Semantic Web-based Interoperability for Intelligent Agents with PSyKE

Semantic Web-based Interoperability for Intelligent Agents with PSyKE

Federico Sabbatini*[1,2] [0000−0002−0532−6777],
Giovanni Ciatto[2] [0000−0002−1841−8996], and
Andrea Omicini[2] [0000−0002−6655−3869]

[1] Dipartimento di Scienze Pure e Applicate (DiSPeA)
Università degli Studi di Urbino "Carlo Bo", Italy
f.sabbatini1@campus.uniurb.it
[2] Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum—Università di Bologna, Italy
{f.sabbatini, giovanni.ciatto, andrea.omicini}@unibo.it

Abstract. Modern distributed systems require communicating agents to agree on a shared, formal semantics for the data they exchange and operate upon. The Semantic Web offers tools to encode semantics in the form of ontologies, where data is represented in the form knowledge graphs (KG). Applying such tools to intelligent agents equipped with machine learning (ML) capabilities is of particular interest, as it may enable a higher degree of interoperability among heterogeneous agents. Indeed, inputs and outputs of ML models can be formalised through ontologies, while the data they operate upon can be represented as KG. In this paper we explore the combination of Semantic Web tools with *knowledge extraction*—that is, a research line aimed at extracting intelligible rules mimicking the behaviour of ML predictors, with the purpose of explaining their behaviour. Along this line, we study whether and to what extent ontologies and KG can be exploited as both the source and the outcome of a rule extraction procedure. In other words, we investigate the extraction of semantic rules out of sub-symbolic predictors trained upon data as KG—possibly adhering to some ontology. In doing so, we extend our PSyKE framework for rule extraction with Semantic Web support. In practice, we make PSyKE able to *(i)* train ML predictors out of OWL ontologies and RDF knowledge graphs, and *(ii)* extract semantic knowledge out of them, in the form of SWRL rules. A discussion among the major benefits and issues of our approach is provided, along with a description of the overall workflow.

Keywords: explainable AI · knowledge extraction · Semantic Web · intelligent agents · PSyKE

1 Introduction

There are two compelling needs in modern computational systems, namely: *intelligence* of the components, and *interoperability* [37,28] among them. On the

* Corresponding author

one side, intelligence – here intended as the set of cognitive capabilities ranging through image, speech, or text recognition, as well as automated reasoning, deliberation, and planning; plus the criterion by which intertwining them –, is required to improve the effectiveness of computational systems, as well as to ease their interaction with humans. On the other side, interoperability – here intended as the feature by which computational agents supported by different computational technologies or platforms are capable to interact with each others –, is necessary to keep the systems open to the addition of novel agents possibly bringing novel capabilities.

Both intelligence and interoperability are increasingly necessary in computational systems, because of their complexity and pervasiveness. Indeed, many capabilities constituting intelligence are not programmed from scratch, but rather learned from examples, via sub-symbolic machine learning (ML), and a plethora of methods and toolkits are being designed and developed to serve this purpose. Interoperability, in turn, is commonly achieved by letting agents exploit shared syntaxes and semantics for the information they exchange—which of course requires some (sufficiently flexible and expressive) common knowledge representation means to be in place. Needless to say, the large variety of data representation formats and tools for data processing and ML hinders interoperability. *Vice versa*, targeting interoperability as the most relevant concern may constrain the choice of the most adequate method/algorithm/technology for ML, hence potentially hindering the way intelligence is attained. So, intelligence and interoperability are competing features as well.

In this paper we address the issue of favouring data-driven intelligence while preserving interoperability. We do so under the assumption that interoperability is attained by letting intelligent agents adopt knowledge graphs and ontologies for *symbolic* knowledge representation, as supported by Semantic Web (SW henceforth) technologies [5] such as RDF [29], OWL [19], and SWRL [22]. In particular, we focus on the problem of letting these agents interoperate despite the different data formats and schemas, and the different ML algorithms and toolkits they leverage upon when dealing with the sub-symbolic knowledge.

Within the explainable artificial intelligence [17] community, sub-symbolic knowledge can be tackled via symbolic knowledge extraction (SKE), which provides methods and algorithms to distil symbolic knowledge – mostly in the form of rule lists or trees – out of sub-symbolic predictors. There, a key goal for SKE is to make the sub-symbolic knowledge agents acquire from data intelligible to human beings. Conversely, in this paper we address the problem of extracting semantic knowledge to reach inter-agent explainability—reifying the vision proposed in [8]. In other words, we aim at letting agents extract *semantic* knowledge – in the form of SWRL rules, possibly adhering to some OWL ontology – out of ML predictors of any shape—hence enabling a wider degree of interoperability among heterogeneous distributed agents. This is clearly based on the assumption that agents are levering upon a data-driven, ML-based approach to support their intelligent behaviour—e.g. by wrapping trained neural networks or other predictors of any sorts, or by training them as part of their operation.

Along this line, we focus on extending the PSyKE framework [35] for symbolic knowledge extraction towards SW-compatibility. Indeed, at the time of writing, PSyKE consists of a Python library supporting the extraction of Prolog rules out of ML predictors of any sort and shape—there including neural networks. However, while this is very interesting for human beings and for logic programmers, intelligent agents may need knowledge to be extracted in semantic form—possibly, out of predictors trained upon semantic data.

Accordingly, in this paper we propose an extension for PSyKE's design and technology aimed at supporting SW technologies. Notably, PSyKE is a general-purpose framework supporting the extraction of logic rules out of ML predictors, via multiple algorithms. To the best of our knowledge, it is also the only available technology providing a general API for symbolic knowledge extraction. Currently, however, PSyKE is only capable of extracting symbolic knowledge in the form of Horn clauses lists (a.k.a. Prolog theories).

In this work, we study whether and to what extent ontologies and knowledge graphs (KG) can be exploited as both the source and the outcome of a rule extraction procedure. In other words, we investigate the extraction of semantic rules out of sub-symbolic predictors trained upon data as KG—possibly adhering to an ontology. In practice, we make PSyKE able to train ML predictors out of OWL ontologies and RDF knowledge graphs, and then to extract semantic knowledge out of them, in the form of SWRL rules. A discussion among the major benefits and issues of our approach is provided as well, along with a description of the overall workflow.

