How Long Will It Take? Accurate Prediction of Ontology Reasoning Performance

Yong-Bin Kang
Monash University, Australia
yongbin.kang@monash.edu

Jeff Z. Pan
University of Aberdeen, UK
jeff.z.pan@abdn.ac.uk

Shonali Krishnaswamy
Inst. for Infocomm Research, Singapore
spkrishna@i2r.a-star.edu.sg

Wudhichart Sawangphol
Monash University, Australia
wudhichart.sawangphol@monash.edu

Yuan-Fang Li
Monash University, Australia
yuanfang.li@monash.edu

Abstract

For expressive ontology languages such as OWL 2 DL, classification is a computationally expensive task—2NExpTime-complete in the worst case. Hence, it is highly desirable to be able to accurately estimate classification time, especially for large and complex ontologies. Recently, machine learning techniques have been successfully applied to predicting the reasoning *hardness category* for a given (ontology, reasoner) pair. In this paper, we further develop predictive models to estimate actual classification time using regression techniques, with ontology metrics as features. Our large-scale experiments on 6 state-of-the-art OWL 2 DL reasoners and more than 450 significantly diverse ontologies demonstrate that the prediction models achieve high accuracy, good generalizability and statistical significance. Such prediction models have a wide range of applications. We demonstrate how they can be used to efficiently and accurately identify *performance hotspots* in a large and complex ontology, an otherwise very time-consuming and resource-intensive task.

1 Introduction

The ontology language OWL 2, the current W3C recommendation, is widely used to represent many complex phenomena in a number of application domains, including software engineering (Pan et al. 2013b) and data management (Li et al. 2013). However, core reasoning tasks such as classification for OWL 2 DL, the most expressive decidable profile of OWL 2, is of worst-case 2NExpTime-complete complexity (Grau et al. 2008b). It has been shown empirically that reasoning on large and complex ontologies in OWL 2 DL and OWL 2 EL (a less expressive profile that enjoys a PTime-complete complexity) can be very time-consuming for state-of-the-art reasoners (Dentler et al. 2011; Kang, Li, and Krishnaswamy 2012b). Such high difficulty of reasoning and the fundamental role inference plays in ontology-based applications make it highly desirable to be able to accurately predict inference performance for ontologies and reasoners.

The problem of predicting reasoning performance was recently investigated (Kang, Li, and Krishnaswamy 2012a), where classification (in machine learning) techniques were applied to predict the *hardness category* (discretized reasoning time) for an (ontology, reasoner) pair. While high accuracy was obtained for 4 state-of-the-art reasoners FaCT++ (Tsarkov and Horrocks 2006), HermiT (Shearer, Motik, and Horrocks 2008), Pellet (Sirin and Parsia 2004) and TrOWL (Thomas, Pan, and Ren 2010; Ren, Pan, and Zhao 2010), that approach has some serious drawbacks. Firstly, it is only able to predict the *hardness category*, but not the actual reasoning time, limiting its utility. Secondly, the choice for hardness categories is ad hoc—5 categories were chosen without empirical evidence of how reasonable or useful these categories are. Thirdly, the values of some metrics may be highly correlated, which may negatively impact prediction accuracy. However, such correlation analysis was not performed. Lastly, the accuracy of this approach suffers from the presence of reasoning time that falls near the boundary between hardness categories.

In this paper, we investigate regression techniques to tackle the above problems and predict actual reasoning time. We develop regression models to predict performance of ontology classification for 6 state-of-the-art and open-source OWL 2 DL reasoners: FaCT++, HermiT, JFact,[1] MORe (Romero, Grau, and Horrocks 2012), Pellet and TrOWL, using syntactic metrics as features. The models are trained on a comprehensive dataset consisting of more than 450 public-domain ontologies, with a very significant variation in their size, metric values and classification time. Our evaluation shows that the models are highly accurate, well generalizable and statistically significant.

Our main contributions can be summarized as follows.

Accurate regression models A regression model is learned for each of the 6 reasoners through 10-fold cross validation. All the models are highly accurate (with R^2 values in $[0.834, 0.942]$ and $RMSE$ values in $[0.81, 1.37]$ for the training sets). At the same time, the models are also very generalizable, as demonstrated by very similar R^2 and $RMSE$ values between the training sets and the separate held-out test sets.

Performance hotspot identification To demonstrate the strengths of the regression models, we also apply the models to the problem of identifying performance hotspots.

[1] http://jfact.sourceforge.net/

Our regression-based algorithm identifies hotspots without running costly reasoning tasks. Experiments show that our identification algorithm can efficiently identify hotspot candidates, significantly outperforming existing approaches in most cases.

