Predicting the Empirical Robustness of the Ontology Reasoners based on Machine Learning Techniques

Nourhène Alaya[1,2], Sadok Ben Yahia[2] and Myriam Lamolle[1]
[1]*LIASD, IUT of Montreuil, University of Paris 8, Montreuil, France*
[2]*LIPAH, Faculty of Sciences of Tunis, University of Tunis, Tunis, Tunisia*

Keywords: Ontology, OWL, Reasoner, Robustness, Supervised Machine Learning, Prediction.

Abstract: Reasoning with ontologies is one of the core tasks of research in Description Logics. A variety of reasoners with highly optimized algorithms have been developed to allow inference tasks on expressive ontology languages such as OWL (DL). However, unexpected behaviours of reasoner engines is often observed in practice. Both reasoner time efficiency and result correctness would vary across input ontologies, which is hardly predictable even for experienced reasoner designers. Seeking for better understanding of reasoner empirical behaviours, we propose to use supervised machine learning techniques to automatically predict reasoner *robustness* from its previous running. For this purpose, we introduced a set of comprehensive ontology features. We conducted huge body of experiments for 6 well known reasoners and using over 1000 ontologies from the *ORE'2014* corpus. Our learning results show that we could build highly accuracy reasoner robustness predictive models. Moreover, by interpreting these models, it would be possible to gain insights about particular ontology features likely to be reasoner robustness degrading factors.

1 INTRODUCTION

The key component for working with OWL ontologies is the *Reasoner*. This is because knowledge in an ontology might not be explicit, then a reasoner is required to deduce its implicit knowledge. However, the high expressivity of OWL has increased the computational complexity of inference tasks. For instance, it has been shown that the complexity of the consistency checking of $\mathcal{SROIQ}$ ontologies, the description logic (DL) underlying OWL 2, is of worst-case 2NExpTime-complete (Horrocks et al., 2006). Therefore, a number of highly-optimized reasoners have been developed, which support reasoning about expressive ontologies (Sirin et al., 2007; Glimm et al., 2012; Steigmiller et al., 2014).

Despite the remarkable progress in optimizing reasoning algorithms, unpredictable behaviours of reasoner engines is often observed in practice, particularly when dealing with real world ontologies. Two main aspects would depict this phenomena. On the one hand, the respective authors of (Weithöner et al., 2007; Gonçalves et al., 2012) have outlined the variability of reasoner's time efficiency across OWL ontologies. Roughly speaking, the reasoner optimization tricks, set up by designers to overcome particular DL complexity sources, would lead to enormous scatter in computational runtime across the ontologies, which is still hardly predictable a priori. These findings have motivated various attempts of reasoner runtime prediction using machine learning techniques (Kang et al., 2012; Sazonau et al., 2014; Kang et al., 2014). On the other hand, results reported from the latest Ontology Reasoner Evaluation Workshops, ORE (Gonçalves et al., 2013; Bail et al., 2014) have revealed another aspect of reasoner behaviours variability, namely the *correctness* of the reasoning results. In fact, the evaluations were surprising as reasoners would derive different inferences for the same input ontology. Therefore, an empirical correctness checking method was established to examine reasoner results. Actually in both ORE 2013 and 2014 competitions, there were no single reasoner, which correctly processed and outperformed on all given inputs. Even the fastest reasoner have failed to derive accurate results for some ontologies, while others less performing engines, have succeeded to correctly process them. Thus, we would admit that the most desired qualities, i.e. result correctness and time efficiency, are not empirically guaranteed by all the reasoners and for every ontology. These observations pinpoints the hardness of understanding reasoner empirical behaviours even for experienced and skilled reasoner designers. Thus, it would be worthwhile to be able to automatically predict both correctness and efficiency of reasoners against given ontologies. More invaluable would be gaining insights about which particular aspects in the ontology are lowering these qualities. Obviously, learning such aspects would further improve reasoner optimizations and enhance ontology design by avoiding reasoner performances degrading factors.

In this paper, we introduce a new approach aiming to predict an ontology classification quality that a particular reasoner would be able to achieve by a specific cutoff time. To the best of our knowledge, no prior work tackled such an issue. We designed by *robustness*, the required reasoner quality. We rely on supervised machine learning techniques in order to learn models of reasoner robustness from their previous running. To achieve our purpose, we proposed a set of valuable ontology features, likely to be good indicators of ontologies' hardness level against reasoning tasks. Then, we carried out a huge body of experiments using the widely recognized *ORE'2014* Framework (Bail et al., 2014). Over 1000 ontologies were processed by 6 well known reasoners. Given these evaluation data, reasoner predictive models were trained by 5 of the most effective supervised learning algorithms. We have further improved the accuracy of our models by employing a set of feature selection techniques. Worth of cite, no prior work made use of the *discretization* to identify ontology relevant features. Then, we discussed and employed a variety of prediction assessment measures, well suited in the case of *imbalanced* datasets. Thanks to our established study, we unveiled a set of *local* and *global* ontology key features, likely to alter the reasoner robustness. In overall, our trained reasoner robustness predictive models have shown to be highly accurate, which witness the worthiness of our learning process.

The rest of the paper is organized as follows. Section 2 briefly recalls basic notions that will be of use throughout the paper. Section 3 scrutinizes the related work approaches. Our reasoner robustness learning process as well as the achieved results are, respectively, detailed in Sections 4 - 9. Concluding remarks as well as our future works are given in Section 10.

2 BACKGROUND

In this paper, we focus on OWL 2 (OWL Working Group, 2009) ontologies. Recommended by the W3C, OWL 2 is based on the highly expressive Description Logics $\mathcal{SROIQ}$ (Horrocks et al., 2006). This logical provides a precisely defined meaning for statements made in OWL and thus, makes it possible to automatically reason over OWL ontologies. Among the reasoning tasks, *classification* is considered as the key one. It computes the full concept and role hierarchies in the ontology (Baader et al., 2003). Explicit and implicit subsumption will be derived to help users navigating through the ontology towards mainly explanation and/or query answering respective tasks. Thus, it's supported by all modern DL reasoners and its duration is often used as a performance indicator to benchmark reasoning engines (Abburu, 2012). From an application point of view, an ontology should be classified regularly during its development and maintenance in order to detect undesired subsumptions as soon as possible.

We recall some basic concepts of machine learning (ML) (Kotsiantis, 2007) for a better understanding of our study. In any dataset used by machine learning algorithms, every instance is represented using the same set of features. In our case, the instances are ontologies belonging to some corpus and the features are metrics characterizing the ontology content and design. Thus, each ontology is represented by a d-dimensional vector $x^{(i)} = [x_1{}^{(i)}, x_2{}^{(i)}, \ldots, x_d{}^{(i)}]$ called a *feature vector*, where i refers to the i-th ontology in the dataset, $i \in [1, N]$, with N denoting the total number of ontologies and d standing for the total number of features. The latter ones may be continuous, categorical or binary. The learning is called *supervised* when the dataset ontologies are given with known labels. In our context, a label would describe a given reasoner performances when processing the considered ontology, for instance a time-bin. The vector of all labels is specific to one reasoner and denoted $\mathcal{Y}$, where y_i is the label of the i-th ontology. Thus, for each under study reasoner a dataset is built in and designed by $\mathbb{D} = [\mathcal{X}_{N,M} \mid \mathcal{Y}^T]$. Later, the dataset is provided to a supervised learning algorithm in order to train its data and establish a predictive model for its corresponding reasoner. Roughly speaking, a model is a mapping function from a set of features, to a specific label. It would be a mathematical function, a graph or a probability distribution, etc. When a new ontology, which does not belong to the dataset is introduced, then the task is to predict its exact label, using a reasoner predictive model.