Accordingly, the remainder of this work is structured as follows. In Section 2 a brief overview on the main topics covered in this paper is reported. In Section 3 the extended design of PSyKE is presented, whereas in Section 4 a concrete applicative example is shown. Open issues are summarised in Section 5; while conclusions are drawn in Section 6.

2 State of the Art

In this section, we provide a brief description of the main topics covered by this paper, namely: symbolic knowledge extraction, the PSyKE framework, and the Semantic Web. Furthermore, as our contribution relies on the Owlready Python library [26], we also provide an overview of its main features.

2.1 Symbolic Knowledge Extraction

ML techniques and, in particular, (deep) artificial neural networks (ANN) are more and more applied to face a growing amount of real-world problems. Despite their impressive predictive capabilities, one of the most critical issues related to most ML solutions is their black-box (BB) behaviour [27], intended as their inability in providing to human users a comprehensible explanation about either the knowledge they acquired during the training, or the logic leading from a

given input to the corresponding output prediction. This *opacity* is inherently bound to the *sub-symbolic* nature of ML algorithms.

Several solutions have been suggested by the XAI community to overcome this inconvenience. One of them is the adoption of more (human-)interpretable models [34], even though they may not have equivalent predictive capabilities. Alternatively, inspection techniques are applicable to the BB predictors [16] to obtain interpretable [9] outputs without sacrificing the underlying model predictive capability.

Among the most promising methods to derive *post-hoc* explanations there are symbolic knowledge extraction techniques, based on the construction of a *symbolic* model that mimics the behaviour of a BB predictor in terms of input-output relationship. Symbols adopted by SKE algorithms to represent intelligible knowledge are, for instance, lists or trees of rules [14,24,31,32,33] that can be used to make human-understandable predictions as well as to shed a light on the internal behaviour of a BB model.

SKE is a precious resource when dealing with critical application fields – e.g., healthcare [6,13,18], financial forecasting [3,4,43], credit card screening [38], but not only [2,21] –, where it is not acceptable to make decisions on the basis of "blind" AI predictions. For example, consider the case of an autonomous vehicle that does not steer when it is about to collide with a pedestrian. This unexpected behaviour may be caused by a misclassification of the pedestrian or by a wrong conclusion corresponding to the detection of a pedestrian on the road.

Within the scope of this paper, SKE is the key mechanism by which semantic knowledge can be grasped by trained ML predictors rather than being manually crafted by humans. Under this perspective, ML predictors can be considered as the tools by which sub-symbolic knowledge is extracted from data, whereas SKE can be considered as the tool by which knowledge is converted from sub-symbolic to symbolic form.

2.2 PSyKE

PSyKE [35] is a general-purpose software library providing a unified application programming interface (API) for SKE algorithms. In other words, it provides a common way of exploiting different SKE algorithms on different kinds of ML predictors.

At the time of writing, PSyKE supports several *pedagogical* [1] SKE procedures (e.g., [7,10,11,23,36]), for both supervised classification and regression tasks, letting users choose the most suitable extraction method w.r.t. the data and task at hand. The library also provides several utilities to help users with ML-related tasks—e.g., data set manipulation, performance assessment, algorithm comparison.

W.r.t. our goal of making PSyKE SW-compatible, two major aspects are currently lacking, namely: *(i)* the capability of training ML predictors out of knowledge graphs, and *(ii)* the capability of extracting knowledge in SWRL format, possibly adhering to an OWL ontology. Indeed, so far, PSyKE enables users to train ML algorithms from data sets structured as *tables* – where each

column is a feature and each row is an instance – and to extract knowledge in the form of lists of Horn clauses—in particular, rules in Prolog format. Hence, in the remainder of this paper, we discuss how the design and implementation of PSyKE can be extended to enable such capabilities.

2.3 Semantic Web

The Semantic Web is considered since its birth as a tool for interoperability—between humans and machines as well as between software agents [5]. It aims at allowing automated systems to consciously handle contents available on the Web by providing methods to formalise data together with their implicit semantics and inference rules useful to reason with the data. One of the Semantic Web enabling technologies is the Resource Description Framework (RDF) [29], used to represent objects and relationships between them. Concepts described through RDF – named *resources* – are represented by a Universal Resource Identifier (URI) and encoded as triples representing a subject (i.e., a thing), a verb (i.e., a property or a relationship) and an object (i.e., a value or another thing). In the SW vision semantic interoperability is possible thanks to *ontologic languages* and *ontologies*—i.e., taxonomies defining classes, subclasses, properties, relationships and inference rules.

The Web Ontology Language (OWL) [19] is an ontologic language extending RDF with First-Order Logic expressiveness. It is expressed in triples as well, but it also provides a semantics for the represented RDF resources, enabling the definition of classes and properties, hierarchies – i.e., subclasses and subproperties –, restrictions and peculiar characteristics—e.g., inverse or transitive properties.

Inductive rules involving Semantic Web entities are represented thanks to the Semantic Web Rule Language (SWRL) [22]. Rules are expressed in terms of OWL concepts—i.e., classes, properties and particular individuals. SWRL also provides a number of built-in concepts similar to the standard Prolog predicates, for instance to represent arithmetic, relational and commonly used string operators. SWRL rules can be added to OWL ontologies and are compatible with automated reasoners. SWRL and other ontologic languages make it possible to perform automatic reasoning with Web resources. Examples of automated reasoners are HermiT [15,30,39] and Pellet [41,42].

Thanks to the Semantic Web heterogeneous agents acting inside a distributed system can communicate and exchange data even if they are not explicitly designed to cooperate together. This is possible since all the involved entities agree on an implicit semantics through a shared ontology.

On the other hand, the main ontology drawbacks are the time and human expertise required to build them and the implications deriving from their decoupled structure, possibly leading to incompleteness or inconsistency of the ontologies [12,20,40].

2.4 Owlready

According to its online documentation,[3] Owlready [26] is a Python package enabling ontology-oriented programming. It considers OWL ontologies as Python objects, allowing users to modify and save them, as well as to add methods to the classes defined in the ontologies. In addition, Owlready supports semantic reasoning via the HermiT or Pellet reasoners.

From a technical perspective, Owlready supports the construction of OWL ontologies, as well as the loading and inspection of pre-existing ontologies. As practical features, it supports enumerating the classes, individuals and rules contained in a given ontology, as well as all the properties of a class. Furthermore, Owlready supports SWRL rules, enabling the empowerment of OWL ontology expressiveness with *if-then* logic rules. In the conditional part of rules it is possible to insert typical SWRL built-in predicates involving class properties and constant values.