2 Preliminaries and Related Work

The continued optimization of sophisticated OWL reasoners has resulted in sustained interests in benchmarking of reasoner performance. Previous works (Horrocks and Patel-Schneider 1998; Pan 2005; Gardiner, Horrocks, and Tsarkov 2006; Bail, Parsia, and Sattler 2010) mostly used small datasets with small to medium-sized ontologies (less than 100 concept expressions). More recently, 8 modern OWL 2 EL reasoners are compared on a number of dimensions (Dentler et al. 2011). The performance dimension is compared on 3 large ontologies (Gene Ontology, NCI Thesaurus and SNOMED CT). It is shown that even for state-of-the-art reasoners, inference on large ontologies is still a very challenging task. In another recent work (Kang, Li, and Krishnaswamy 2012b), 4 OWL 2 DL reasoners were benchmarked on a set of more than 350 ontologies. It is observed that the reasoners exhibit significantly different performance characteristics and hence choosing an efficient reasoner for an ontology is a non-trivial task.

Metrics have been proposed to measure quality (Burton-Jones et al. 2005), complexity (Zhang, Li, and Tan 2010) and cohesion (Yao, Orme, and Etzkorn 2005) of ontologies. A suite of metrics that can be calculated efficiently was proposed (Zhang, Li, and Tan 2010). These metrics are used to measure different aspects of the *design complexity* of ontologies. They include metrics about (1) an ontology as a whole (size of vocabulary, entropy of the node-edge graph, etc.) and (2) individual classes (number of direct super/sub classes, in/out-degree, etc.).

Recently, classifiers were developed for predicting ontology classification performance categories for FaCT++, HermiT, Pellet and TrOWL, using metrics as predictors (Kang, Li, and Krishnaswamy 2012a). The raw reasoning time is discretized into 5 increasingly large categories: $[0s, 0.01s]$, $(0.01s, 1s]$, $(1s, 10s]$, $(10s, 100s]$ and $(100s, \infty)$. High prediction accuracy of over 80% is achieved for all 4 reasoners.

Although highly accurate, the effectiveness and utility of this approach are significantly limited for four reasons. (1) Only the hardness category is predicted, not the actual reasoning time. Especially with large categories such as $(10s, 100s]$ and $(100s, \infty)$, such predictions are only a rough estimate of reasoning performance. (2) The 5 hardness categories are not selected methodologically, but in a rather ad hoc manner. Such a choice of categories, or any choice of categories, may not be suitable for all prediction needs. (3) Some classifiers are sensitive to correlation among features. However, correlation analysis was not performed in (Kang, Li, and Krishnaswamy 2012a). We performed the analysis on the original dataset and found that a significant subset of the metrics (20 out of 58) are indeed highly correlated, with Pearson correlation coefficient over 0.9. (4) reasoning time that falls around category boundaries may adversely affect prediction accuracy. Except for (3), the above problems can be addressed by *regression* analysis.

Regression analysis is a statistical method for estimating a numeric *response variable* from some *predictor variables* (simply predictors). In this paper, for each reasoner, we build a regression model in which the response variable represents predicted classification time and predictors are metrics.

We denote the response variable by Y and the set of predictors by a vector $\mathbf{X}$ (consisting of predictors $X_1, X_2, \ldots, X_k$, where k denotes the number of predictors). The true relationship between Y and X_i's can be approximated by the regression model: $Y \approx f(\mathbf{X}) + \varepsilon$, where ε denotes an error accounting for the failure of the model to fit the data (Friedman, Hastie, and Tibshirani 2001).

Random Forests (Breiman 2001) is an *ensemble* learning method capable of dealing with both classification (in machine learning) and regression problems. The basic premise of ensemble methods is that by combining multiple weak learners, better overall predictive performance can be achieved. Random Forests combine a number of decision trees, each of which is trained using a subset of all instances. Each node of the tree is split based on a random subset of the features. A best split is determined on this subset of features according to some objective function. A binary split is then performed on the node. Random Forests are efficient, robust, and have produced good predictive performance on many real-world data (Breiman 2001). In this paper we train Random Forests-based regression models.

Hotspots (Gonçalves, Parsia, and Sattler 2012) are small subsets of logical axioms whose removal significantly decrease reasoning time for the remaining ontology. Hotspots are performance bottlenecks for reasoning, and they represent refactoring opportunities for ontology developers. The efficient identification of hotspots gives an ontology developer more choices to refactor an ontology, hence is highly desirable.

Definition 1 (Hotspot). *Let $|\mathcal{O}|$ denote the size of an ontology $\mathcal{O}$ and $RT(\mathcal{O}, R)$ denote the reasoning time for an ontology $\mathcal{O}$ using a reasoner R. A subset $\mathcal{M} \subseteq \mathcal{O}$ is a* hotspot *of $\mathcal{O}$ for reasoner R if $|\mathcal{M}| \ll |\mathcal{O}|$ while $RT(\mathcal{O} \setminus \mathcal{M}, R) \ll RT(\mathcal{O}, R)$. Note here symbol $\ll$ means* much less than.

