Data Min Knowl Disc (2014) 28:1222–1265
DOI 10.1007/s10618-014-0363-0

Ontology of core data mining entities

Panče Panov · Larisa Soldatova · Sašo Džeroski

Received: 3 March 2013 / Accepted: 7 June 2014 / Published online: 5 July 2014

Abstract In this article, we present OntoDM-core, an ontology of core data mining entities. OntoDM-core defines the most essential data mining entities in a three-layered ontological structure comprising of a specification, an implementation and an application layer. It provides a representational framework for the description of mining structured data, and in addition provides taxonomies of datasets, data mining tasks, generalizations, data mining algorithms and constraints, based on the type of data. OntoDM-core is designed to support a wide range of applications/use cases, such as semantic annotation of data mining algorithms, datasets and results; annotation of QSAR studies in the context of drug discovery investigations; and disambiguation of terms in text mining. The ontology has been thoroughly assessed following the practices in ontology engineering, is fully interoperable with many domain resources and is easy to extend. OntoDM-core is available at http://www.ontodm.com.

Responsible editor: Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, Filip Železný

P. Panov (✉) · S. Džeroski
Department of Knowledge Technologies, Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
e-mail: pance.panov@ijs.si

L. Soldatova
Department of Computer Science, Brunel University, Uxbridge, London UB8 3PH, UK
e-mail: larisa.soldatova@brunel.ac.uk

S. Džeroski
Jožef Stefan International Postgraduate School, Jamova cesta 39, 1000 Ljubljana, Slovenia

S. Džeroski
Centre of Excellence for Integrated Approaches in Chemistry and Biology of Proteins, Jamova cesta 39, 1000 Ljubljana, Slovenia
e-mail: saso.dzeroski@ijs.si

Keywords Ontology of data mining · Mining structured data · Domain ontology

1 Introduction

Intelligent information systems rely on advanced representations of application domains, tasks, methods to solve those tasks, and the available data. Ontologies as data and knowledge models offer logically defined, flexible and interoperable representations of principal domain entities to empower information systems. Ontology—the "science of being"—typically has different meanings in different contexts. In the late twentieth Century, Artificial Intelligence (AI) adopted the term in the sense of a "specification of a conceptualization"—an ontology defines a set of representational primitives to model a domain of knowledge. The primitives are typically classes (or sets), attributes (or properties), and relations among class members (Gruber 2009).

The recent success of ontologies has been driven by the popularity of semantic web technologies. Ontologies are used to specify controlled vocabularies, which can be used to exchange data among different systems, provide services for answering queries, publish reusable knowledge bases, and offer services to facilitate interoperability across multiple, heterogeneous systems and databases (Gruber 2009). The Open Bio-Ontologies (OBO)[1] are a leading effort in ontology engineering. As biological sciences generate big and complex data, the development of bio-ontologies has been critical in handling these data and enabling interoperability between databases and between applications (Robinson and Bauer 2011).

Biomedical computing has become critically dependent on the use of ontologies. Resources such as the Gene Ontology,[2] the National Cancer Institute's Thesaurus,[3] the Foundational Model of Anatomy[4] (FMA), SNOMED-CT[5], and the Ontology for Biomedical Investigations[6] (OBI) have become integral components of modern biomedical research and practice. In the past, ontologies were perceived as arcane, overcomplicated, and perhaps over-hyped. Now they serve as essential infrastructure for contemporary biology and medicine (Soldatova et al. 2010). Currently, an open repository of biomedical ontologies BioPortal[7] contains over 350 ontologies. Ontologies are used to annotate experimental data, to assist information retrieval, to enable integration of heterogeneous data, to drive literature mining, and to build knowledge bases (Soldatova et al. 2010).

The OBO Foundry is coordinating the efforts on establishing a set of principles for ontology development in order to create a suite of orthogonal interoperable reference ontologies in the biomedical domain for supporting data and knowledge sharing and to avoid duplication of efforts. To ensure interoperability of bio-medical ontologies

[1] OBO: http://www.obofoundry.org (accessed 1 June 2014).