3 RELATED WORKS

Works that attempted to predict reasoner's runtime using supervised machine learning techniques are the closest to our context. Authors of (Kang et al., 2012) were the first to apply supervised machine learning techniques (Kotsiantis, 2007) to predict the ontology classification computational time, carried by a specific reasoner. 27 ontology metrics were computed for each ontology. These metrics were previously proposed by a work stressing on ontology design complexity (Zhang et al., 2010). The labels to be predicted were time bins specified by the authors. They learned Random Forest based models for 4 state of art reasoners and obtained high prediction accuracy. Moreover, they proposed an algorithm to compute the impact factors of ontology metrics according to their effectiveness in predicting classification performances for the different reasoners. Kang et al. have further improved their approach, in a more recent work (Kang et al., 2014). They replaced time bin labels by concrete values of reasoner's runtime and proposed more ontology metrics. They learned regression models for 6 widely known reasoners. In addition, they demonstrated the strengths of their predictive models by applying them to the problem of identifying ontology performance *Hotspots* (Gonçalves et al., 2012). On the other hand, in (Sazonau et al., 2014), authors claimed that Kang's et al. metrics based on graph translation of OWL ontologies are not effective. Thus, they proposed another set of metrics and deployed a dimensionality reduction method to remove the intercorrelations between ontology features. They further proposed a new approach to build predictive models based on examining single ontologies rather than the whole corpus.

In all these previous works, machine learning techniques were set up to estimate the amount of time a given reasoner would spend to process any input ontology. However, no attention was paid to assess the correctness of the reasoner derived results. In our opinion, a reasoner that quickly but incorrectly process an ontology is of little use. Therefore, we believe that the effectiveness and the utility of these approaches are still limited in meeting the need of predicting reasoner empirical behaviours. Nevertheless, these works have succeeded to establish highly accurate reasoner performances models, which is a promising advance towards the practical understanding of reasoners. Thus, we are convinced that employing machine learning techniques would be the ultimate approach to gain insights about the reasoner robustness facing real world ontologies. Certainly, previously deployed ML techniques need to be reviewed and improved for a better fit to our learning context.

4 PREDICTING THE ROBUSTNESS OF THE ONTOLOGY REASONERS

In this section, we specify, at first, the notion of reasoner robustness and then, the main steps to automatically learn it from empirical data.

4.1 Why Robustness?

The research question of this paper is whether it is possible to predict the robustness of modern reasoners using the results of their previous running. Worth of mention, the notion of *robustness* differs by the field of research. For software developer, (Mikolàšek, 2009) defined the robustness as *the capability of the system to deliver its functions and to avoid service failures under unforeseen conditions*. Recently, (Gonçalves et al., 2013) bought forward this definition in order to conduct an empirical study about the robustness of DL reasoners. Authors underlined the need to specify the robustness judgement constrains before assessing the reasoners. These constrains are: 1) the range of the input ontologies, 2) the reasoner functional and non functional properties of interest, and 3) the definition of the failure state. The instantiation of the constrains would describe some reasoner usage scenario. Thus, a reasoner may be considered as robust under a certain scenario and non-robust under another.

Given these findings, we started by setting our proper constrains in order to describe an online execution scenario of reasoners. In addition to the computational time which should be maintained as short as possible, we focused on assessing the correctness of the reasoner computed results. Reasoning engines can load and process an ontology to achieve some reasoning task, but they can also deliver quite distinct results. Disagreement over inferences or query answers, computed over one input ontology, would make it hard to provide interoperability in the Semantic Web. Therefore, we consider the reasoner robustness as its ability to correctly achieve a reasoning task for a given ontology within a fixed cut-off time. Consequently, the most robust reasoner over an ontology corpus would be the one satisfying the correctness requirement for the greatest number of ontologies while maintaining the shortest computational time. Based on this specification, we tried to conduct a new reasoner robustness empirical study. One major obstacle we faced was about whether it is possible to automatically check the correctness of the reasoning results. In fact, little works have addressed the issue. As previously outlined by (Gardiner et al., 2006), manually testing the correctness of reasoner inferences would be relatively easy for small ontologies, but usually infeasible for real world ontologies. Authors claimed that the most straightforward way to automatically determine the correctness is by comparing answers derived by different reasoners for the same ontology. Consistency across multiple answers would imply a high probability of correctness, since the reasoners have been designed and implemented independently and using different algorithms. Luckily, this testing approach was implemented in the ORE evaluation Framework (Gonçalves et al., 2013). The reasoner output was checked for correctness by a majority vote, i.e. the result returned by the most reasoners was considered to be correct. Certainly, this is not a faultless method. Improving it would be advantageous for our study, however it is out of the scope of this paper.

Afterwards, we designed four labels that would describe the termination state of a reasoning task. The first label describes the state of success and the others distinguish three types of failure. These labels are: 1) *Correct* (**C**) standing for an execution achieved within the time limit and delivered expected results; 2) *Unexpected* (**U**) in the case of termination within the time limit but delivered results that are not expected, otherwise incorrect; 3) *Timeout* (**T**) in the case of violating the time limit; and 4) *Halt* (**H**) describing a sudden stop of the reasoning process owing to some execution error. Thus, the set $\{C,U,T,H\}$ designs our label space $\mathcal{L}$ admitted for the learning process. Intuitively, the reasoner robustness is close to the reasoning task to be processed. In our study, we focus on the ontology *classification* task (Section 2).

4.2 The Learning Steps

We propose the following steps for predicting the robustness of a given reasoner on individual ontologies. Our learning steps are partially inspired by the earlier work (Kang et al., 2012).

Step 1. Features Identification. We carried a rigorous investigation on the most valuable ontology features, likely to have an impact on the reasoner robustness. The results of this investigation is detailed in Section 5, where we introduced a rich set of features covering a wide range of ontology characteristics. The latter ones depict our features space $\mathcal{F}^d$, where d stands for the space dimension.

Step 2. Ontologies Selection. An ontology corpus $\mathcal{C}(O)$ should be provided to carry out the reasoner evaluations. This corpus should be highly diversified in terms of ontology characteristics, in order to reduce the probability for a given reasoner to encounter only problems it is particularly optimized for. Features, identified in the previous step, will be computed for each ontology $O_i \in \mathcal{C}(O)$, to obtain its corresponding d-dimensional feature vector $X^{(O_i)} \in \mathcal{F}^d$.

Step 3. Reasoner Selection. Any reasoner would be enough for the study; no knowledge about its algorithm neither details about its internal working mechanism are needed for the training process. Thus, given a set of reasoners $\mathcal{S}(\mathcal{R})$, a reasoner $R_k \in \mathcal{S}(\mathcal{R})$ will be iteratively picked to carry on the process.