The hotspot identification algorithm (Gonçalves, Parsia, and Sattler 2012) involves identification of signatures that generate $\top\bot^*$-modules (Sattler, Schneider, and Zakharyaschev 2009). A $\top\bot^*$-module $\mathcal{M}$ for a given signature Σ in ontology $\mathcal{O}$ is a syntactically minimal subset of $\mathcal{O}$ that includes all relevant information about Σ.

For a given ontology $\mathcal{O}$ and a reasoner R, a set of hotspots can be identified as follows:

1. Collect satisfiability checking time for all atomic concepts in $\mathcal{O}$ using R.
2. Rank the runtime for all concepts and pick the top one C (with the highest runtime) that has not been considered before.
3. Construct a signature $\Sigma \subseteq \texttt{Sig}(\mathcal{O})$ by including all terms co-occurring with C in some axioms in $\mathcal{O}$, and extract the $\top\bot^*$*-module* $\mathcal{M}$ for Σ from $\mathcal{O}$.

4. Subtract $\mathcal{M}$ from $\mathcal{O}$. Identify $\mathcal{M}$ as a hotspot if $|\mathcal{M}| \ll |\mathcal{O}|$ and $RT(\mathcal{O} \setminus \mathcal{M}, R) \ll RT(\mathcal{O}, R)$.
5. Repeat steps 2–4 until a prescribed number of hotspots is found (e.g., 3) or a prescribed max number of concepts is reached (e.g., 1,000).

One serious drawback of the above algorithm is that it may be very time-consuming, as subontologies in both procedures need to be classified repeatedly for already large and difficult ontologies, and satisfiability needs to be checked for all classes in the hotspot identification procedure. The hotspot detection algorithm was evaluated on a set of 13 large biomedical ontologies, where a number of the $(\mathcal{O}, R)$ pairs were found to contain hotspots. We will present in Section 5 a regression-based approach that is able to identify performance hotspots much more efficiently.

3 Data Collection

Reasoners. 6 state-of-the-art OWL 2 DL reasoners are selected for the experiment: FaCT++ (version 1.5.3), HermiT (version 1.3.6), JFact (version 0.9), MORe (version 0.1.6, with HermiT as the underlying OWL 2 DL reasoner), Pellet (version 2.2.0) and TrOWL (version 0.8).[2]

Ontologies. 451 real-world, public-domain ontologies are collected, some of which from the Tones Ontology Repository and the BioOntology repository.[3] All experiments are conducted on a high-performance server running OS Linux 2.6.18 and Java 1.6 on an Intel Xeon X7560 CPU at 2.27GHz. A maximum of 32GB memory is allocated to each of the 6 reasoners to accommodate potential memory leak in reasoners from repeated invocations.

Consistency checking and classification is performed on each ontology for each reasoner for 3 runs. The average *user time* is recorded. Some ontologies are also excluded from the experiment because they result in parsing errors or they are logically inconsistent. We also apply a 20,000-second timeout as some ontologies take a very long time to classify. Ontologies exceeding this timeout are excluded from the experiment for the respective reasoner, as classification time cannot be collected within a reasonable time frame. Table 1 shows, for each reasoner, the number of ontologies that timed out, the number of ontologies successfully classified, and brief statistics of classification time. It can be observed that the classification time spans over a large range for all the reasoners.

Metrics. We extend the set of 27 metrics proposed previously (Zhang, Li, and Tan 2010; Kang, Li, and Krishnaswamy 2012a) to more comprehensively capture ontology complexity. New metrics include the number of general class inclusions (GCI), number of individuals, and the count of additional types of logical axioms (including reflexive properties, irreflexive properties and domain/range axioms). In total, values for 91 metrics are collected for each ontology. The metrics can be organized in 4 categories:

Table 1: Summary statistics of the dataset. Classification time is in seconds.

Dataset	No. timeout	No.	Classification time			
			Median	Mean	Max	St. dev.
FaCT++	9	349	0.01	77.10	13,400	932.01
HermiT	7	414	0.05	11.02	1,760	107.3
JFact	1	387	0.02	39.05	4,144	318.02
MORe	0	423	0.05	9.98	2,714	134.10
Pellet	8	419	0.04	5.96	539.5	39.62
TrOWL	1	424	0.03	157.2	10,900	1,433.74

ONT: The 24 ontology-level metrics measure the overall size and complexity of an ontology. New metrics include ones that measure

- the ratio of subclass axioms involving someValuesFrom, class union and intersection expressions (ESUB%, DSUB% and CSUB%),
- the ratio of class expressions and axioms in the OWL 2 EL profile (ELCLS% and ELAX%),
- the count and ratio of axioms that involve potentially *hard* language constructs (disjunction, transitive, inverse role and role hierarchy) in HLC and HLC%,
- the number and depth (where depth > 1) of chained class expressions containing someValuesFrom expressions as the subclass (SUBCECHN, DSUBECHN) and as the super class (SUPECHN, DSUPECHN),
- the number and depth of chained class expressions containing class union as the super class (SUPDCHN, DSUPDCHN), as well as
- the number and depth of class expressions containing class intersection as the sub class (SUBCCHN, DSUBCCHN).