The HermiT and Pellet reasoners included in Owlready make it possible to perform automated reasoning on the basis of the information included in ontologies. Thus, they may be exploited to grant predictive capabilities to ontologies, especially in classification tasks, if they contains SWRL rules explaining how to perform such predictions. In addition, they allow users to highlight inconsistencies in the ontologies—for instance, rules in contradictions between each others or w.r.t. specific individuals.

In this work we use version 2 of Owlready.

3 Interoperability via PSyKE

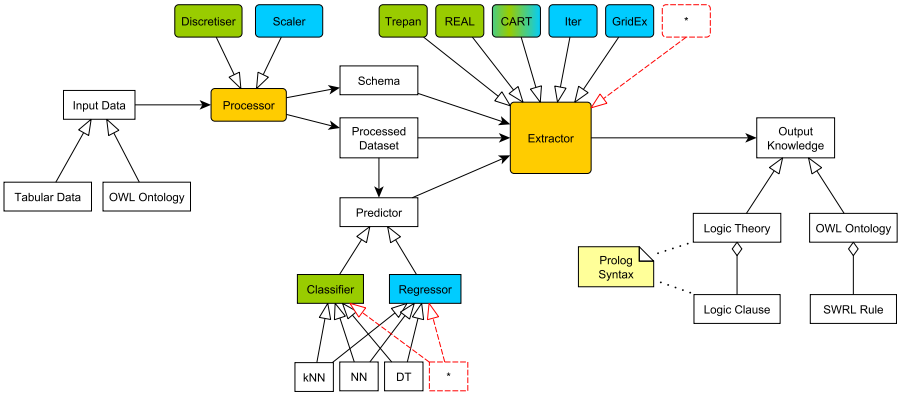

Fig. 1. PSyKE design.

[3] https://owlready2.readthedocs.io [Online; last accessed February 28, 2022]

PSyKE is a software tool to extract logic rules from opaque ML predictors. It offers several interoperable and interchangeable SKE algorithms, to be chosen on the basis of the task and data at hand, and it exposes a unified application programming interface (API) for all of them. The architecture of PSyKE – reported in Figure 1 – is designed around the idea of *extractor*, i.e., a generic knowledge extraction procedure accepting BB predictors as inputs, together with the data set used during their training, and producing theories of logic rules as output. The input data set should be provided along with its corresponding schema—that is, a formal description of the data set's features in terms of names and data types. This allows extractors to inspect the knowledge the BB predictor has acquired from that data set, as well as to take data- and type-driven decisions while the extraction procedure is going on. This, in turn, enables the generation of more readable logic rules, possibly leveraging on the feature names and data types described into the data schema.

As the reader may notice from Figure 1, any data set provided to some extractor is assumed to be manipulated by some *processor* entity. This is where input data is discretised or scaled through one of the utilities included in PSyKE. Of course, where and how data should be discretised or scaled really depends on the task at hand, other than on the requirements of the particular extraction algorithm chosen by the user (among the many available in PSyKE).

The presented architecture is here extended to support SW-related features, following the purpose of making PSyKE suitable as an interoperability tool for intelligent heterogeneous agents. While pursuing this goal, we assume agents to be entities capable to perform ML tasks on locally available data. In our vision – depicted in Figure 2 –, agents apply SKE techniques to gain the ability to represent in a symbolic form what they learn from local data. The extracted symbolic knowledge may then be exchanged among agents, by assuming that its form follows a shared syntax and semantics. Semantic Web technologies – such as RDF, OWL, and SWRL – fit the picture by playing exactly this role.

Along this line, we enrich the design of PSyKE to loosen its strict dependency on *(i)* input tabular data sets and *(ii)* output Prolog theories. In other words, the extended version of PSyKE accepts more than one kind of input data and produces results in more than one format, to be chosen by the user. In more details, PSyKE gains the ability to work on input data encoded in the form of OWL ontologies and to represent the extracted knowledge as agent-interpretable SWRL rules inserted into an OWL ontology, for instance the one given as input. This brings at least two benefits to the users of PSyKE. The first benefit is that in this way it is possible to apply ML techniques to data gathered by intelligent agents and encoded as knowledge graphs. On the other hand, the output knowledge is no longer bounded to be a human-readable Prolog theory alone, but it can now be represented in a new format that is suitable to be exchanged between heterogeneous entities as well.

To serve these purposes, PSyKE is enriched with two further modules: one aimed at loading input training data from either tabular (e.g. CSV files) or semantic (e.g. OWL or RDF files) sources, and the other aimed at representing

the extracted rules in some output format of choice—currently, either Prolog or SWRL, which can be chosen interchangeably without information loss.

Behind the scenes, both modules rely upon software utilities aimed at converting tabular data in semantic form and *vice versa*. These processes are called *relationalisation* and *propositionalisation*, respectively. Propositionalisation is required to apply existing ML algorithms to data that is not represented according to the expected format—i.e., a tensor. Thanks to this conversion it is possible to obtain a proper representation to avoid reinventing all the ML layer of PSyKE. Conversely, relationalisation is necessary when there is the need to extract semantic rules from tabular data sets, since the output SWRL rules produced by PSyKE assume the existence of an OWL ontology containing the definitions of the classes involved in the SWRL rules.

It is worth mentioning that, besides interoperability, the proposed extensions bring key benefits to the SKE playground as well. For instance, by extracting rules in semantic format, one may detect the presence of inconsistencies in the extracted rules themselves, as well as between these rules and the individuals of an ontology

Accordingly, in the following subsections we delve into the details of *(i)* how SWRL rules are constructed, *(ii)* how propositionalisation and relationalisation work, and *(iii)* which benefits Semantic Web technologies brings to the SKE playground.

3.1 Output Rules in SWRL Format

The extension of PSyKE presented in this paper is able to output extracted knowledge in the form of SWRL rules, more agent-interpretable than the Prolog rules supported in the previous version of our framework. SWRL rules are structured as logical implications, where a list of preconditions imply a postcondition—that is, if all of the preconditions are satisfied, then the postcondition is true. All conditions are expressed as triples composed of subject, predicate and object. Subjects are generally data set instances or properties. In the first case the predicate is a "has-a" relationship and the object is a property. Otherwise, the predicate is a relational operator and the object is a constant value. Property names recall those of the input features to ease human-readability, even if it is not the definite goal of this work.