Step 4. Dataset Building. At this step, each ontology belonging to the corpus $O_i \in \mathcal{C}(O)$ will be processed by the previously selected reasoner R_k to achieve the classification task. Then, the termination state of this task will be retained $l_{R_k}(O_i) \in \{C,U,T,H\}$. Thus, the final training dataset of the selected reasoner R_k, designed by $\mathcal{D}_{R_k}$, is built in by gathering the pairs (feature vector, label), i.e. $\mathcal{D}_{R_k} = \{(X^{(O_i)}, l_{R_k}(X^{(O_i)})), i = 1 \cdots N\}$, where $N = |\mathcal{C}(O)|$.

Step 5. Feature Selection. A huge amount of features does not necessarily improve the accuracy of the prediction model. Commonly, feature selection or dimensionality reduction techniques are applied to identify the relevant features. In our study, we will compare three different methods of feature selection, described in Section 7. Consequently, three variants of the initial reasoner dataset are established, called *featured* datasets, each with a different and eventually reduced subset of ontology features. The initial dataset is also maintained and called *"RAWD"*.

Step 6. Learning and Assessing the Models. Each reasoner dataset $\mathcal{D}^{(j)}_{R_k}$ established in the Step 5 is provided to a supervised machine learning algorithm. The latter train the data and build a reasoner predictive model, M_{R_k} (section 2):

$$M_R \;: X \in \mathcal{F}^d \mapsto \hat{l}_R(X) \in \mathcal{Y} \qquad (1)$$

Worth to cite, we investigated 5 well known supervised machine learning algorithms. Therefore, this step will be repeated as far as the number of algorithms and the number of datasets for a given reasoner. Then, the established models will be evaluated against a bunch of assessment measures. Further, we introduced a method to compare these models and figure out their *"best"* one. Details about the learning algorithms, the assessment measures and the selection procedure are given in Section 8.

Step 7. Unveil the Key Features. Steps 3-6 are repeated K times to cover all reasoners in the set $\mathcal{S}(\mathcal{R})$, with $K = |\mathcal{S}(\mathcal{R})|$. As a result, K best predictive models are identified each for a reasoner, and each having its own most relevant feature subset. Accordingly, we believe that the key ontology features likely to have impact on reasoner robustness are those the most involved across the whole set of best models. We denote this subset as the *Global Key Features*, in contrast to *Local Key Features*, which designs features employed in one given reasoner best predictive model.

5 ONTOLOGY FEATURES (STEP 1)

When reviewing the state of art, we noticed that there is no known, automatic way of constructing *"good"* ontology feature sets. Instead, distinct domain knowledge should be used to identify properties of ontologies that appear likely to provide useful informations. To accomplish our study, we reused some of previously proposed ontology features and defined new ones. Mainly, we discarded those computed based on specific graph translation of the OWL ontology, as there is no agreement of the way an ontology should be translated into a graph (Kang et al., 2014). We split the ontology features into 4 categories: (*i*) size description; (*ii*) expressivity description; (*iii*) structural features; and (*iv*) syntactic features. Within these categories, features are intended to characterize specific aspect of an OWL ontology. Some of the categories are further split up. In overall, 101 ontology features were characterized. Figure 1 lists the main ones. In the following, we will shortly depict our feature categories. We give a more detailed description in (Alaya et al., 2015).

Ontology Size Description. The purpose of this feature category is to characterize the size of the ontology considering both the amount of its terms and axioms. Therefore, we designed 5 features, each records the names of a particular OWL entity. In addition, we computed the number of its axioms (**SA**) and more particularly the logical ones (**SLA**).

Ontology Expressivity Description. We retained two main features to identify the expressivity of the ontology language, namely the OWL profile [1] (**OPR**) and the DL family name (**DFN**).

Ontology Structural Description. We paid a special attention to characterize the taxonomic structure of an ontology, i.e. its inheritance hierarchy. The latter sketches the tree like structure of subsumption relationships between named classes $A \sqsubseteq B$ or named properties $R \sqsubseteq S$. In this category, we gathered various features that have been defined in literature to describe the ontology hierarchies. These are, basically, metrics widely used by the ontology quality evaluation community (Gangemi et al., 2006; Tartir et al., 2005; LePendu et al., 2010). The following subcategories describe the essence of the retained features.

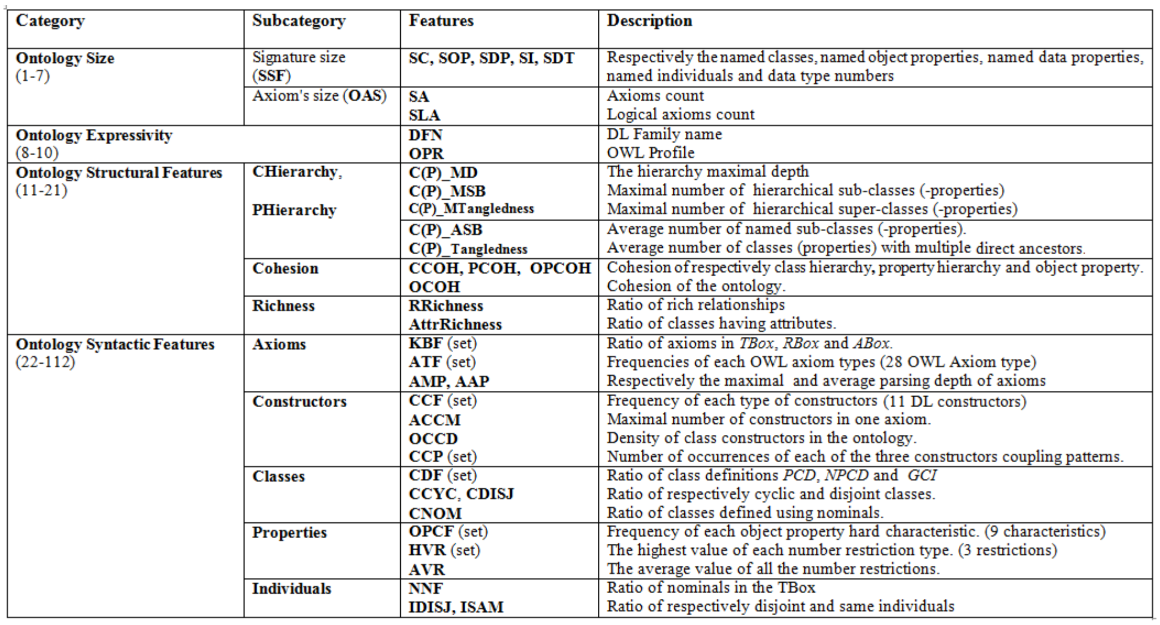
- **Class and Property Hierarchical Features:** five features were specified to outline the design of the class hierarchy (**CHierarchy**). They consider the depth of this tree like structure and the distribution of the subclasses as well as the super-classes. These features were also used to characterize the design of the property hierarchy (**PHierarchy**).
- **Cohesion Features:** the literature provides a plethora of various metrics to design the *Cohesion* of the ontology, otherwise the degree of relatedness of its entities. We retained the ones introduced by (Faezeh and Weichang, 2010). Roughly speaking, the ontology cohesion (**OCOH**) is a weighted aggregation of the class cohesion (**CCOH**), the property cohesion (**PCOH**) and the object property cohesion (**OPCOH**).
- **Schema Richness Features:** we enriched the ontology structural category by two additional features proposed by (Tartir et al., 2005): the schema relationship richness (**RRichness**), and the schema attribute richness (**AttrRichness**). These features are well known for ontology evaluation community as they are part of the *OntoQA* tool.