CLS: Class-level metrics measure characteristics of OWL classes in an ontology, and they are the same as those used previously (Kang, Li, and Krishnaswamy 2012a). There are 15 CLS metrics in total.

ACE: Anonymous class expression metrics capture different types of class axioms, and they are the same as those used previously (Kang, Li, and Krishnaswamy 2012a). In total, there are 22 ACE metrics.

PRO: Property definition and axiom metrics capture different types of property declarations and axioms. New PRO metrics include the count and ratio of property chains (CHN, CHN%), the ratio of EL properties (ELPROP%), and the number of axioms that make use of role hierarchy, inverse roles and transitive roles (IHR, IIR and ITR). In total, there are 30 PRO metrics.

The metrics have been designed so that they can be calculated efficiently. The complexity of all the metrics calculation algorithm is polynomial in the size of the graph representation of the ontology (number of nodes and edges).

[2]We note that TrOWL is efficiently sound but could be incomplete and the other 5 are using sound and complete algorithms.

[3]http://owl.cs.manchester.ac.uk/repository/, http://www.bioontology.org/

4 Prediction Models for Classification Performance

4.1 Data Preprocessing

Before training regression models for the 6 reasoners, we perform four preprocessing steps on each dataset, which contains metric values as well as reasoning time of all ontologies of the respective reasoner.

Cleansing. Firstly, as the dataset is obtained from multiple repositories, it may contain duplicates. All but one ontology is removed from each set of ontologies with duplicate metric values.

Normalization. Secondly, in each dataset, values of some of the metrics span a large range and are very skewed. In fact, 30 of the 91 metrics span at least 3 orders of magnitude, including 16 that span at least 5 orders of magnitude. As can be seen in Table 1, the response variable (classification time) is very skewed as well, spanning at least 5 orders of magnitude. Hence, for the dataset for each reasoner, we apply a commonly-used log-transformation on metrics spanning a large range. For the same reason, log-transformation is also performed on reasoning time.

Metric removal. Thirdly, near-zero-variance and highly-collinear metrics are removed, as it is widely known that they could negatively affect regression accuracy and overfitting. As a rule of thumb, two metrics with correlation coefficients above 0.9 are considered very highly correlated.

Splitting. Lastly, the dataset of each reasoner is divided up into a training set and a test set in a 80/20 split. The training set is used for training the regression model (with 10-fold cross-validation) and the test set is held out and used later for assessing the performance of the model. Stratified sampling is performed with data points divided into 5 equal percentile groups on the response variable (reasoning time).

After the above preprocessing steps, 46 metrics remain for FaCT++, 52 for HermiT, 55 for MORe, 53 for JFact, 53 for Pellet and 53 for TrOWL, and they are given as input to the regression models for the 6 reasoners

4.2 Prediction Model Construction & Assessment

We build a random forest-based prediction model for each reasoner with the metrics (i.e., predictors) identified in the preprocessing procedure above. Standard 10-fold cross-validation is performed to ensure the generalizalibity of the model. The ontologies and the prediction models are available at http://bit.ly/1hSTy87.

The quality of the regression models is assessed using two widely-used criteria: R^2 and $RMSE$, on the training set. Additionally, we also report the R^2 and $RMSE$ values of the 6 models applied to the testset that is held out during the training process. The values of these quality assessment metrics are shown in Table 2, and their interpretation is given below.

The *coefficient of determination*, R^2, represents the propotion of the variation in the response variable Y that can be explained by the model. For example, 0.853 in M_F indicates that 85.3% of the variation in Y is accounted for by M_F. In other words, the higher the R^2 value, the more accurate the model is. The range of the observed R^2 for the training set is in $[0.834, 0.942]$, indicating that all the models show a good measure for accounting for the variance in Y.

Table 2: Model quality assessment summary.