For problems described by m input features the precondition list is composed of at least $m + 3$ triples, since *(i)* the first predicate ensures that the instance at hand belongs to a class defined in the ontology; *(ii)* the following m predicates bind each input feature to a variable to be used in other predicates; *(iii)* one predicate is used in the same manner for the output variable; *(iv)* at least one predicate discriminates the rule by introducing some constraints on the input variables. Since rules are not ordered, it is not possible to have the equivalent of Prolog facts, because facts would be default rules always true, causing inconsistencies with any other rule having a different output value.

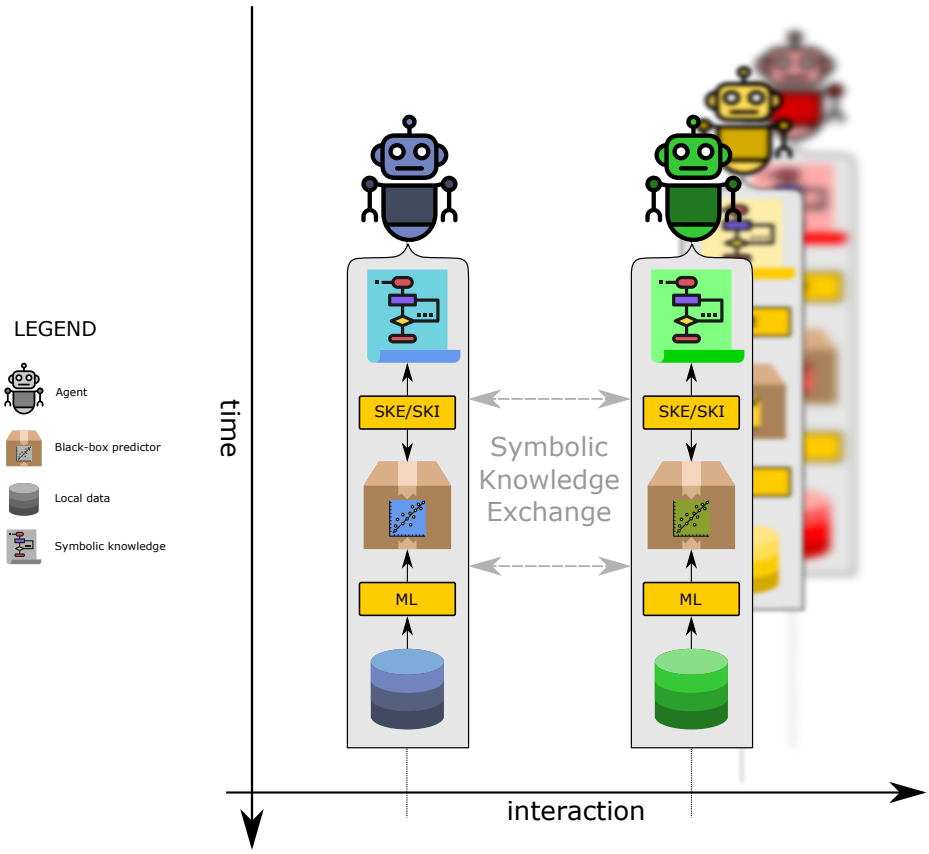

Fig. 2. Example of generic agent workflow and interaction.

A mono-dimensional classification task on a data set having m input features is represented by the following SWRL rules:

$$\begin{aligned} &\texttt{Object(?o)}, \\ &\texttt{Prop\_1(?o, ?p1)}, \ldots, \texttt{Prop\_m(?o, ?pm)}, \\ &\texttt{Cond(?p1, c11)}, \ldots, \texttt{Cond(?p1, c1j)}, \\ &\qquad\qquad\qquad\qquad\qquad\qquad \ldots, \\ &\texttt{Cond(?pm, cm1)}, \ldots, \texttt{Cond(?pm, cmk)} \implies \texttt{Output(?o, out)}, \end{aligned}$$

where `o` is an entity of class `Object` (first row), having `Prop_1, ..., Prop_m` properties which corresponding values are represented by the `p1, ..., pm` variables (second row). Each variable `pi` is subject to a non-fixed number of conditions `Cond`. Conditions express equality or inequality constraints related to the variables `pi` w.r.t. the specified constants `cij`. Specifically, available conditions are equal, not equal, less than, less or equal than, greater than and greater or equal than. Constant values can be numeric or strings. If all the preconditions are true – i.e., if `o` is an object of a certain class with specific properties and these properties assume values in defined ranges –, then the value of the `Output` property of `o` is equal to the constant `out`. Such property represents the target of the classification task.

A similar rule structure is adopted for regression tasks having constant output values and, in general, for all BB having discrete outputs. So far, regression SWRL rules are not supported, since their triple-based syntax does not allow to encode linear combinations of input variables without having an explosion of the number of rule preconditions.

3.2 Propositionalisation

Propositionalisation [25] accepts data encoded as knowledge graphs as input, and outputs the equivalent tabular representation. This means that all the entities and relations mentioned in the KG are extracted and rearranged as a table.

In PSyKE the propositionalisation of an ontology containing n individuals having m distinct properties produces a tabular data set composed of n rows and m columns and is performed as follows. Individuals contained in an ontology are sequentially examined and for each individual a new row is created in the tabular translation. For each ontology individual the corresponding properties are copied into the proper column of the table.

Formally, the i-th individual of class `ClassName` encoded in an OWL ontology, having m properties of which the first is numeric and the second and the last are string, is encoded as follows:

```
<ClassName rdf:about="#individual_i">
 <rdf:type
  rdf:resource="http://www.w3.org/2002/07/owl#NamedIndividual"/>
 <Prop_1 rdf:datatype="http://www.w3.org/2001/XMLSchema#decimal">v_1</Prop_1>
 <Prop_2 rdf:datatype="http://www.w3.org/2001/XMLSchema#string">v_2</Prop_2>
 ...
 <Prop_m rdf:datatype="http://www.w3.org/2001/XMLSchema#string">v_m</Prop_m>
</ClassName>
```

The individual can be propositionalised into the following tabular instance:

#	Prop_1	Prop_2	...	Prop_m
...	...	...	...	...
i	v_1	v_2	...	v_m
...	...	...	...	...

3.3 Relationalisation

Dually w.r.t. propositionalisation, relationalisation accepts tabular data as input and it outputs an equivalent OWL ontology. This is achieved by creating a class for the concept represented by the table and a property for each data set column, having the domain equal to the created class and a range equal to the type of the column data. Then, for each row of the table an individual is added to the ontology, by copying the corresponding table values. This means that in PSyKE a table having n rows and m columns is converted to an ontology with 1 class, m functional properties and n individuals.