[2] GO: http://www.geneontology.org (accessed 1 June 2014).

[3] NCI Thesaurus: http://ncit.nci.nih.gov (accessed 1 June 2014).

[4] FMA: http://sig.biostr.washington.edu/projects/fm(accessed 1 June 2014).

[5] SNOMED-CT: http://www.ihtsdo.org/snomed-ct (accessed 1 June 2014).

[6] OBI: http://www.obi-ontology.org (accessed 1 June 2014).

[7] BioPortal: http://bioportal.bioontology.org (accessed 1 June 2014).

the OBO Foundry recommends using the Basic Formal Ontology[8] (BFO) as an upper-level ontology and relations defined by the Relation Ontology[9] (RO) to guide the development of application ontologies.

Unfortunately, ontology research in the domain of Data Mining (DM) and Machine Learning is far less advanced. There are a dozen of relevant ontologies (see Sect. 2), but their development has been mostly independent with little coordination of the efforts. In 2011, there has been an attempt to establish a Data Mining Ontologies (DMO) Consortium. The developers of the Data Mining Optimization (DMOP) ontology (Hilario et al. 2011), the Exposé ontology (Vanschoren et al. 2012), the Ontology of Data Mining (OntoDM) (Panov et al. 2008, 2010) and others have agreed that a DM Core resource, which provides formal standardized definitions for the principal DM entities, e.g., *algorithm, data, dataset, task, implementation* would be of benefit to the whole DM research community. Since then, however, there has been little progress in the development of a community agreed standard in this area.

Data mining is concerned with analyzing different types of data. Besides data in the format of a single table (with primitive datatypes as attributes), most commonly used in data mining, structured (complex) data are receiving an increasing amount of interest (Bakir et al. 2007). These include graphs, sequences, networks, texts, images, multimedia and relational data. Many data mining algorithms are designed to solve data mining tasks for specific types of data, most frequently defined for data represented in a 'single table' (e.g., classification, regression). In essence, these can be defined on an arbitrary datatype (Džeroski 2007).

One of the major challenges in data mining is to treat and represent the mining of different types of structured data in a uniform fashion (Yang and Wu 2006). In addition, the mining of complex data, the use of domain knowledge, and the support for complex knowledge discovery processes have been identified as the top-most open issues in DM that have the best chance of providing the tools for building integrated AI systems (Kriegel et al. 2007; Dietterich et al. 2008).

In this paper, we report on the OntoDM-core ontology as a step towards the development of a data and knowledge exchange standard for the area of data mining. The paper goes far beyond our earlier work on OntoDM (Panov et al. 2008, 2010). The preliminary versions of the ontology reported in the conference paper and in the book chapter contain 165 and 280 classes accordingly, while the OntoDM-core ontology described in this paper has 856 classes. The version presented here is not a mere extension of the earlier work, but a substantial re-design of the initial representation.

OntoDM-core introduces taxonomies for the key DM entities, based on the type of data (e.g., taxonomy of datasets, data mining tasks, and data mining algorithms). The ontology includes a definition of constraints, a taxonomy of constraints, constraint-based data mining (CBDM) tasks, scenarios and workflows. We populated a segment from OntoDM-core with instances, performed reasoning, and queried the ontology. The ontology has been evaluated following best practices in ontology engineering and we show three use-cases: the use of OntoDM-core classes as a mid-level ontology

[8] BFO: http://www.ifomis.org/bfo (accessed 1 June 2014).

[9] RO: http://obofoundry.org/ro (accessed 1 June 2014).

for the Exposé ontology, annotation of Quantitative Structure-Activity Relationship (QSAR) studies, and disambiguation of terms in text mining.