Model	Training set		Test set	
	R^2	$RMSE$	R_t^2	$RMSE_t$
M_F	0.853	1.24	0.905	0.810
M_H	0.868	1.13	0.913	0.918
M_J	0.834	1.37	0.922	0.786
M_M	0.882	0.81	0.833	0.986
M_P	0.835	1.15	0.904	0.913
M_T	0.942	0.89	0.934	0.909

As we discussed earlier, for each reasoner, the overall dataset is divided into a training and a test set, where only the training set is used for training the regression model. The test set is held out and used later for model assessment purposes. Table 2 above also shows the R^2 values of the regression model applied to the test set for each reasoner. As can be seen, the test set R_t^2 values are very similar to those of the training set, with the values for FaCT++, HermiT, JFact and Pellet exceeding those of the training set. Such results further demonstrate the good generalizability of our models.

The *root mean square error*, $RMSE$, is a widely-used measure of the difference between values predicted by a model and those actually observed, and it represents the sample standard deviation of the differences between predicted and observed values. Hence, the smaller the $RMSE$ value, the more accurate the prediction model is. The $RMSE$ values range in $[0.81, 1.37]$ for the training sets, and in $[0.786, 0.986]$ for the test sets. Similar to the R^2 values, we observe that the test $RMSE_t$ values are close to those of the training sets, with those of FaCT++, HermiT, JFact and Pellet lower than their training counterpart.

It can be observed that for the training set of a number of models, the R^2 values are higher than those of the test set, and the $RMSE$ values are lower than those of the test set. We attribute this phenomenon to the randomness in data division.

The above observations evidently provide an insight into how well the regression models for the 6 reasoners can fit the given data. The similar values of R^2 and $RMSE$ between the training and test sets also suggest high generalizability of our regression models. This indicates that the regression models can be used to accurately predict classification time for the population.

5 Efficient Identification of Performance Hotspots—An Application

Accurate prediction models for classification time have a wide range of applications in ontology engineering and reasoner optimization. In this section we present one such case study, the identification of actual hotspots (Gonçalves, Parsia, and Sattler 2012). Through the case study we demonstrate (1) that the prediction models are well-generalizable beyond the training data, and (2) that the models can be practically applied to non-trivial tasks.

The regression models presented in the previous section can accurately predict the performance of a given ontology. In this subsection we apply the models to the problem of identifying performance hotspots.

Given a pair $(\mathcal{O}, R)$ of ontology and reasoner, the prediction model-based hotspot detection algorithm `detect_hotspot` in Algorithm 1 starts by adaptively extracting a hotspot candidate $\mathcal{M}_C$ for each named concept C (procedure `generate_candidate` on line 4). A $\top\bot^*$-module for Σ_C, a signature including C and its immediately neighboring named entities, is firstly extracted as candidate $\mathcal{M}_C$. If $\mathcal{M}_C$ is larger than a predefined ratio threshold of the size of the ontology, `generate_candidate` then extracts a $\top\bot^*$-module for $\Sigma_C = \{C\}$. While the module is still too large, `generate_candidate` then attempts to extract axioms involving entities that are at most l steps away. This process is repeated until we find a suitable $\mathcal{M}_C$ or $l = 0$. In our experiments we set the ratio threshold to 10% of the number of logical axioms in the ontology and l to 6.

Metric values are then calculated for the candidate $\mathcal{M}_C$. After the same preprocessing steps in Sec 4.1 are applied, the regression model, given the pair $\mathcal{M}_C, R$ as input, predicts the classification time for the candidate $\mathcal{M}_C$ and reasoner R (lines 5–6). The k candidates with the highest predicted classification time is then returned as potential hotspots. The algorithm maintains only k candidates with the highest predicted reasoning time (lines 7–11). k is set to 1,000 in our experiments. Finally, the candidate list is sorted and returned.

The complexity of Algorithm 1 can be determined from its main components: hotspot candidate generation, metrics calculation and regression. The calculation of $\top\bot^*$-module is of polynomial complexity in the size of the ontology and the signature (Grau et al. 2008a). As we showed previously, the metrics calculation algorithm is polynomial in the size of the ontology graph. The regression step takes constant time. Hence, the overall complexity of the algorithm is still polynomial in the size of the ontology. This is lower than the complexity of some common description logics, including $\mathcal{SHOIN}(\mathbf{D})$ (OWL 1 DL) and $\mathcal{SROIQ}(\mathbf{D})$ (OWL 2 DL).

We validate the effectiveness of Algorithm 1 using the same dataset as in (Gonçalves, Parsia, and Sattler 2012), using the 8 ontologies that are known to contain performance hotspots, by running classification on the original ontology $\mathcal{O}$ and the residual ontology $\mathcal{O} \setminus \mathcal{M}_C$, for each candidate $\mathcal{M}_C$ generated by Algorithm 1. We note that these ontologies are not used in training the regression models in Section 4. $\mathcal{M}_C$ is deemed a performance hotspot if its size is much smaller than the original ontology (at most 10% in terms of number of axioms), and the actual reasoning time of the residual ontology $\mathcal{O} \setminus \mathcal{M}_C$ is much lower than that of the original ontology (at most 30%). The detection results of our algorithm is summarized in Table 3 below.

