceptance threshold $\text{ARI}(\phi) > 1/3$ to decide whether to accept an axiom based on the possibilistic score, while [1] proposed accepting axioms whose probabilistic score is greater than 0.7. We may notice that both criteria roughly partition the score range into three and consider as valid axioms those whose score lies in the upper third of the range.

The improvement in accuracy of the possibilistic score, however, comes at a very high cost. Testing a single axiom against an RDF dataset of the size of DBpedia can take as little as a few milliseconds in some cases, but in many cases it can even take days on a powerful workstation. The circles representing the axioms in Figure 1 are colored using a "terrain" palette based on the test time: green represents the short times, rosish brown the longest times. The color scale is logarithmic. A way to speed up testing is badly needed, lest the method we propose be practically useless. Fortunately, a closer analysis of the time taken to test different axioms pinpoints some regularities. We will base this analysis on the above-mentioned results. Altogether, a staggering 25,037,053,021 ms (a little less than 290 days) of CPU time have had to be spent to gather them.

According to Algorithm 1, testing `SubClassOf` axioms requires executing at least three SPARQL queries. Of these, the query in Equation 15, to count the number of counterexamples, is where most CPU time is spent, order of magnitudes more than in the other queries taken as a whole. The reason of this resides in the use of the `FILTER NOT EXISTS` clause in the $Q(\neg D, \texttt{?x})$ graph pattern (see Equation 10).

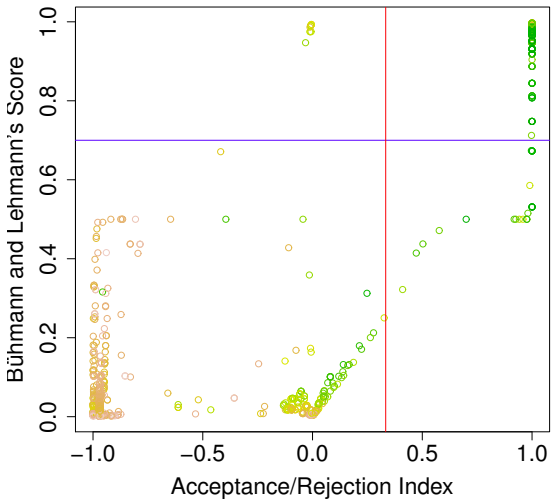

Figure 1: A comparison of the acceptance/rejection index and the probability-based score used in [1] on axioms tested without time capping. The vertical red line shows the acceptance threshold $\text{ARI}(\phi) > 1/3$; the horizontal blue line the acceptance threshold of 0.7 for the probabilistic score.

This approximation of an open-world semantics is a critical ingredient of our scoring heuristic and cannot be dispensed with.

Now, if we denote $T(\phi)$ the time it takes to test an axiom ϕ and we plot $T(\phi)$ as a function of its score, $\text{ARI}(\phi)$ (see Figure 2), a rather striking pattern emerges: all axioms ϕ having a positive ARI (thus $\Pi(\phi) = 1$ and $N(\phi) > 0$) have a very small $T(\phi)$. $N(\phi) > 0$ means no counterexample could be found ($u_\phi^- = 0$) and some confirmations have been found ($u_\phi^+ > 0$). For axioms having an ARI around zero or negative, $T(\phi)$ may vary widely: from Figure 2, it appears that the range of this variation increases the more $ARI(\phi)$ approaches -1. The relation between $T(\phi)$ and $\text{ARI}(\phi)$ is probably more complex, but for our purposes we may describe it by saying that $T(\phi) = O\left((1 + \text{ARI}(\phi))^{-1}\right)$ or, perhaps, $T(\phi) = O\left(\exp(-\text{ARI}(\phi))\right)$.

Be that as it may, an axiom which takes too long to test will likely end up being rejected. This naturally suggests that a strategy to speed up the test might be to cap the time allowed to execute the query on Line 6 of Algorithm 1 and to reject an axiom if the test runs out of time.