Starting from the table shown in the previous subsection, and assuming the same conditions about the property types, the following class and functional properties are produced:

```
<owl:Class rdf:about="#ClassName">
 <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>

<owl:DatatypeProperty rdf:about="#Prop_1">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
 <rdfs:domain rdf:resource="#ClassName"/>
 <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#Prop_2">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
 <rdfs:domain rdf:resource="#ClassName"/>
 <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>

...

<owl:DatatypeProperty rdf:about="#Prop_m">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
 <rdfs:domain rdf:resource="#ClassName"/>
 <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
```

The individual corresponding to the i-th row of the table is exactly the one reported at the beginning of the previous Subsection.

3.4 Semantic Web for SKE: Pros and Cons

The extraction of knowledge adhering to a specific SW format enables, above all, direct interoperability between heterogeneous intelligent agents, intended as mutual exchange of symbolic knowledge. Ontologies containing SWRL rules represent self-contained objects able to perform (agent-)interpretable predictions but not only. Indeed, the reasoning capabilities provided by ontologies can be

exploited to check the quality of the extracted knowledge in terms of *consistency*. An ontology is consistent if *(i)* all the SWRL rules do not express contradictions among each others, and *(ii)* all the individuals follow at the same time all the ontology rules, without contradictions. For instance, two rules having the same preconditions but opposite postconditions produce a contradiction, so the ontology containing them is not consistent. Analogously, there is a contradiction – and therefore an inconsistency – for a classification task when the extracted rules list contains *(i)* a rule r_1 with a precondition on the input feature f_1 and corresponding postcondition equal to class c_1, and *(ii)* a different rule r_2 with a precondition on the input feature $f_2 \neq f_1$ and corresponding postcondition equal to class $c_2 \neq c_1$. Hence, if there is an individual with features f_1 and f_2 satisfying at the same time both r_1 and r_2 the ontology is not consistent.

In the case of PSyKE extraction mechanism, contradictions between rules may occur after using a knowledge extraction algorithm that do not provide exclusive rules, whereas contradictions between individuals and rules may occur when extracted rules are inserted into an ontology containing individuals with known output values, if such values are different w.r.t. those provided by the rules—for instance, if some rules predict wrong labels in classification tasks. However, it is possible to reason in presence of overlapping rules, if for each individual to be analysed there is at most one SWRL rule encompassing it.

The inference of missing data is the mechanism enabling to make predictions based on the ontology and without any supplementary tool. Since all the extracted rules refer to the output variable of a data set on the basis of its input variables, these rules can be exploited to predict the output of unknown instances, if the required inputs are provided—as happens with any predictive model. This inference mechanism may be exploited to remove inconsistencies between individuals obtained from a data set – and thus containing known output values – and rules extracted via SKE methods—possibly leading to output values different from the true ones, since they approximate a BB approximating, in turn, the input/output relationship of the data set itself. It is sufficient to remove the true output values and to use the inferred output instead.

4 An Example: The Iris Data Set

In the following we provide a simple relationalisation example performed with PSyKE on a real-world data set, then we exploit its extended capabilities to extract symbolic knowledge in the form of SWRL rules. We use the well-known Iris data set,[4] composed of 150 instances representing individuals of Iris plants. Each exemplary is described by 4 numeric input features – i.e., width and length of petals and sepals – and a single output label—corresponding to the Iris species. The data set is commonly used to perform classification tasks and there are 3 possible different classes.

For our experiment we consider the tabular data set available on the UCI ML Repository. We use it to train a k-NN predictor and then we relationalise

[4] https://archive.ics.uci.edu/ml/datasets/iris [Online; last accessed 5 March 2022]

it to obtain an equivalent knowledge graph. Finally, we extract knowledge from the k-NN via the CART algorithm in the form of SWRL rules, merging them with the knowledge graph to obtain the resulting OWL ontology.

#	SepalLength	SepalWidth	PetalLength	PetalWidth	iris
1	5.1	3.5	1.4	0.2	setosa
2	7.0	3.2	4.7	1.4	virginica
3	6.3	3.3	6.0	2.5	versicolor
⋮	⋮	⋮	⋮	⋮	⋮

Table 1. A portion of the Iris dataset

Table 1 depicts (a portion of) the Iris dataset and its structure. Conversely, the following listing reports the corresponding ontology structure—i.e., the Iris class and two example properties (the sepal length, real-valued, and the output iris class, having type string):

```
<owl:Class rdf:about="#Iris">
 <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>

<owl:DatatypeProperty rdf:about="#SepalLength">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
 <rdfs:domain rdf:resource="#Iris"/>
 <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="#iris">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
 <rdfs:domain rdf:resource="#Iris"/>
 <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
```

An example of individual – assuming all input and output features have been defined –, is the following:

```
<Iris rdf:about="#iris1">
 <rdf:type
  rdf:resource="http://www.w3.org/2002/07/owl#NamedIndividual"/>
 <SepalLength rdf:datatype="http://www.w3.org/2001/XMLSchema#decimal">
  5.1
 </SepalLength>
 <SepalWidth rdf:datatype="http://www.w3.org/2001/XMLSchema#decimal">
  3.5
 </SepalWidth>
 <PetalLength rdf:datatype="http://www.w3.org/2001/XMLSchema#decimal">
  1.4
 </PetalLength>
 <PetalWidth rdf:datatype="http://www.w3.org/2001/XMLSchema#decimal">
  0.2
 </PetalWidth>
 <iris rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
  setosa
 </iris>
</Iris>
```

Finally, the knowledge graph representing the complete domain structure and the other 2 individuals previously described is graphically represented in Figure 3.