In total 19 (ontology, reasoner) pairs are tested for hotspot, including all 11 pairs from (Gonçalves, Parsia, and Sattler 2012). Our algorithm outperforms the SAT-guided method (Gonçalves, Parsia, and Sattler 2012) for 9 of the 11 pairs, in terms of the number of hotspots found (10 hotspots compared to 3 hotspots) and the number of tests required to find those hotspots. In addition, hotspots are found for 8 additional (ontology, reasoner) pairs, including two each for FaCT++, MORe and TrOWL, and one each for HermiT and Pellet.

Note that we are unable to reproduce the hotspot detection results reported in (Gonçalves, Parsia, and Sattler 2012) for 2 (ontology, reasoner) pairs: (GO-Ext, Pellet) and (NEMO, HermiT). Specifically, with all concepts (30,282 and 1,422 respectively) tested as candidates (but not only 1,000), we

Algorithm 1: Regression-based performance hotspot identification.

```
Input: Ontology $\mathcal{O}$, reasoner $R$ and number of candidates $k$
Output: A sorted list $k$ hotspot candidates $(M_{C_1}, \ldots, M_{C_k})$
        ranked by predicted reasoning time of $M_{C_i}$'s
Algorithm detect_hotspot $(\mathcal{O}, R, k)$
1     $candidates \leftarrow \emptyset$    ▷ $candidates$: $\mathbb{O} \rightarrow \mathbb{R}$ is a
      mapping from a candidate to its
      predicted reasoning time for reasoner
      $R$
2     $CS \leftarrow \{C \mid C \in \mathcal{O} \land C \sqsubseteq \top\}$  ▷ $CS$ is the set
      of all classes
3     for $C \in CS$ do
4         $\mathcal{M}_C \leftarrow$ generate_candidate$(\mathcal{O}, C)$
          ▷ Generate a hotspot candidate
5         $mt_{\mathcal{M}_C} \leftarrow$ preprocess(metrics($\mathcal{M}_C$))
6         $t_{\mathcal{M}_C} \leftarrow$ reg$(mt_{\mathcal{M}_C}, R)$
7         Let $t_{min} \leftarrow \min(\mathrm{ran}(candidates))$ and
          $(m_{min}, t_{min}) \in candidates$  ▷ $m_{min}$ has the
          smallest predicted reasoning time
          $t_{min}$ in $candidates$
8         if $t_{\mathcal{M}_C} > t_{min}$ then
9             if $\#candidates = k$ then
10                $candidates \leftarrow$
                  $candidates \setminus \{(m_{min}, t_{min})\}$ ▷ Remove
                  a smallest element
11            $candidates \leftarrow candidates \cup \{(\mathcal{M}_C, t_{\mathcal{M}_C})\}$
12    $candidates \leftarrow$ sort$(candidates)$    ▷ Sort
      $candidates$ by reasoning time from high
      to low
13    return dom$(candidates)$  ▷ Return the domain
      of $candidates$
Procedure generate_candidate $(\mathcal{O}, C)$
14    $sig_C \leftarrow$ extract_neighbors$(\mathcal{O}, \{C\}, 1)$
15    $\mathcal{M}_C \leftarrow$ extract_$\top\bot^*$_module$(\mathcal{O}, sig_C)$
16    while $|\mathcal{M}_C| \geq \frac{|\mathcal{O}|}{10} \land l > 0$ do
17        $sig_C \leftarrow$ extract_neighbors$(\mathcal{O}, \{C\}, l)$
          ▷ Extract a signature with named
          entities up to $l$ steps away from $C$
18        $\mathcal{M}_C \leftarrow$ extract_axioms$(\mathcal{O}, sig_C)$
19        $l \leftarrow l - 1$
20    if $|\mathcal{M}_C| < \frac{|\mathcal{O}|}{10}$ then return $\mathcal{M}_C$
      else return $\emptyset$     ▷ Return an empty set if
      candidate still over size threshold
```

Table 3: Hotspot identification results using Algorithm 1. The pairs of (ontology, reasoner) where we obtain more hotspots, or our hotspots are obtained using fewer tests than the SAT-guided approach (Gonçalves, Parsia, and Sattler 2012) are highlighted. The '*' besides a reasoner name represents the pair for which we cannot reproduce the results reported in (Gonçalves, Parsia, and Sattler 2012), and '†' besides a reasoner name represents the pair for which hotspots are found by our approach for a reasoner not used in (Gonçalves, Parsia, and Sattler 2012).