Ontology Syntactic Features. When collecting features for this category, we conducted an investigation about the main reasoning algorithms (Baader and Sattler, 2000; Motik et al., 2009). Our main purpose was to quantify some of the general theoretical knowledge about DL complexity sources. Thus, we gathered relevant ontology features, that have inspired the implementation of well known reasoning optimization techniques (Horrocks, 2003; Tsarkov et al., 2007). Features of the current group are divided in 6 subcategories, each specific to a particular ontology syntactic component. This organization was inspired by the one introduced by (Kang et al., 2012).

- **Axioms Level:** reasoners process differently each type of axiom with different computational cost (Baader et al., 2003). We introduced two sets of features, (**KBF**) and (**ATF**), in order to characterize the different types of OWL axioms as well as their respective relevance in the ontology. Further, we computed the maximal and the average parsing depth of axioms (**AMP**, **AAP**).
- **Constructors Level:** these concern particularly DL class constructors [2] (Baader et al., 2003). In previous reasoner prediction works (Kang et al.,

[1] For further details about OWL 2 profiles, the reader is kindly referred to http://www.w3.org/TR/owl2-profiles/.

[2] By DL class constructor, we refer to conjunction ($\cup$), disjunction ($\cap$), negation ($\neg$), quantification ($\exists$, $\forall$), etc.



Category	**Subcategory**	**Features**	**Description**
Ontology Size (1-7)	Signature size (**SSF**)	**SC, SOP, SDP, SI, SDT**	Respectively the named classes, named object properties, named data properties, named individuals and data type numbers
	Axiom's size (**OAS**)	**SA**	Axioms count
		SLA	Logical axioms count
Ontology Expressivity (8-10)		**DFN**	DL Family name
		OPR	OWL Profile
Ontology Structural Features (11-21)	**CHierarchy, PHierarchy**	**C(P)_MD**	The hierarchy maximal depth
		C(P)_MSB	Maximal number of hierarchical sub-classes (-properties)
		C(P)_MTangledness	Maximal number of hierarchical super-classes (-properties)
		C(P)_ASB	Average number of named sub-classes (-properties).
		C(P)_Tangledness	Average number of classes (properties) with multiple direct ancestors.
	Cohesion	**CCOH, PCOH, OPCOH**	Cohesion of respectively class hierarchy, property hierarchy and object property.
		OCOH	Cohesion of the ontology.
	Richness	**RRichness**	Ratio of rich relationships
		AttrRichness	Ratio of classes having attributes.
Ontology Syntactic Features (22-112)	**Axioms**	**KBF** (set)	Ratio of axioms in *TBox*, *RBox* and *ABox*.
		ATF (set)	Frequencies of each OWL axiom types (28 OWL Axiom type)
		AMP, AAP	Respectively the maximal and average parsing depth of axioms
	Constructors	**CCF** (set)	Frequency of each type of constructors (11 DL constructors)
		ACCM	Maximal number of constructors in one axiom.
		OCCD	Density of class constructors in the ontology.
		CCP (set)	Number of occurrences of each of the three constructors coupling patterns.
	Classes	**CDF** (set)	Ratio of class definitions *PCD*, *NPCD* and *GCI*
		CCYC, CDISJ	Ratio of respectively cyclic and disjoint classes.
		CNOM	Ratio of classes defined using nominals.
	Properties	**OPCF** (set)	Frequency of each object property hard characteristic. (9 characteristics)
		HVR (set)	The highest value of each number restriction type. (3 restrictions)
		AVR	The average value of all the number restrictions.
	Individuals	**NNF**	Ratio of nominals in the TBox
		IDISJ, ISAM	Ratio of respectively disjoint and same individuals

Figure 1: Ontology Features Catalog.

2012; Kang et al., 2014), authors simply counted axioms that involve potentially hard constructors. However, they missed that one constructor could be invoked more than once in the same axiom. Considering this fact, we proposed a metric to compute a class constructor frequency in the ontology (**CCF**). Moreover, we defined the *density* of the overall constructors (**OCCD**). We also introduced three modelling patterns of particular combinations of constructors. We believe that these combinations, whenever used in an axiom, would increase the inference computational cost.

- **Classes Level:** classes in the ontology could be named ones or complex expressions. They would be cyclic or disjoin ones. In this subcategory, we introduced features that pinpoint different methods to define a class and track their impact in the ontology *TBox* part.
- **Properties Level:** we were interested in capturing two aspects of the ontology properties syntactic description. First, we tried to outline the relevance of specific object property characteristics, such as transitivity, symmetry, reflexivity, etc. Thus, we introduced a metric, (**OPCF**), that measures their respective frequencies in the ontology. Then, we proposed two further metrics (**HVC, AVC**) aiming at examining the impact of using high values in number restrictions.
- **Individuals Level:** we specify some of the interesting characteristics of named individuals that would be declared in the ontology. We examined the ratio of nominals in the *TBox* (**NNF**) and computed the number of individuals declared as equal ones (**ISAM**) or disjoint ones (**IDISJ**).

6 DATA COLLECTION (STEPS 2-4)

To ensure the reliability of reasoner evaluation results, we have chosen to reuse the ORE'2014 evaluation Framework[3], as well as, its test set ontologies. Under this experimental environment, we conducted new DL and EL classification evaluations, for 6 reasoners and using over 1000 ontologies. The motivation behind our choice is multi-fold: first, the event is widely recognized by the Semantic Web community; the ontology corpus is well established and balanced throughout easy and hard cases; and finally the description of reasoner results is consistent with the specification of the robustness criterion, designed in the previous Section 4.1.

Ontologies. In overall, we retained 1087 ontologies from the ORE'2014 corpora[4]. The testing ontologies fall into both the OWL 2 DL and the OWL 2 EL profiles. For each profile, the ontologies were further binned according to their sizes[5]. The latter varies

[3]The ORE'2014 Framework is available at https://github.com/andreas-steigmiller/ore-2014-competition-framework/

[4]The whole corpus of ontologies is available for download at http://zenodo.org/record/10791

[5]The size corresponds to the number of logical axioms.

from small ([100,1000[), going to very large one ($\geq$ 100 000). In addition, the set covers more than 100 distinct DL families. Table 1 describes the distribution of ontology size bins over the OWL profiles. **#O** stands for the number of ontologies.

Table 1: Ontology Test Set Description Summary.