The question now is: how much time should we allow in order to be reasonably sure we are not throwing the baby out with the bathwater, while avoiding to waste time on hopeless tests? Knowing the relation between the score of a candidate axiom ϕ and $T(\phi)$ is of little help, since we do not have $\text{ARI}(\phi)$ before testing ϕ. However, studying the elapsed times for candidate axioms ϕ that ended up being accepted (such that $\text{ARI}(\phi) > 1/3$) we observed that the time it takes to test a `SubClassOf` axiom of the form $C \sqsubseteq D$ tends to be proportional to the product of its support $u_{C \sqsubseteq D}$ and the number of classes that have at least a known instance in

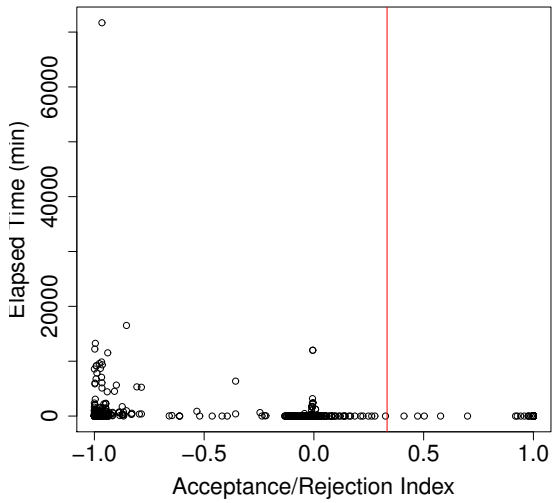

Figure 2: Plot of the time taken for testing the systematically generated `SubClassOf` axioms without time capping as a function of ARI. The vertical red line shows the acceptance threshold $\mathrm{ARI}(\phi) > 1/3$**.**

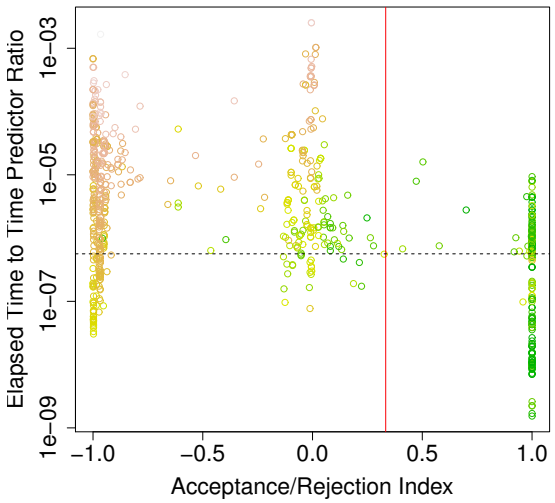

Figure 3: Ratio of the time taken to test the axioms to the time predictor as a function of their acceptance-rejection index without time capping. The vertical red line shows the acceptance threshold $\mathrm{ARI}(\phi) > 1/3$**; the horizontal dashed line corresponds to the slope (**b **coefficient) of the linear regression of the time predictor of accepted axioms on their test time.**

common with C. Since such product may be used to *predict* the time testing a valid `SubClassOf` axiom will take, we will call it its *time predictor*:

$$\mathrm{TP}(C \sqsubseteq D) = u_{C \sqsubseteq D} \cdot nic_C, \tag{16}$$

where nic_C denotes the number of intersecting classes of C. A computational definition of nic_C is the following SPARQL query:

```
SELECT (count(DISTINCT ?A) AS ?nic)
WHERE { Q(C,?x)  ?x a ?A . }
```
(17)

where A represents an atomic class expression.

Figure 3 is a plot of the ratio of the elapsed time for testing axioms to their time predictor. This diagram clearly shows that this ratio is very small for accepted axioms (on the right of the acceptance threshold, shown as a red vertical line in the figure), while it may soar to large values for ARIs around 0 and -1.

The time predictor only depends on the subclass (i.e., the left-hand side) of a `SubClassOf` axiom. If we compute the time predictor for all the classes of DBpedia and we sort them by increasing value of their time predictor, we get the diagram shown in Figure 4. The y-axis of the diagram is in logarithmic scale and we can observe that the value of the time predictor increases by more than seven orders of magnitude as we go from the least to the most time-consuming axioms. This suggests that, in order to maximize the number of axioms tested, one might begin by the less time-consuming axioms, whose number is large, and leave the more time-consuming axioms, whose number is small, to the end.