Ontology	No. axioms	No. concepts	Reasoner	Avg. RT($\mathcal{O}$)	No. hotspots	No. tests	Avg. RT($\mathcal{O}\backslash M$)	Avg. boost	Avg. $\#M$	Avg. $\%\#\mathcal{O}$
ChEBI	60,085	28,869	Pellet	174.1	10	10	6.90	96.0%	281	0.5%
EFO	7,493	4,143	Pellet	78.80	10	16	19.27	75.5%	93	1.2%
GO-Ext	60,293	30,282	Pellet*	37.76	0	30,282	-	-	-	-
IMGT	1,122	112	HermiT	122.73	3	14	25.24	79.4%	76	6.8%
			MORe†	108.71	10	16	25.45	76.6%	97	8.7%
			Pellet	$> 3,600$	2	14	0.86	$> 99.98\%$	76	6.8%
NEMO	2,405	1,422	FaCT++†	$> 3,600$	4	46	193.22	$> 94.6\%$	154	6.4%
			HermiT*	207.2	0	1,422	–	–	–	–
			Pellet†	275.47	10	825	6.56	97.6%	198	8.2%
OBI	25,257	3,060	FaCT++†	79.96	10	10	1.15	98.6%	1,284	5.1%
			HermiT	190.94	10	10	3.97	97.9%	1,275	5.0%
			JFact	126.54	10	10	4.60	96.4%	1,370	5.4%
			Pellet	307.86	10	10	18.49	94.0%	1,462	5.8%
			TrOWL†	126.94	10	10	2.77	97.8%	1,172	4.6%
VO	8,488	3,530	HermiT†	65.21	10	10	1.0	98.5%	390	4.6%
			Pellet	$> 3,600$	10	10	68.62	$> 98.1\%$	385	4.5%
			TrOWL†	115.31	10	10	2.54	97.8%	385	4.5%
NCIt	116,587	83,722	HermiT	1,020.9	10	10	93.8	90.8%	3,882	3.3%
			MORe†	643.2	10	10	72.64	88.7%	3,838	2.5%

still could not find any hotspot for these two pairs, even though the same hotspot extraction method is used. Further analysis is required to understand this discrepancy.

For the 19 pairs where hotspots are found, on average 62 tests are performed to identify up to 10 hotspots for each pair, with the presence of one pair (NEMO, Pellet) with 825 tests. On average the hotspots are 3.6% of the original ontology size, with an average performance boost of at least 96.2%. For efficiency reasons a one-hour timeout is imposed on ontology classification testing. Still, FaCT++ and Pellet timeout on three ontologies, hence the actual performance boost for these 3 pairs is thus greater.

6 Conclusion

Ontology classification is a time- and resource-consuming process for large and complex ontologies. Therefore, accurately and efficiently predicting actual ontology reasoning time is valuable in ontology-based applications. In this paper, we have developed regression models that accurately predict classification performance for 6 state-of-the-art OWL 2 DL reasoners, on a large and diverse dataset containing 451 ontologies, the largest to the best of our knowledge. Cross-validation and testing on a held-out test set show that our models are highly accurate with good generalizability, with $R^2 \geq 0.834$ for the training set and $R^2 \geq 0.833$ for the test set).

Moreover, we apply the regression models to the problem of efficiently identifying performance hotspots over a set of large BioPortal ontologies, significantly outperforming existing approaches, where the calculation may be otherwise computationally very expensive.

We have planned a number of future work directions: Firstly, we will further improve the prediction model by designing more sophisticated metrics to encompass more features that may impact reasoning performance, and by investigating more sophisticated learning models. Secondly, we will develop algorithms that make use of the prediction models to generate realistic benchmark ontologies (beyond existing ones such as FMA (Pan et al. 2013a)) with predictable performance characteristics. Thirdly, our prediction models focus on classification, a TBox reasoning task. We will investigate the applicability of this technique to ABox reasoning tasks such as realisation and conjunctive query answering.

Acknowledgment

We would like to thank Rafael S. Gonçalves for providing us with the ontology and hotspots dataset. We are also grateful to Yuan Ren for the many helpful discussions.

References

Bail, S.; Parsia, B.; and Sattler, U. 2010. JustBench: a framework for OWL benchmarking. In *Proceedings of the 9th international semantic web conference on The semantic web - Volume Part I*, ISWC'10, 32–47. Springer-Verlag.

Breiman, L. 2001. Random forests. *Mach. Learn.* 45(1):5–32.

Burton-Jones, A.; Storey, V. C.; Sugumaran, V.; and Ahluwalia, P. 2005. A Semiotic Metrics Suite for Assessing the Quality of Ontologies. *Data Knowl. Eng.* 55(1):84–102.