Profile	**#O**	**#O per Size Bin**			
		Small	**Medium**	**Large**	**V-Large**
DL	487	125	124	124	113
EL	600	173	150	150	127

Reasoners. We investigated the 6 best ranked reasoners[6] in both the DL and EL classification challenges of the ORE'2014 competition. These reasoners are included in our study set $\mathcal{S}(\mathcal{R})$. They are namely ***Konclude***, ***MORe***, ***HermiT***, ***TrOWL***, ***FaCT++*** and ***JFact***. We excluded ***ELK*** despite its good results and high rank, as it doesn't support the classification of DL ontologies. We run the ORE Framework in the sequential mode on one machine equipped with an Intel Core I7-4470 CPU running at 3.4GHz and having 32GB RAM, where 12GB were made available for each reasoner. We set the condition of 3 minutes time limit to classify an ontology by a reasoner. This tight schedule would be consistent with the chosen scenario, i.e. the online classification of the ontology.

Table 2 summarizes the new classification challenge results [7]. We did not distinguish between results of DL and EL ontology classification. Reasoners are listed based on their robustness rank, that is the number of correctly processed ontologies within the fixed cutoff time. For each reasoner, Table 2 shows the number of ontologies within a specific robustness bin (Section 4.1) and the range of processing time of the correct cases.

Building the Reasoner Datasets. Each testing ontology was examined to compute and to record its features. Having the evaluation results and the ontology feature vectors, 6 datasets were established each for a specific reasoner. It's important to notice that our reasoner datasets are ***imbalanced***. Based on the description made by (Hu and Dong, 2014), a dataset is considered as imbalanced, if one of the classes (minority class) contains much smaller number of examples than the remaining classes (majority classes). In our context, the classes are reasoner robustness labels. We take as an example the *Konclude*'s dataset described in the table 2, we can easily notice the huge difference

Table 2: Ontology classification results. The time is given in seconds. **#C**, **#U**, **#T** and **#H** stand respectively for the number of ontologies labelled by *Correct*, *Unexpected*, *Timeout* or *Halt* after the classification.

Reasoner	**#C**	**#U**	**#T**	**#H**	**Runtime (correct)**		
					Min	**Max**	**Mean**
Konclude	1030	22	27	8	0.001	54.73	1.48
MORe	971	3	110	3	0.259	144.56	3.77
HermiT	954	29	102	2	0.082	144.12	9.59
TrOWL	927	106	52	2	0.027	109.95	3.36
FaCT++	821	13	205	48	0.019	141.87	4.88
JFact	683	13	323	68	0.019	138.34	9.68

between the number of ontologies labelled as *Correct* **#C**(the majority) and those labelled as *Unexpected* **#U** or *Timeout* **#T** (the minority). However, its obvious that predicting the minor cases is much more interesting for both ontology and reasoner designers, since they describe a failure situation in processing the ontology. In fact, a user would probably want to know, if it would be *safe* to choose *Konclude* to classify its ontology. The learning from imbalanced datasets is considered as one of the most challenging issue in the data mining (Hu and Dong, 2014). We will be considering this aspect when training the predictive models.

7 FEATURE SELECTION (STEP 5)

We started by performing some preprocessing steps on the reasoner datasets. Mainly, we applied *feature selection* methods. The latter ones were designed to recognize the most relevant features, by weeding out the useless ones. In our study, we tried to track down the subset of features, which correlate the most with the robustness of a given reasoner. Thus, we chose to investigate the utility of employing *discretization* (Garcia et al., 2013), as a feature selection technique. Basically, discretization stands for the transformation task of continuous data into discrete bins. More specifically, we opted for the well known Fayyad & Irani's *supervised* discretization method (**MDL**). This technique makes use of the ontology label to achieve the transformation. If the continuous values of an ontology feature are discretized to a single value, then it means the feature is useless for the learning. Consequently, it can be safely removed from the dataset. Seeking of more validity, we decided to compare the discretization results to further feature selection techniques. Thus, we carefully chose two well known methods representative of two distinct categories of feature selection algorithms: first, the *Relief* method (**RLF**), which finds relevant features based on a ranking mechanism; then, the *CfsSubset* method **(CFS)**,

[6]All ORE'2014 reasoners are available for download at https://zenodo.org/record/11145/

[7]All the results of our experiments are available at https://github.com/PhdStudent2015/Classification_Results_2015.

Table 3: Summary of feature selection results. ($\cap$) stands for the intersection of feature subsets.

Dataset	MDL	RLF	CFS	($\cap$)
$\mathcal{D}_{Konclude}$	53	64	8	4
$\mathcal{D}_{MORe}$	74	61	12	8
$\mathcal{D}_{HermiT}$	85	61	15	10
$\mathcal{D}_{TrOWL}$	81	51	13	10
$\mathcal{D}_{FaCT++}$	79	58	16	12
$\mathcal{D}_{JFact}$	86	56	19	15
($\cap$)	50	49	1	1

which selects subsets of features that are highly correlated with the target label while having low intercorrelations. All of aforementioned feature selection methods are available in the machine learning working environment, *Weka* (Hall et al., 2009). Table 3 summarizes the feature selection results when applied to the feature space, $\mathcal{F}^d$, of each of the reasoner datasets. The reported values are the sizes of the reduced feature subsets[8] computed by the respective method. We further investigated the possible presence of common features across these subsets, by computed the size of their respective intersections.

Interestingly enough, in the all cases, we observe that the initial feature dimension was reduced. This would confirm that there are some particular features, which are more correlated to the reasoner robustness bins. However, the reduction level of the feature selection methods varies even for the same reasoner dataset. Nevertheless, it would be noticed that, for each reasoner dataset, feature selection methods agreed on the predictive power of some number of ontology features. Moreover, given a selection method, common features were identified across the datasets of the different reasoners, particularly when using *MDL* and *RLF*. However, the *CFS* method delivered just one shared feature, this is the frequency value (**CCF**) of the OWL constructor `hasSelf`. Worth of cite, this feature also figures in the intersection set of MDL's feature subsets. Certainly, at this stage, it is hard to decide which feature subset is having the most relevant features for a given reasoner. This would be concluded only after training predictive models from these featured datasets. We believe that the subset leading to the most accurate predictive model, is the one having the key ontology features, likely to impact the reasoner robustness. As mentioned in the Section 4.2, we will conduct a comparison between predictive models derived from reasoner initial datasets, ***RAWD***, and those trained form the featured ones respectively by i.e. **MDL**, **RLF** and **CFS**.

[8]The initial dimension of our feature space is 101.

8 LEARNING METHODS OF REASONER ROBUSTNESS (STEP 6)

In this paragraph, different learning algorithms and assessment measures will be shortly described.

Supervised Machine Learning Algorithms. Seeking for diversity in the learning process, we selected 5 candidate algorithms, each one is representative of a distinct and widely recognized category of machine learning algorithms (Kotsiantis, 2007). All the used implementations are available in the *Weka* framework, and was applied with the default parameters of this tool. They are namely: *1) Random Forest* (**RF**) a combination of C4.5 decision tree classifiers; *2) Simple Logistic* (**SL**) a linear logistic regression algorithm; *3) Multilayer Percetron* (**MP**) a back propagation algorithm used to build an Artificial Neural Network model; *4)* ***SMO*** a Support Vector Machine learner with a sequential minimal optimization; and finally *5)* ***IBk*** a K-Nearest-Neighbour algorithm with normalized euclidean distance.