As a result, we define two heuristics to scale axiom scoring:

- We dynamically time-cap the SPARQL queries to compute the ARI of a candidate axiom by a time out defined as

$$t_{\max}(\phi) = a + b \cdot \mathrm{TP}(\phi), \tag{18}$$

  where b is the slope of the regression line of the maximum times observed for each value of the time predictor and the intercept a is large enough to include, below the time out, all the accepted axioms in the set of the axioms tested without time cap; we then interrupt the test of any ϕ that takes longer than $t_{\max}(\phi)$ and reject it, ϕ being highly likely to get a negative ARI and be rejected anyway. This heuristic yields a new version of the axiom testing procedure, which is summarized by Algorithm 2.

- We construct candidate axioms of the form $C \sqsubseteq D$, by considering the subclasses C in increasing order of time predictor. This enables us to maximize the number of tested and accepted axioms in a given time period.

5. EVALUATION

5.1 Experimental Protocol

We evaluated the proposed dynamic time-capping heuristics summarized in Algorithm 2 by performing tests of subsumption axioms using DBpedia 3.9 in English as the reference RDF fact repository. In particular, to obtain comparability with [17], we used a local dump of DBpedia English version 3.9, along with the DBpedia ontology, version 3.9. This local dump of DBpedia, consisting of 812,546,748 RDF triples, has been bulk-loaded into Jena TDB and a prototype for performing axiom tests using the proposed method

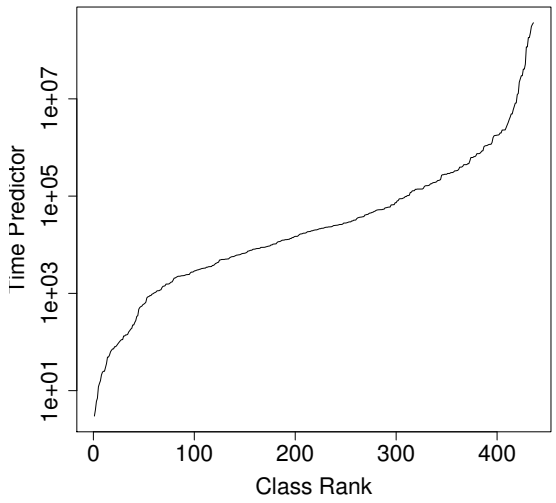

Figure 4: Plot of the time predictor for all the classes of DBpedia as a function of their rank when sorted by increasing value of the time predictor.

has been coded in Java, using Jena ARQ and TDB to access the RDF repository.

We systematically generated and tested subsumption axioms involving atomic classes only, according the following protocol: for each of the 442 classes C referred to in the RDF repository, we constructed all axioms of the form $C \sqsubseteq D$ such that C and D share at least one instance. and we tested these axioms in increasing time-predictor order. To determine the dynamic time cap according to Equation 18, we used $a = 2$ and $b = 5.63682 \cdot 10^{-7}$, on the basis of the results previously collected using the plain implementation of the scoring scheme without time cap summarized in Algorithm 1.

All experiments have been performed on a Fujitsu CELSIUS workstation equipped with twelve six-core Intel Xeon CPU E5-2630 v2 processors at 2.60GHz clock speed, with 15,360 KB cache each, 128 GB RAM, 4 TB of disk space with a 128 GB SSD cache, under the Ubuntu 12.04.4 LTS 64-bit operating system. This is the same machine used for [17]; in both cases the running times reported refer to (user + system) CPU time given by the `ru_utime.tv_sec` field of the structure returned by the Linux system call `getrusage(RUSAGE_SELF, ...)`; as a consequence, the times are commensurable.

5.2 Results

Thanks to the greatly reduced overhead of the dynamic time-capping heuristics, we managed to test 5050 axioms in 1,232,801,812 ms (a little less than 342 hours and a half, 244 s per axiom on average) at the time of writing (the experiment is still running). To have an idea of the time saved, let us consider that testing that many axioms would have taken 175,120,661,712 ms ($\approx 2,027$ days) without time cap. We estimated this duration based on the experiment described in [17] of scoring 722 axioms without time cap. This gives a 142-fold reduction in computing time. Figure 5 shows a summary of the results with a comparison to the probabilistic score.