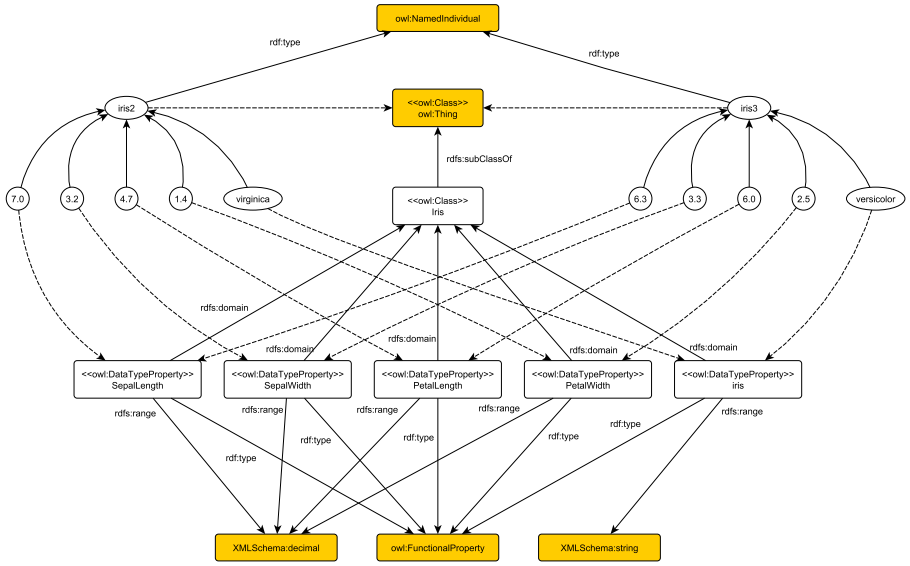

Fig. 3. Knowledge graph corresponding to the Iris data set obtained with PSyKE.

The rules extracted during the experiment are the following: *(i)* class is "setosa" if the petal length is less than or equal 2.75 cm; *(ii)* otherwise, class is "virginica" if the petal width is greater than 1.68 cm; *(iii)* otherwise, class is "versicolor".

The same concepts can be formalised as SWRL rules and appended to the previous ontology:

```
Iris(?iris), SepalLength(?iris, ?sepalLength),
SepalWidth(?iris, ?sepalWidth), PetalLegth(?iris, ?petalLength),
PetalWidth(?iris, ?petalWidth), lessThanOrEqual(?petalLength, 2.75)
-> iris(?iris, "setosa")

Iris(?iris), SepalLength(?iris, ?sepalLength),
SepalWidth(?iris, ?sepalWidth), PetalLegth(?iris, ?petalLength),
PetalWidth(?iris, ?petalWidth), greaterThan(?petalLength, 2.75),
greaterThan(?petalWidth, 1.68) -> iris(?iris, "virginica")

Iris(?iris), SepalLength(?iris, ?sepalLength),
SepalWidth(?iris, ?sepalWidth), PetalLegth(?iris, ?petalLength),
PetalWidth(?iris, ?petalWidth), greaterThan(?petalLength, 2.75),
lessThanOrEqual(?petalWidth, 1.68) -> iris(?iris, "versicolor")
```

5 Open Issues

Even though the new design of the PSyKE platform proved to be efficient and reliable in the presented case study, during our work we highlighted several critic situations that require further investigations.

The first issue is related to the semantics of output SWRL rules w.r.t. the classical PSyKE Prolog rules. Prolog rules, for their nature, are always non-overlapping, since they are evaluated *in order* and, thus, only one rule can be applied at a time. Conversely, the SWRL rules belonging to an ontology are not ordered, so a given individual can match more than one rule contemporaneously. This may lead to the detection of inconsistencies during predictions performed via PSyKE, since some SKE algorithms supported by the platform produce output lists with possibly overlapping rules. In the case of classification this implies that an individual, according to the ontology rules, can belong to more than one different class, resulting in an inconsistency. Since the described issue comes along with a positive potential, we plan to exploit the ontology capability of highlighting inconsistencies to enable a better inspection/debugging of the BB behaviour as well as of the extraction algorithms' implementations, providing to developers exact motivations for BB misclassifications and the precise boundaries of overlapping rules.

The second issue is related to the representation of continuous output values in SWRL rules. Due to their reduced expressiveness w.r.t. Prolog language, they do not allow a linear combination of input variables to be used in the consequent part of rules. This means that only constant values can be associated to rule outputs. While this is not a problem in classification tasks, where the output predictions should be exactly a constant value representing the class labels of classified individuals, the inability to handle continuous output limits the application of some SKE algorithms devoted to regression tasks. However, since many SKE procedures supported by PSyKE that are designed for regression introduce – due to their design – a discretisation of continuous outputs, SWRL rules can be produced in the majority of cases.

Finally, inconsistencies may arise after the addition of the extracted knowledge in the form of SWRL rules to the ontology containing the input data. This is caused – besides by conflicts in overlapping extracted rules – by *(i)* wrong class predictions of the underlying BB classifier, or *(ii)* discretised or approximated output values in regression rules. In the first case, the underlying model gives for some individuals a wrong prediction. An extraction procedure is then applied to the model, resulting in the production of rules having misclassification issues following those of the model itself. When added to the input ontology these rules are inconsistent, since lead to a (partial) wrong classification of the training set individuals, having known class label encoded in the ontology. In the second case, a similar reasoning holds, since the inconsistency is a mismatch between continuous output values given in input as training individuals and the approximated/discretised rule output values. This issue can be overcome by removing the output variable values from the ontology and by relying only on the extracted SWRL rules to obtain the *predicted* output values instead.

6 Conclusions

In this paper we present an extension of the design of the PSyKE platform aimed at combining SW tool and SKE from black-box predictors. PSyKE is extended with the capability of managing knowledge graphs and ontologies other than tabular data as inputs, since these are the most common formats shared in the SW. As for the output knowledge, the new version of PSyKE can provide SWRL rules included into an ontology, thus enabling automatic reasoning and knowledge consistency checks. In addition, several utilities to relationalise and propositionalise data encoded in various formats are added. Thanks to the presented extension, the knowledge extraction workflow of PSyKE is generalised, since it can now begin and terminate in the semantic domain, without being bounded to specific input data and output rule formats.

Notably, our contributions – in particular, w.r.t. the extraction of *semantic* knowledge out of ML models – promotes interoperability (based on extracted KG) between heterogeneous intelligent agents leveraging upon sub-symbolic AI—provided, of course, that they adopt PSyKE for knowledge extraction. Furthermore, despite our contribution is tailored on PSyKE – at least, at the technological level –, we argue that this paper can also be read as guide describing how to extract semantic knowledge out of ML predictors *in the general case*. Hence, in our future works, we plan to describe the extraction of semantic knowledge in a general, technology-agnostic way.