Prediction Accuracy Measures. In previous works (Kang et al., 2012), the accuracy standing for the fraction of the correct predicted labels out of the total number of the dataset samples, was adopted as main evaluation metric of the predictive models. However, accuracy would be misleading in the case of imbalanced datasets as it places more weight on the majority class(es) than the minority one(s). Consequently, high accuracy rates would be reported, even if the predictive model is not necessarily a good one. Thus, we looked for assessment measures known for their appropriateness in the case of imbalanced data. Based on the comparative study conducted by (Hu and Dong, 2014), we retained the following ones. Worth to cite, all of these measures are computed based on the *confusion matrix*, that we describe in what follows.

- ***Confusion Matrix*** it is a square matrix, $L \times L$, where L is the number of labels to be predicted. It shows how the predictions are made by the model. The rows correspond to the known labels of the data, i.e. in our case $\{C,U,T,H\}$. The columns correspond to the predictions made by the model. Each cell (i,j) of the matrix contains the number of ontologies from the dataset that actually have the label i and were predicted as with label j. A perfect prediction model would have its confusion matrix with all elements on the diagonal.
- ***Precision*** (**PR**), ***Recall*** (**RC**), and ***F-Measure*** (**FM**): these are common measures of model effectiveness. *Precision* is a measure of how many er-

rors we make in predicting ontologies as being of some label *l*. On the other hand, *Recall* assesses how good we are in not leaving out ontologies that should have been predicted with the label *l*. However, both measures are misleading when considered separated. Usually, we rather employ the *F-Measure* (**FM**) to assess the model. This measure combines both *Precision* and *Recall* into a single value: $FM = \frac{2 \times RC \times PR}{RC + PR}$.

- ***Kappa Coefficient*** **(κ):** it is used to measure the agreement between the predicted labels and the real ones. The value of *Kappa* lies between -1 and 1, where 0 represents agreement due to chance. The value 1 represents a complete agreement between both values. In rare cases, *Kappa* can be negative.
- ***Matthews Correlation Coefficient*** **(MCC):** it is performance measure barely influenced by imbalanced test sets since it considers mutually accuracies and error rates on all labels. So, it involves all values of the confusion matrix. *MCC* ranges from 1 for a perfect prediction to -1 for the worst possible prediction. A *MCC* value close to 0 indicate a model that performs randomly.

Training and Selecting the Best Predictive Models. A reasoner dataset, D_R, is trained by each of the above-mentioned supervised machine learning algorithms. Thus, five distinct predictive models are learned. We made use of the standard *cross-validation* technique for evaluating these models. We applied a stratified 10-fold cross-validation to ensure the *"generalizability"* of the results. For the sake of simplicity, we only retained the average values of the computed assessment measures over all the cross-validation steps.
We recall, our study covers 6 reasoners, for each 4 datasets were built in and then provided to 5 supervised learning algorithms, each of which have trained it separately and produced its corresponding predictive model.To sum up, 20 models were learned for every reasoner, and assessed using 3 distinct evaluation measures. All these steps were put forward to reveal the reasoner best predictive model, according to the aforementioned assessment measures. Being aware of the amount of data to analyse, we propose to compute a score index per model. By referring to it, we will be able to establish a total order of the reasoner's predictive models. We called it, the *Matthews Kappa Score* (***MKS***). As the acronym would suggest, *MKS* is a weighted aggregation of the *MCC* and the *Kappa* values computed to assess a reasoner model M_R.

$$MKS(M_R) = \frac{\alpha * MCC(M_R) + \beta * Kappa(M_R)}{\alpha + \beta} \quad (2)$$

,where $(\alpha + \beta) = 1$. Thus, the best predictive model for a given reasoner is the one having the maximal value of the *MKS* score:

$$M_{RBest} = arg \max_{M_R^i \in \mathcal{MR}} \left(MKS(M_R^i) \right) \quad (3)$$

,where $\mathcal{MR}$ denotes the set of predictive models learned for a reasoner *R*. In the case where multiple maximal models are identified, the model with the highest *F-Measure* (FM) is selected. We believe that using *MKS* is the straightforward way to automatically determine the best robustness predictive model for a given reasoner. The rational behind this proposal is twofold: first, *MCC* and *Kappa* are widely recognized as powerful assessment measures, even more effective that *FM*, in the case of imbalanced datasets; second both measures are ranging in $[-1, 1]$, which makes the aggregation coherent.

9 RESULTS OF REASONER ROBUSTNESS PREDICTIVE MODELS

The selection of a reasoner best predictive model is achieved within two stages: the best model given a specific reasoner dataset variant; then the best model across all the datasets. Results of each stage will be discussed in the following:

9.1 Best Models from RAWD Datasets

For the sake of brevity, we only report the assessment results of the predictive models learned from reasoner RAWD datasets. Indeed in this type of datasets, ontology feature vectors are full ones, counting 101 distinct features. Table 4 shows the distributions of *F-Measures* (FM), *MCC* and *Kappa* (κ) across the 5 learned models and for every reasoner. The ending line of table 4 displays the name of the reasoner best predictive model, denoted by M^d_{RBest}, under the RAWD datasets. The selection was made according to the MKS values[9]. The assessment results of M^d_{RBest} are denoted in boldface.

A number of important observations can be made from this table. Obviously, the RF algorithm is the most performing learning algorithm in training the RAWD datasets, since it derived the best predictive models for all the 6 reasoners. Moreover, it would be noticed that the maximal reported value of the F-Measure (FM) for the 6 reasoner ranges from 0.86 by RF in the case of TrOWL, going to 0.96 also by RF in the case of Konclude. These close to optimum FM

[9]In our experimentations, we set up $\alpha = \beta = 0.5$.

Table 4: Assessment summary of the reasoners' models learned from *RAWD* datasets.

	Konclude			**MORe**			**HermiT**			**TrOWL**			**FaCT++**			**JFact**		
M_R	FM	MCC	κ	FM	MCC	κ	FM	MCC	κ	FM	MCC	κ	FM	MCC	κ	FM	MCC	κ
RF	**0.96**	**0.66**	**0.57**	**0.95**	**0.79**	**0.77**	**0.89**	**0.66**	**0.65**	**0.86**	**0.51**	**0.48**	**0.91**	**0.75**	**0.74**	**0.92**	**0.86**	**0.86**
SL	0.94	0.40	0.33	0.94	0.74	0.72	0.88	0.62	0.62	0.82	0.34	0.27	0.87	0.66	0.64	0.90	0.80	0.80
MP	0.95	0.53	0.47	0.94	0.68	0.64	0.83	0.48	0.43	0.79	0.24	0.18	0.81	0.50	0.44	0.75	0.51	0.50
IBk	0.95	0.56	0.48	0.94	0.73	0.72	0.87	0.61	0.60	0.84	0.45	0.43	0.88	0.68	0.67	0.90	0.81	0.80
SMO	0.94	0.43	0.36	0.93	0.69	0.68	0.83	0.57	0.57	0.83	0.40	0.33	0.85	0.61	0.57	0.87	0.75	0.75
M^d_{RBest}	**RF (MKS: 0.62)**			**RF (MKS: 0.78)**			**RF(MKS: 0.66)**			**RF (MKS: 0.50)**			**RF (MKS: 0.75)**			**RF (MKS: 0.86)**		

Table 5: Best reasoner models across the different types of datasets.