The paper is organized as follows. In Sect. 2, we present a critical overview of related work. In Sect. 3, we present the design and implementation, while in Sect. 4, we review the key classes of OntoDM-core. Furthermore, in Sect. 5, we describe the assessment of the ontology. In Sect. 6, we show examples of queries and reasoning over a populated segment of OntoDM-core, and in Sect. 7, we present three cases of using the ontology. Next, in Sect. 8, we discuss the lessons learned while performing this research and present a mapping of DM core terms from different ontologies. Finally, in Sect. 9, we give conclusions and point out directions for further work.

2 Related work and motivation

There are several projects relevant to the ontological modeling of DM entities. The main developments in formalized representations of data mining entities take place in the areas of DM dealing with description of DM services and resources on the GRID, DM workflow construction, representation of machine learning experiments in the context of experiment databases, and meta-learning. We first provide a brief overview of the most relevant projects and then analyze their limitations. Finally, we elaborate the need for OntoDM-core and its distinction from the related representations.

2.1 Related ontologies in the domain of data mining

In the context of GRID programming, Cannataro and Comito (2003) proposed an ontology of data mining, named DAMON, in order to allow the semantic search of DM software and other DM resources and to assist the user by suggesting the software to use on the basis of the user's requirements and needs. Brezany et al. (2007) introduced an ontology-based framework for the automated construction of complex interactive data mining workflows as a means of improving the productivity of GRID-enabled data systems. They developed an ontology which is based on concepts from industry standards (e.g., PMML language,[10] CRISP-DM (Chapman et al. 1999)).

Bernstein et al. (2005) proposed a prototype of an Intelligent Discovery Assistant (IDA), that provides users with systematic enumerations of valid sequences of DM operators. IDA uses an ontology of data mining operators (algorithm implementations) divided into three main classes (preprocessing operators, induction algorithms, and post processing operators). Žáková et al. (2010) proposed an ontology (with descriptions of propositional algorithms and algorithms for relational mining) directly integrated with a planner in order to automatically propose knowledge discovery workflows. The methodology was implemented in the Orange4WS Environment for service-oriented data mining (Podpečan et al. 2011).

Diamantini and Potena (2008) introduced a semantic-based, service-oriented framework for tools sharing and reuse, to support semantic enrichment and deployment of

[10] PMML: http://www.dmg.org/ (accessed 1 June 2014).

tools as web services. For that purpose, they propose an ontology named KDDONTO. Kietz et al. (2010) presented a DM ontology for workflow planning, designed to contain all the information necessary to support a 3rd generation KDD Support System. Vanschoren et al. (2012) proposed an ontology for machine learning experiments, named Exposé (see Sect. 7.1).

Hilario et al. (2011) proposed the Data Mining Optimization (DMOP) Ontology with its main goal to support meta-mining (more commonly known as meta-learning or meta-analysis) of data mining experiments in order to extract workflow patterns. The ontology contains representations of DM tasks, algorithms, models, workflows and experiments, limited to the case of propositional data mining. In addition, the authors have developed a knowledge base by populating the ontology with instances. The DMOP ontology version 5.3 (Keet et al. 2013) started a preliminary alignment of classes and relations with the DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering)[11] upper-level ontology (Gangemi et al. 2002).

Along with OntoDM-core, reported in this article, we are developing two closely related ontologies: OntoDT and OntoDM-KDD. OntoDT (Panov 2012) is an ontology for datatypes based on the ISO standard for datatypes (ISO 2007). It is currently the only ontology that can successfully support the characterization of structured datatypes, and is imported into OntoDM-core. OntoDM-KDD (Panov et al. 2013) is an ontology for representing the knowledge discovery process based on the CRISP-DM process model (Chapman et al. 1999). Due to the adherence to the same design principles, we have seamlessly integrated OntoDT, OntoDM-KDD, and OntoDM-core into the OntoDM ontology.