There are 632 axioms that were tested both with and without time capping; the outcome of the test is different on just 25 of them. That represents an error rate of 3.96%. If we take into account the dramatic improvement in terms of speed, this looks like a very reasonable price to pay in terms of accuracy degradation. In addition, it should be observed that, by construction, the errors are all in the same direction, i.e., some axioms which should be accepted are in fact rejected: at least, this is a conservative heuristic, since it does not generate false positives. Moreover, these axioms supposedly rejected by error repeatedly involve a set of classes whose semantics is not clear, e.g. `gml:_Feature`. These are further discussed in Section 6.

To validate the results of our scoring in absolute term, we took all `SubClassOf` axioms in the DBpedia ontology and added to them all `SubClassOf` axioms that can be inferred from them, thus obtaining a "gold standard" of axioms that *should* be all considered as valid. Of the 5050 tested axioms, 1915 occur in the gold standard; of these, 327 get an ARI below 1/3, which would yield an error rate of about 17%. In fact, in most cases, the ARI of these axioms is around zero, which means that our heuristic gives a suspended judgment. Only 34 axioms have an ARI below $-1/3$. If we took these latter as the real errors, the error rate would fall to just 1.78%. In most of these cases, the superclass is `dbo:PopulatedPlace` or `dbo:Settlement`; however, it is not obvious if one can draw any conclusion from these results.

Algorithm 2 Test a `SubClassOf` axiom (time-capped version).

Input: ϕ, an axiom of the form `SubClassOf`$(C\ D)$;
a, b, the coefficients of the linear time cap equation.
Output: $\Pi(\phi)$, $N(\phi)$, a list of confirmations, and a list of counterexamples.

Compute u_ϕ using the query in Equation 13;
Compute nic using the query in Equation 17;
$\mathrm{TP}(\phi) \leftarrow u_\phi \cdot nic$;
compute u_ϕ^+ using the query in Equation 14;
if $0 < u_\phi^+ \leq 100$ **then**
  query a list of confirmations;
if $u_\phi^+ < u_\phi$ **then**
  $t_{\max}(\phi) \leftarrow a + b \cdot \mathrm{TP}(\phi)$
  waiting up to $t_{\max}(\phi)$ min **do**
    compute u_ϕ^- using the query in Equation 15;
  if time-out then
    $u_\phi^- \leftarrow u_\phi - u_\phi^+$;
  else if $0 < u_\phi^- \leq 100$ **then**
    query a list of counterexamples;
else
  $u_\phi^- \leftarrow 0$;
compute $\Pi(\phi)$ and $N(\phi)$ using Equations 5 and 6.

6. CONCLUSION

We have presented a possibilistic axiom scoring heuristic which is a viable alternative to statistics-based heuristics. We have tested it by applying it to the problem of testing `SubClassOf` axioms against the DBpedia database. We have also proposed an additional heuristic to greatly reduce its computational overhead, consisting of setting a dynamic

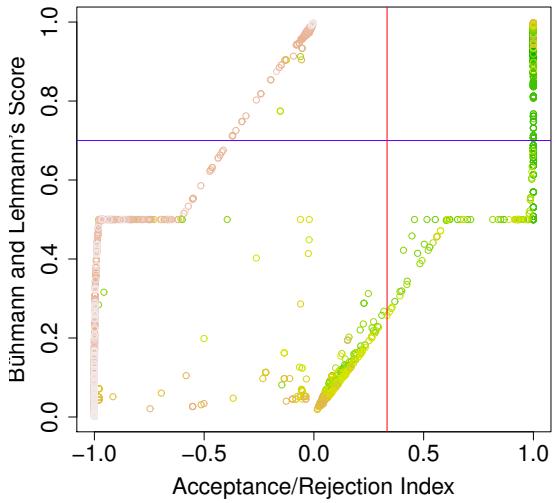

Figure 5: A comparison of the acceptance/rejection index and the probability-based score used in [1] on axioms tested with time capping. The vertical red line shows the acceptance threshold $\mathrm{ARI}(\phi) > 1/3$**; the horizontal blue line the acceptance threshold of 0.7 for the probabilistic score.**

time-out on the test of each axiom.

Our results strongly support the validity of our hypothesis that it is possible to alleviate the computation of the ARI without loosing too much in terms of accuracy.