Dataset	**Konclude**			**MORe**			**HermiT**			**TrOWL**			**FaCT++**			**JFact**		
variant	M^d_{best}	MKS	FM	M^d_{best}	MKS	FM	M^d_{best}	MKS	FM	M^d_{best}	MKS	FM	M^d_{best}	MKS	FM	M^d_{best}	MKS	FM
RAWD	RF	0.62	0.96	RF	0.78	0.95	RF	0.66	0.89	RF	0.50	0.86	RF	0.75	0.91	RF	0.86	0.92
MDL	**RF**	**0.65**	**0.96**	**SL**	**0.80**	**0.96**	**SL**	**0.71**	**0.91**	**MP**	**0.61**	**0.89**	**RF**	**0.77**	**0.92**	**SMO**	**0.89**	**0.92**
RLF	RF	0.62	0.96	RF	0.77	0.95	RF	0.67	0.89	RF	0.50	0.87	RF	0.75	0.91	RF	0.86	0.92
CFS	**RF**	**0.65**	**0.96**	RF	0.74	0.95	RF	0.70	0.90	RF	0.58	0.88	RF	0.74	0.90	RF	0.87	0.90
M_{RBest}	RF+CSF(8f)			SL+MDL(74f)			SL+MDL(85f)			MP+MDL(81f)			RF+MDL(79f)			SMO+MDL(86f)		

values indicate that our proposed set of ontology features entails highly accurate predictive models, even when no feature selection or dimensionality reduction techniques are deployed a priori. Thereby, the relatively high correlation between our ontology features and the reasoners' robustness is confirmed. In overall, the reasoner best models have achieved good MCC and Kappa values, as their MKS scores vary between 0.5 (by RF for TrOWL) and 0.86 (by RF for JFact). This finding proves the ability of the learned models to predict the minor classes (Timeout, Unexpected and Halt). Nevertheless, TrOWL's and Konclude's predictive models are less effective than the other reasoner models. At this stage, we can not conclude, which precise aspect is lowering the MKS scores of both models. It would be either the biased nature of the datasets, or probably some noisy features, which should be removed.

9.2 Best Models Across the Dataset Types

In the table 5, each reasoner best model given a dataset type is reported as well as its assessment values of MKS and FM. The last line of the table sum ups the comparison by revealing, the across datasets, reasoner best robustness predictive model, denoted by M_{RBest}. The dataset type and the dimension of the feature space of the M_{RBest} are also indicated.

Not once the RAWD dataset, with its entire set of 101 ontology features, was listed in the final selection of the reasoners' best models. In most cases, the MKS and FM values of reasoner predictive models derived from featured datasets exceed the ones computed from RAWD models. Therefore, it is quite certain, that restricting the full initial set of ontology features to some particular subsets would improve predictive power of the reasoner robustness models. It would also observed that the size of feature vectors of the M_{RBest} models varies from 8 in the case of Konclude going to 86 in the case of JFact. This observation pinpoints that key ontology features, are close to the reasoner under study. Reasoners implement different optimization techniques to overcome particular complexity sources in the ontology and thus, indicators of their robustness would also vary. Nevertheless, we are not sure which feature selection method would be the most suited in discovering key features for all the reasoners. In fact, the MDL method have outperformed in major cases, but beaten by the CFS's technique in one reasoner instance, i.e. Konclude. For this reasoner, the assessment measures of the MDL's and CFS's predictive models were equal. However, the CFS method have delivered a more compact feature set fully included in the one identified by MDL. Given this fact, we chose the CFS' model as the best one for the reasoner Konclude, since it is much easier to interpret. On the other hand, SL, SMO, MP and RF have shared the podium of the most performing supervised learning algorithms. Considering these findings, we would confirm that there is no ultimate best combination, i.e. feature selection technique and learning algorithm, that suits all the reasoners. Accordingly, we admit that even if the learning process may be maintained the same, the learning techniques must be diversified to grasp, in a better way, the reasoner-specific empirical behaviours. In overall, the learned reasoners' best predictive models showed to be highly accurate, achieving in the most cases, an over to 0.90 FM value. In addition, they are well resisting to the problem of minor classes, as in all cases, their MKS scores were over 0.60. In particular, the MKS of the JFact best predictive model was 0.89, indicating almost a *perfect* predictive model. All of these findings witness the high generalizability of the learned model and their effectiveness in predicting reasoner robustness for the ontology classification task.

9.3 Unveil the Key Features

The high accuracy of the reasoner robustness predictive models trained from featured datasets confirmed the validity of our assumption that particular ontology features would help tracking down the reasoner empirical behaviours. Henceforth, the robustness of reasoners would be explained in terms of the reduced set of ontology features, involved in their respective best predictive models. We have called these subsets, a reasoner *local key features*. We have also investigated possible presence of shared features across the different reasoners' best predictive models, i.e. *global key features*. We pinpointed their contribution to the reasoner robustness prediction, by computing their occurrences in the best models. A group of 8 features were recognized to be the most relevant indicators as they were involved in each of the 6 best predictive models. Namely, these are: the number of named object property (**SOP**), the highest value of the max and the exact number restrictions, i.e. **HVC**(max) and **HVC**(exact), as well as the average value of numbers used in the different cardinality restrictions (**AVR**), two members of the set of class constructors frequency **CCF**(`owl:hasSelf`) and **CCF**(`owl:maxCardinality`), the object property hard characteristics frequency **OPCF**(`owl:symmetricObjectProperty`), and finally the ratio **ATF**(`owl:functionalDataProperty`). Worth of mention, these features are the local key ones of Konclude, but also good indicators for the other reasoners. To have further insights about the contribution of the remaining features, we distinguished four levels of frequencies. Given O_f the number of occurrences of the feature f in the set of the 6 best models, f has *high* frequency if $O_f \geq 5$, *medium* one if $3 \leq O_f \leq 4$, *low* in case $1 \leq O_f \leq 2$ and eventually the feature could be *NotUsed* having $O_f = 0$. In overall, 69 features were found to be highly frequent, 10 were medium ones, 9 had low frequencies and 13 never been involved in the set of reasoners' best predictive models. The important number of highly frequent features indicates the worthiness of our conducted investigation about the most valuable ontology features. As we can not detail the frequencies

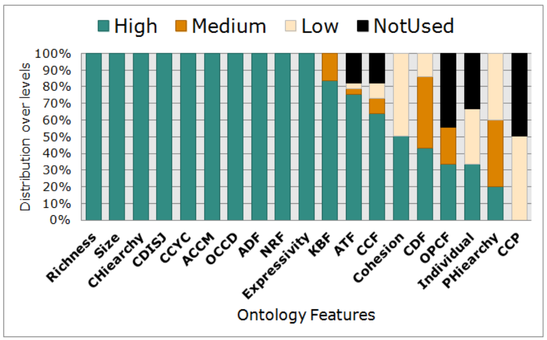


Figure 2: Frequency levels of ontology feature sets.

of each of the 101 features, we choose to report the distribution of feature subcategories, over the frequency levels. Figure 2 illustrates the results.