Even though our platform provides the expected results for classification tasks, further investigations should be carried out regarding problems having regressive nature, since the SWRL rules provided in output are not suited to represent linear combinations of variables. Our future works will be focused also on addressing consistency issues after the extraction of overlapping rules. Finally, we plan to perform more complete tests, especially on complex real-world data sets and involving end-users.

Acknowledgments

This paper is partially supported by the CHIST-ERA IV project CHIST-ERA-19-XAI-005, co-funded by EU and the Italian MUR (Ministry for University and Research).

References

1. Andrews, R., Diederich, J., Tickle, A.B.: Survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge-Based Systems **8**(6), 373–389 (1995). https://doi.org//10.1016/0950-7051(96)81920-4
2. Azcarraga, A., Liu, M.D., Setiono, R.: Keyword extraction using backpropagation neural networks and rule extraction. In: The 2012 International Joint Conference on Neural Networks (IJCNN 2012). pp. 1–7. IEEE (2012). https://doi.org/10.1109/IJCNN.2012.6252618

3. Baesens, B., Setiono, R., De Lille, V., Viaene, S., Vanthienen, J.: Building credit-risk evaluation expert systems using neural network rule extraction and decision tables. In: Storey, V.C., Sarkar, S., DeGross, J.I. (eds.) ICIS 2001 Proceedings. pp. 159–168. Association for Information Systems (2001), http://aisel.aisnet.org/icis2001/20
4. Baesens, B., Setiono, R., Mues, C., Vanthienen, J.: Using neural network rule extraction and decision tables for credit-risk evaluation. Management Science **49**(3), 312–329 (2003). https://doi.org/10.1287/mnsc.49.3.312.12739
5. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American **284**(5), 34–43 (2001), https://www.scientificamerican.com/article/the-semantic-web/
6. Bologna, G., Pellegrini, C.: Three medical examples in neural network rule extraction. Physica Medica **13**, 183–187 (1997), https://archive-ouverte.unige.ch/unige:121360
7. Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A.: Classification and Regression Trees. CRC Press (1984)
8. Ciatto, G., Calegari, R., Omicini, A., Calvaresi, D.: Towards XMAS: eXplainability through Multi-Agent Systems. In: Savaglio, C., Fortino, G., Ciatto, G., Omicini, A. (eds.) AI&IoT 2019 – Artificial Intelligence and Internet of Things 2019, CEUR Workshop Proceedings, vol. 2502, pp. 40–53. Sun SITE Central Europe, RWTH Aachen University (Nov 2019), http://ceur-ws.org/Vol-2502/paper3.pdf
9. Ciatto, G., Schumacher, M.I., Omicini, A., Calvaresi, D.: Agent-based explanations in AI: Towards an abstract framework. In: Calvaresi, D., Najjar, A., Winikoff, M., Främling, K. (eds.) Explainable, Transparent Autonomous Agents and Multi-Agent Systems, LNCS, vol. 12175, pp. 3–20. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51924-7_1
10. Craven, M.W., Shavlik, J.W.: Using sampling and queries to extract rules from trained neural networks. In: Machine Learning Proceedings 1994, pp. 37–45. Elsevier (1994). https://doi.org/10.1016/B978-1-55860-335-6.50013-1
11. Craven, M.W., Shavlik, J.W.: Extracting tree-structured representations of trained networks. In: Touretzky, D.S., Mozer, M.C., Hasselmo, M.E. (eds.) Advances in Neural Information Processing Systems 8. Proceedings of the 1995 Conference, pp. 24–30. The MIT Press (Jun 1996), http://papers.nips.cc/paper/1152-extracting-tree-structured-representations-of-trained-networks.pdf
12. d'Amato, C.: Machine learning for the semantic web: Lessons learnt and next research directions. Semantic Web **11**(1), 195–203 (2020). https://doi.org/10.3233/SW-200388
13. Franco, L., Subirats, J.L., Molina, I., Alba, E., Jerez, J.M.: Early breast cancer prognosis prediction and rule extraction using a new constructive neural network algorithm. In: Computational and Ambient Intelligence (IWANN 2007). LNCS, vol. 4507, pp. 1004–1011. Springer (2007). https://doi.org/0.1007/978-3-540-73007-1_121
14. Freitas, A.A.: Comprehensible classification models: a position paper. ACM SIGKDD Explorations Newsletter **15**(1), 1–10 (Jun 2014). https://doi.org/10.1145/2594473.2594475
15. Glimm, B., Horrocks, I., Motik, B., Stoilos, G., Wang, Z.: HermiT: An OWL 2 reasoner. Journal of Automated Reasoning **53**(3), 245–269 (2014). https://doi.org/10.1007/s10817-014-9305-1
16. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Computing Surveys **51**(5), 1–42 (2018). https://doi.org/10.1145/3236009