2.2 Critical comparison of current data mining ontologies

Most of the current ontologies in the domain of data mining are application ontologies. They are engineered for a specific use or application focus and their scope is specified through testable use cases (Malone and Parkinson 2010). In addition some of the ontologies were successfully used as are backbones of intelligent discovery assistant systems (Serban et al. 2013). For example, the DAMON ontology provides semantic search of DM software resources on the GRID, KDDOnto supports semantic enrichment and deployment of tools as web services, the DMOP ontology provides descriptions of data and algorithms for meta-mining of data mining experiments in order to extract workflow patterns, and Exposé is a base for the database schema of the machine learning experiments database.

In the absence of any DM-specific ontology development guidelines, the developers of DM ontologies have followed different design principles. They have used different upper-level ontologies, hierarchies and relations, and therefore it is now hard to integrate them to support new applications.

For example, in this paper we describe the annotation of QSAR studies used in drug design (see Sect. 7.2). We propose a simple annotation scheme that imports a number of terms from several ontologies. An integration of several ontologies, including at least

[11] DOLCE: http://www.loa.istc.cnr.it/old/DOLCE.html (accessed 1 June 2014).

two ontologies defining DM entities, is required for more sophisticated modeling of the information about QSAR studies. However, we show that not only the integration, but even the mapping of ontologies is an issue.

Consider the DMOP ontology mentioned above and the Information Artifact Ontology[12] (IAO), which is widely used by biomedical ontologies and its applications, including those relevant to QSAR studies. It is currently hard to determine if the class *DMOP: DataSet*, defined as the subclass of entities that have spatial and temporal properties and is disjoint with any abstract entities, has the same semantic meaning as the class *IAO: data set*, defined as an information content entity (an entity that stays in the aboutness relation with some entity in the real world). Therefore an import of DMOP terms defining properties of datasets could cause problems with the description of properties of QSAR datasets.

Automated mapping techniques would map these two classes by their highly similar labels (*DataSet* and *data set*), but they would also identify that their parent classes and underlying axioms, and therefore their semantics differ. Careful manual inspection is required to ensure the logical consistency and correctness of the integrated representations. The process of re-using formally defined entities, relevant to the applications, from other resources is more efficient if those resources are developed following the same design principles, upper-level classes and relations.

Regarding domain coverage, current DM ontologies deal mostly with the representation of mining propositional, 'single table', data with primitive attributes as datatypes and focus on predictive data mining tasks, such as classification and regression. An exception is the KD Ontology (Žáková et al. 2010) that includes some very specific relational mining tasks and algorithms. In this respect, we would like OntoDM-core to provide a vocabulary and schema that would enable flexible extension and generalization towards representing data mining tasks and algorithms for mining structured data. For example, the structure of the ontology should allow easy representation of data mining tasks (e.g., hierarchical classification) and algorithms (e.g., hierarchical classification algorithm) that are defined on arbitrarily complex types of data (e.g., directed acyclic graph of class labels). In addition, in contrast to the existing DM ontologies, we would like OntoDM-core to provide a representation of constraint-based data mining tasks and algorithms.

2.3 Why do we need OntoDM-core?

Ontology development is an expensive and time-consuming process. Most ontologies relevant to DM have been developed for specific tasks. It is likely that a new task would require a re-design and/or an integration of existing representations, or even the development of a new one. With this in mind, OntoDM-core aims to be a reference domain ontology for DM. We attempt to develop a representation of key DM entities that is as generic as possible, following the proposal for a general framework for DM by Džeroski (2007). In this way, the ontology can be easily extended to allow

[12] IAO: http://code.google.com/p/information-artifact-ontology (accessed 1 June 2014).

the representation of new datatypes, data mining tasks, data mining algorithms, and generalizations and thus support a wide range of possible applications.