According to the established inspections, we would assume that the highly frequent features among best reasoner predictive models form a core group of good indicators of reasoner robustness considering the ontology classification task. These features would be recommended as starting points to conduct further improvement in ontology modelling as well as reasoning optimization. We must stress that we are not concluding that the low frequent features are unimportant for the reasoners. Quite the contrary, these features could be very specific to a particular reasoner behaviour.

10 CONCLUSIONS

In this paper, we conducted a rigorous investigation about empirically predicting reasoner robustness. Our main purpose was to be able to explain the empirical behaviour of reasoners when inferring particular ontologies. To fulfill this purpose, we learned predictive models of the robustness of 6 well known reasoners. We put into comparison various supervised learning algorithms and feature selection techniques in order to train best reasoner predictive models. Thanks to this predictive study, we unveiled sets of local and global ontology key features likely to give insights of ontology hardness level against reasoners. We want to stress that these features are key ones under the selected reasoner quality criterion, i.e. the *robustness*. However, we cannot confirm that the same features would be always key ones when moving to different criteria. Investigating this point would be interesting, as to gain more insights about the most relevant features across different reasoning quality criteria. In our case, this purpose would be easily established, since the whole learning steps described in this paper were implemented in one single prototype. Our implementation is generic enough, that it would be applied to any reasoner, with the only requirement of providing enough amount of its running results. Our present work could open further research perspectives. We assume that the most important one would be extending our learning steps by a ranking stage, where reasoners could be compared based on their predicted robustness for a given ontology. Such ranking would made it possible to automatically recommend the most robust reasoner for any input ontology.

REFERENCES

Abburu, S. (2012). A survey on ontology reasoners and comparison. *International Journal of Computer Applications*, 57(17):33–39.

Alaya, N., Lamolle, M., and Yahia, S. B. (2015). Towards unveiling the ontology key features altering reasoner performances. Technical report, IUT of Montreuil, http://arxiv.org/abs/1509.08717, France.

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., and Patel-Schneider, P. F., editors (2003). *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, USA.

Baader, F. and Sattler, U. (2000). Tableau algorithms for description logics. In *Proceedings of the International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX)*.

Bail, S., Glimm, B., Jimnez-Ruiz, E., Matentzoglu, N., Parsia, B., and Steigmiller, A. (2014). Summary ore 2014 competition. In *the 3rd Int. Workshop on OWL Reasoner Evaluation (ORE 2014), Vienna, Austria*.

Faezeh, E. and Weichang, D. (2010). Canadian semantic web. chapter A Modular Approach to Scalable Ontology Development, pages 79–103.

Gangemi, A., Catenacci, C., Ciaramita, M., and Lehmann, J. (2006). Modelling ontology evaluation and validation. In *Proceedings of the 3rd European Semantic Web Conference*.

Garcia, S., Luengo, J., Saez, J., Lopez, V., and Herrera, F. (2013). A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. *IEEE Transactions on Knowledge and Data Engineering*, 25:734–750.

Gardiner, T., Tsarkov, D., and Horrocks, I. (2006). Framework for an automated comparison of description logic reasoners. In *Proceedings of the International Semantic Web Conference, USA*, pages 654–667.

Glimm, B., Horrocks, I., Motik, B., Shearer, R., and Stoilos, G. (2012). A novel approach to ontology classification. *Web Semant.*, 14:84–101.

Gonçalves, R. S., Matentzoglu, N., Parsia, B., and Sattler, U. (2013). The empirical robustness of description logic classification. In *Informal Proceedings of the 26th International Workshop on Description Logics, Ulm, Germany*, pages 197–208.

Gonçalves, R. S., Bail, S., Jiménez-Ruiz, E., Matentzoglu, N., Parsia, B., Glimm, B., and Kazakov, Y. (2013). Owl reasoner evaluation (ore) workshop 2013 results: Short report. In *ORE*, pages 1–18.

Gonçalves, R. S., Parsia, B., and Sattler, U. (2012). Performance heterogeneity and approximate reasoning in description logic ontologies. In *Proceedings of the 11th International Conference on The Semantic Web*, pages 82–98.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., and Reutemann, P. (2009). The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11:10–18.

Horrocks, I. (2003). Implementation and optimisation techniques. In *The Description Logic Handbook: Theory, Implementation, and Applications*, chapter 9, pages 306–346. Cambridge University Press.

Horrocks, I., Kutz, O., and Sattler, U. (2006). The even more irresistible $\mathcal{SROIQ}$. In *Proceedings of the 23rd Benelux Conference on Artificial Intelligence*, pages 57–67.

Hu, B. and Dong, W. (2014). A study on cost behaviors of binary classification measures in class-imbalanced problems. *CoRR*, abs/1403.7100.

Kang, Y.-B., Li, Y.-F., and Krishnaswamy, S. (2012). Predicting reasoning performance using ontology metrics. In *Proceedings of the 11th International Conference on The Semantic Web*, pages 198–214.

Kang, Y.-B., Li, Y.-F., and Krishnaswamy, S. (2014). How long will it take? accurate prediction of ontology reasoning performance. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 80–86.

Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. In *Proceedings of the Emerging Artificial Intelligence Applications in Computer Engineering Conference.*, pages 3–24, The Netherlands. IOS Press.

LePendu, P., Noy, N., Jonquet, C., Alexander, P., Shah, N., and Musen, M. (2010). Optimize first, buy later: Analyzing metrics to ramp-up very large knowledge bases. In *Proceedings of The International Semantic Web Conference*, pages 486–501.

Mikolàšek, V. (2009). Dependability and robustness: State of the art and challenges. In *Software Technologies for Future Dependable Distributed Systems*, pages 25–31.

Motik, B., Shearer, R., and Horrocks, I. (2009). Hypertableau reasoning for description logics. *Journal of Artificial Intelligence Research*, 36:165–228.

OWL Working Group, W. (27 October 2009). *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation. Available at http://www.w3.org/TR/owl2-overview/.

Sazonau, V., Sattler, U., and Brown, G. (2014). Predicting performance of owl reasoners: Locally or globally? In *Proceedings of the Fourteenth International Conference on Principles of Knowledge Representation and Reasoning*.

Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., and Katz, Y. (2007). Pellet: A practical owl-dl reasoner. *Web Semant.*, 5:51–53.

Steigmiller, A., Liebig, T., and Glimm, B. (2014). Konclude: System description. *Web Semantics: Science, Services and Agents on the World Wide Web*, 27(1).

Tartir, S., Arpinar, I. B., Moore, M., Sheth, A. P., and Aleman-Meza, B. (2005). OntoQA: Metric-based ontology quality analysis. In *Proceedings of IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources*.

Tsarkov, D., Horrocks, I., and Patel-Schneider, P. F. (2007). Optimizing terminological reasoning for expressive description logics. *J. of Automated Reasoning*, 39(3):277–316.

Weithöner, T., Liebig, T., Luther, M., Böhm, S., Henke, F., and Noppens, O. (2007). Real-world reasoning with owl. In *Proceedings of the 4th European Conference on The Semantic Web*, pages 296–310.

Zhang, H., Li, Y.-F., and Tan, H. B. K. (2010). Measuring design complexity of semantic web ontologies. *J. Syst. Softw.*, 83(5):803–814.