Cudré-Mauroux, P.; Heflin, J.; Sirin, E.; Tudorache, T.; Euzenat, J.; Hauswirth, M.; Parreira, J. X.; Hendler, J.; Schreiber, G.; Bernstein, A.; and Blomqvist, E., eds. 2012. *The Semantic Web - ISWC 2012 - 11th International Semantic Web Conference, Boston, MA, USA, November 11-15, 2012, Proceedings, Part I*, volume 7649 of *Lecture Notes in Computer Science*. Springer.

Dentler, K.; Cornet, R.; ten Teije, A.; and de Keizer, N. 2011. Comparison of reasoners for large ontologies in the OWL 2 EL profile. *Semantic Web Journal* 2(2):71–87.

Friedman, J.; Hastie, T.; and Tibshirani, R. 2001. *The elements of statistical learning*, volume 1. Springer Series in Statistics.

Gardiner, T.; Horrocks, I.; and Tsarkov, D. 2006. Automated benchmarking of description logic reasoners. In *Proceedings of the 2006 International Workshop on Description Logics (DL2006)*.

Gonçalves, R. S.; Parsia, B.; and Sattler, U. 2012. Performance heterogeneity and approximate reasoning in description logic ontologies. In Cudré-Mauroux et al. (2012), 82–98.

Grau, B. C.; Horrocks, I.; Kazakov, Y.; and Sattler, U. 2008a. Modular reuse of ontologies: Theory and practice. *J. Artif. Intell. Res. (JAIR)* 31:273–318.

Grau, B. C.; Horrocks, I.; Motik, B.; Parsia, B.; Patel-Schneider, P.; and Sattler, U. 2008b. OWL 2: The next step for OWL. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web* 6:309–322.

Horrocks, I., and Patel-Schneider, P. F. 1998. DL systems comparison (summary relation). In *Proceedings of the 1998 International Workshop on Description Logics (DL'98)*, volume 11 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Kang, Y.-B.; Li, Y.-F.; and Krishnaswamy, S. 2012a. Predicting reasoning performance using ontology metrics. In Cudré-Mauroux et al. (2012), 198–214.

Kang, Y.-B.; Li, Y.-F.; and Krishnaswamy, S. 2012b. A rigorous characterization of reasoning performance – a tale of four reasoners. In *Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012)*.

Li, Y.-F.; Kennedy, G.; Ngoran, F.; Wu, P.; and Hunter, J. 2013. An ontology-centric architecture for extensible scientific data management systems. *Future Gener. Comput. Syst.* 29(2):641–653.

Pan, J. Z.; Ren, Y.; Jekjantuk, N.; and Garcia, J. 2013a. Reasoning the FMA Ontologies with TrOWL. In *Proc. of the OWL Reasoner Evaluation Workshop (ORE2013)*, volume 1015, 107–113. CEUR-WS.org.

Pan, J. Z.; Staab, S.; Aßmann, U.; Ebert, J.; and Zhao, Y., eds. 2013b. *Ontology-Driven Software Development*. Springer.

Pan, Z. 2005. Benchmarking DL reasoners using realistic ontologies. In Grau, B. C.; Horrocks, I.; Parsia, B.; and Patel-Schneider, P. F., eds., *OWLED*, volume 188 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Ren, Y.; Pan, J. Z.; and Zhao, Y. 2010. Soundness Preserving Approximation for TBox Reasoning. In *the Proc. of the 25th AAAI Conference Conference (AAAI2010)*.

Romero, A. A.; Grau, B. C.; and Horrocks, I. 2012. More: Modular combination of owl reasoners for ontology classification. In Cudré-Mauroux et al. (2012), 1–16.

Sattler, U.; Schneider, T.; and Zakharyaschev, M. 2009. Which kind of module should I extract? In *Proc. of the 2009 Description Logic Workshop (DL 2009)*, CEUR (http://ceur-ws.org/).

Shearer, R.; Motik, B.; and Horrocks, I. 2008. HermiT: A Highly-Efficient OWL Reasoner. In *Proceedings of the 5th International Workshop on OWL: Experiences and Directions (OWLED 2008)*.

Sirin, E., and Parsia, B. 2004. Pellet: An OWL DL Reasoner. In Haaslev, V., and Moller, R., eds., *Proceedings of the International Workshop on Description Logics (DL2004)*.

Thomas, E.; Pan, J. Z.; and Ren, Y. 2010. TrOWL: Tractable OWL 2 Reasoning Infrastructure. In *ESWC (2)*, 431–435. Springer.

Tsarkov, D., and Horrocks, I. 2006. FaCT++ description logic reasoner: System description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006)*, 292–297. Springer.

Yao, H.; Orme, A. M.; and Etzkorn, L. 2005. Cohesion Metrics for Ontology Design and Application. *Journal of Computer Science* 1(1):107–113.

Zhang, H.; Li, Y.-F.; and Tan, H. B. K. 2010. Measuring design complexity of Semantic Web ontologies. *Journal of Systems and Software* 83(5):803–814.