17. Gunning, D., Stefik, M., Choi, J., Miller, T., Stumpf, S., Yang, G.: XAI - explainable artificial intelligence. Science Robotics **4**(37) (2019). https://doi.org/10.1126/scirobotics.aay7120
18. Hayashi, Y., Setiono, R., Yoshida, K.: A comparison between two neural network rule extraction techniques for the diagnosis of hepatobiliary disorders. Artificial intelligence in Medicine **20**(3), 205–216 (2000). https://doi.org/10.1016/s0933-3657(00)00064-6
19. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S.: OWL 2 Web Ontology Language Primer (Second Edition). W3C Recommendation 11 December 2012 (2012), https://www.w3.org/TR/owl2-primer
20. Hoekstra, R.: The knowledge reengineering bottleneck. Semantic Web **1**(1-2), 111–115 (2010). https://doi.org/10.3233/SW-2010-0004
21. Hofmann, A., Schmitz, C., Sick, B.: Rule extraction from neural networks for intrusion detection in computer networks. In: 2003 IEEE International Conference on Systems, Man and Cybernetics. vol. 2, pp. 1259–1265. IEEE (2003). https://doi.org/10.1109/ICSMC.2003.1244584
22. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission 21 May 2004 (2004), https://www.w3.org/Submission/SWRL
23. Huysmans, J., Baesens, B., Vanthienen, J.: ITER: An algorithm for predictive regression rule extraction. In: Data Warehousing and Knowledge Discovery (DaWaK 2006). pp. 270–279. Springer (2006). https://doi.org/10.1007/11823728_26
24. Huysmans, J., Dejaeger, K., Mues, C., Vanthienen, J., Baesens, B.: An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. Decision Support Systems **51**(1), 141–154 (2011). https://doi.org/10.1016/j.dss.2010.12.003
25. Lachiche, N.: Propositionalization. In: Sammut, C., Webb, G.I. (eds.) Encyclopedia of Machine Learning, pp. 812–817. Springer US, Boston, MA (2010). https://doi.org/10.1007/978-0-387-30164-8_680
26. Lamy, J.: Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. Artificial Intelligence In Medicine **80**, 11–28 (2017). https://doi.org/10.1016/j.artmed.2017.07.002
27. Lipton, Z.C.: The mythos of model interpretability. Queue **16**(3), 31–57 (Jun 2018). https://doi.org/10.1145/3236386.3241340
28. Maamar, Z., Moulin, B.: Interoperability of distributed and heterogeneous systems based on software agent-oriented frameworks. In: Kandzia, P., Klusch, M. (eds.) Cooperative Information Agents, First International Workshop, CIA' 97, Kiel, Germany, February 26-28, 1997, Proceedings. Lecture Notes in Computer Science, vol. 1202, pp. 248–259. Springer (1997). https://doi.org/10.1007/3-540-62591-7_38
29. Manola, F., Miller, E., McBride, B.: Resource Description Framework (RDF) Primer. W3C Recommendation 10 February 2004 (2004), https://www.w3.org/TR/rdf-primer
30. Motik, B., Shearer, R.D.C., Horrocks, I.: Hypertableau reasoning for description logics. Journal of Artificial Intelligence Research **36**, 165–228 (2009). https://doi.org/10.1613/jair.2811
31. Murphy, P.M., Pazzani, M.J.: ID2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees. In: Machine Learning Proceedings 1991, pp. 183–187. Elsevier (Jun 1991). https://doi.org/10.1016/B978-1-55860-200-7.50040-4, 8th International Conference (ML 1991), Evanston, IL, USA
32. Quinlan, J.R.: Simplifying decision trees. International Journal of Man-Machine Studies **27**(3), 221–234 (1987). https://doi.org/10.1016/S0020-7373(87)80053-6

33. Quinlan, J.R.: C4.5: Programming for machine learning. Morgan Kauffmann (1993), https://dl.acm.org/doi/10.5555/152181
34. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence **1**(5), 206–215 (2019). https://doi.org/10.1038/s42256-019-0048-x
35. Sabbatini, F., Ciatto, G., Calegari, R., Omicini, A.: On the design of PSyKE: A platform for symbolic knowledge extraction. In: Calegari, R., Ciatto, G., Denti, E., Omicini, A., Sartor, G. (eds.) WOA 2021 – 22nd Workshop "From Objects to Agents". CEUR Workshop Proceedings, vol. 2963, pp. 29–48. Sun SITE Central Europe, RWTH Aachen University (Oct 2021), http://ceur-ws.org/Vol-2963/paper14.pdf, 22nd Workshop "From Objects to Agents" (WOA 2021), Bologna, Italy, 1–3 Sep. 2021. Proceedings
36. Sabbatini, F., Ciatto, G., Omicini, A.: GridEx: An algorithm for knowledge extraction from black-box regressors. In: Calvaresi, D., Najjar, A., Winikoff, M., Främling, K. (eds.) Explainable and Transparent AI and Multi-Agent Systems. Third International Workshop, EXTRAAMAS 2021, Virtual Event, May 3–7, 2021, Revised Selected Papers, LNCS, vol. 12688, pp. 18–38. Springer Nature, Basel, Switzerland (2021). https://doi.org/10.1007/978-3-030-82017-6_2
37. Saleem, A., Honeth, N., Nordström, L.: A case study of multi-agent interoperability in IEC 61850 environments. In: IEEE PES Conference on Innovative Smart Grid Technologies, ISGT Europe 2010, October 11-13, 2010, Gothenburg, Sweden. pp. 1–8. IEEE (2010). https://doi.org/10.1109/ISGTEUROPE.2010.5638876
38. Setiono, R., Baesens, B., Mues, C.: Rule extraction from minimal neural networks for credit card screening. International Journal of Neural Systems **21**(04), 265–276 (2011). https://doi.org/10.1142/S0129065711002821
39. Shearer, R.D.C., Motik, B., Horrocks, I.: HermiT: A highly-efficient OWL reasoner. In: Dolbear, C., Ruttenberg, A., Sattler, U. (eds.) Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions, collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26-27, 2008. CEUR Workshop Proceedings, vol. 432. CEUR-WS.org (2008), http://ceur-ws.org/Vol-432/owled2008eu_submission_12.pdf
40. Siorpaes, K., Hepp, M.: OntoGame: Towards overcoming the incentive bottleneck in ontology building. In: Meersman, R., Tari, Z., Herrero, P. (eds.) On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops, OTM Confederated International Workshops and Posters, AWeSOMe, CAMS, OTM Academy Doctoral Consortium, MONET, OnToContent, ORM, PerSys, PPN, RDDS, SSWS, and SWWS 2007, Vilamoura, Portugal, November 25-30, 2007, Proceedings, Part II. Lecture Notes in Computer Science, vol. 4806, pp. 1222–1232. Springer (2007). https://doi.org/10.1007/978-3-540-76890-6_50
41. Sirin, E., Parsia, B.: Pellet: An OWL DL reasoner. In: Haarslev, V., Möller, R. (eds.) Proceedings of the 2004 International Workshop on Description Logics (DL2004), Whistler, British Columbia, Canada, June 6-8, 2004. CEUR Workshop Proceedings, vol. 104. CEUR-WS.org (2004), http://ceur-ws.org/Vol-104/30Sirin-Parsia.pdf
42. Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Journal of Web Semantics **5**(2), 51–53 (2007). https://doi.org/10.1016/j.websem.2007.03.004
43. Steiner, M.T.A., Steiner Neto, P.J., Soma, N.Y., Shimizu, T., Nievola, J.C.: Using neural network rule extraction for credit-risk evaluation. International Journal of

Computer Science and Network Security **6**(5A), 6–16 (2006), http://paper.ijcsns.org/07_book/200605/200605A02.pdf