In addition, a human evaluation of the axioms scored by the system shows that most of the axioms accepted by mistake are inverted `subClassOf` relations between concepts (e.g. `dbo:Case` $\sqsubseteq$ `dbo:LegalCase` instead of `dbo:LegalCase` $\sqsubseteq$ `dbo:Case`). This occurs when counterexamples are missing (all instances of a class are instances of the other class too and the two axioms are positively scored). Other mistakes are on axioms involving vague concepts (e.g., it seems that anything that can appear on a map could be typed with `gml:_Feature` and therefore many classes should be subclasses of it, but it is not clear wether this is correct or not) or used in a more general sense thant it could be expected (e.g., `dbo:PokerPlayer` $\sqsubseteq$ `dbo:Athlete`; this is not really a mistake in the sense that there are several other such concepts involving `dbo:Athlete`). Another example of mistakes is the use of a concept in at least two senses, e.g. `dbo:Library` designating both a building and an institution. Other frequent mistakes are on axioms involving a concept both used as a zoological class name, a taxon, and therefore marked as subclass of `dbp:Species`, and as a set of animals, and therefore subclass of `dbo:Animal` and `dbo:Eukaryote`. The same confusion between the instance level and the ontological level explains the results on axioms involving skos:Concept.

These considerations confirm the interest of using axiom scoring heuristics like ours not only to learn axioms from the LOD, but also to drive the validation and debugging of ontologies and RDF datasets.

7. REFERENCES

[1] L. Bühmann and J. Lehmann. Universal OWL axiom enrichment for large knowledge bases. EKAW 2012, pages 57–71. Springer, 2012.

[2] B. Cuenca Grau, B. Motik, and P. Patel-Schneider. OWL 2 web ontology language direct semantics (second edition). W3C recommendation, W3C, December 2012.

[3] D. Dubois and H. Prade. Fuzzy sets and probability: Misunderstandings, bridges and gaps. *Fuzzy Sets and Systems*, 40(1):143–202, 1991.

[4] N. Fanizzi, C. d'Amato, and F. Esposito. DL-FOIL concept learning in description logics. ILP 2008, pages 107–121. Springer, 2008.

[5] M. Fernández, A. Gómez-Pérez, and N. Juristo. METHONTOLOGY: From ontological art towards ontological engineering. Technical Report SS-97-06, AAAI, 1997.

[6] D. Fleischhacker, J. Völker, and H. Stuckenschmidt. Mining RDF data for property axioms. OTM 2012, pages 718–735, Springer, 2012.

[7] A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann. A theoretical framework for ontology evaluation and validation. SWAP 2005. CEUR-WS.org, 2005.

[8] A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann. Modelling ontology evaluation and validation. ESWC 2006, pages 140–154. Springer, 2006.

[9] S. Hellmann, J. Lehmann, and S. Auer. Learning of OWL class descriptions on very large knowledge bases. *Int. J. Semantic Web Inf. Syst.*, 5(2):25–48, 2009.

[10] D. Kontokostas, P. Westphal, S. Auer, S. Hellmann, J. Lehmann, R. Cornelissen, and A. Zaveri. Test-driven evaluation of linked data quality. WWW 2014.

[11] J. Lehmann and J. Völker, editors. *Perspectives on Ontology Learning*, volume 18 of *Studies on the Semantic Web*. IOS Press, Amsterdam, 2014.

[12] S. Muggleton, L. De Raedt, D. Poole, I. Bratko, P. Flach, K. Inoue, and A. Srinivasan. ILP turns 20: Biography and future challenges. *Machine Learning*, 86:3–23, 2012.

[13] M. Poveda-Villalón, M. del Carmen Suárez-Figueroa, and A. Gómez-Pérez. Validating ontologies with OOPS! EKAW 2012, pages 267–281. Springer, 2012.

[14] I. Scheffler and N. Goodman. Selective confirmation and the ravens: A reply to Foster. *The Journal of Philosophy*, 69(3):78–83, Feb. 10 1972.

[15] E. Sirin and J. Tao. Towards integrity constraints in OWL. OWLED 2009. CEUR-WS.org, 2009.

[16] S. Tartir, I. Budak Arpinar, and A. P. Sheth. Ontological evaluation and validation. In R. Poli, M. Healy, and A. Kameas, editors, *Theory and Applications of Ontologies: Computer Applications*. Springer, 2010.

[17] A. G. B. Tettamanzi, C. Faron-Zucker, and F. L. Gandon. Testing OWL axioms against RDF facts: A possibilistic approach. EKAW 2014, pages 519–530. Springer, 2014.

[18] L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.