By developing the OntoDM-core ontology, we are addressing the challenge of representing the mining of structured data (datasets, tasks, algorithms, generalizations) in a generic fashion, treating the traditional single table data mining as a special case. Furthermore, we address the task of representing constraints in DM and CBDM tasks and algorithms. To address this goal, we propose a simple structure of key core entities from the domain, taking into account the types of the data at all levels (see Sect. 4). This allows us to define taxonomies of datasets, DM tasks, generalizations, and algorithms that are dependent of the datatype. These are unique features that have not been tackled in any related DM ontology. In this context, OntoDM-core could be used to extend the experiment databases framework (Vanschoren et al. 2012) to represent machine learning experiments with algorithms for mining structured data (see Sect. 7.1).

Many of the OBO Foundry recommended design principles are appropriate for the development of DM ontologies and also represent the state-of-the-art and best practices in ontology engineering. Consequently, we have chosen to follow the OBO Foundry recommendations in the development of OntoDM-core (see Sect. 3). This includes the use of an upper-level ontology as a template, the use of a community agreed set of relations based on the upper-level classes, and the massive reuse of classes from other ontologies (avoiding the duplication of efforts). Therefore, OntoDM-core is interoperable with other state-of-the-art ontologies. This aspect can be used in cross-domain applications that need the representation of data mining entities in combination with other domain ontologies built on same design principles. For example, by using OntoDM-core in combination with a domain ontology for drug design investigations we would be able to represent data mining algorithms that operate on structured QSAR data in the process of annotation of QSAR studies (see Sect. 7.2).

3 Design and implementation

In this section, we present the design and implementation of OntoDM-core. First, we summarize the state-of-the-art and terminology of ontological engineering (see Table 1). Next, we present the design methodology used for the design of OntoDM-core. Finally, we provide an overview of the proposed three layered ontology structure.

3.1 Design methodology

OntoDM-core is expressed in OWL-DL,[13] a de facto standard for representing ontologies (see Table 2). The ontology is being developed using the Protégé[14] ontology editor. The ontology is freely available at http://www.ontodm.com and at BioPortal.

In order to ensure the extensibility and interoperability of OntoDM-core with other resources, in particular with biomedical applications, the OntoDM-core ontology follows the Open Bio-Ontologies (OBO) Foundry design principles,[15] such as the use of

[13] OWL-DL: http://www.w3.org/TR/owl-guide (accessed 1 June 2014).

[14] Protégé: http://protege.stanford.edu (accessed 1 June 2014).

[15] OBO Foundry principles: http://obofoundry.org/crit.shtml (accessed 1 June 2014).

Table 1 Ontological Engineering in a Nutshell

1. Ontology components

   A typical ontology consists of: *classes* (e.g., bank, bank account), *instances* (e.g., HSBC bank, Mr Smith's account), *relations* (e.g., *is-a*, *instance-of*, *participates-in*,*has-part*), and *axioms* (e.g., church is disjoint with government). Classes and instances form a hierarchy structured by the *is-a* and *instance-of* relations. Subclasses and instances inhere all properties of upper classes in the hierarchy.
2. TBox and ABox

   TBox contains the axioms defining the classes and relations in an ontology. *ABox* contains the assertions about the instances in the domain.
3. Ontology architecture

   Upper-level ontology is a high-level, domain independent ontology, providing a framework by which different systems can use a common knowledge base and from which more domain-specific ontologies can be derived. *Mid-level ontology* serves as a bridge between general entities defined in the upper-level ontology and the low-level domain ontology. *Domain ontology* specifies entities particular to a domain of interest and represents these entities and their relationships from a domain specific perspective.
4. Languages

   Ontologies have many usages, including the definition of a semantic layer in the Semantic Web. Semantic Web standard languages are listed below: *RDF (Resource Description Framework)* is a standard model for data interchange on the Web. An RDF database (a triple store) contains RDF triples in the form object-predicate-subject, where each element has an URI. Example: HSBC bank instance-of bank, where a corresponding ontology may define each entity and supply URIs. *RDF-S (Schema)* provides basic elements for describing ontologies. *OWL (Web Ontology Language)* is a language for modeling ontologies. The most popular variants are: OWL-DL (Description Logic) and OWL2 with enhanced functionality. *SPARQL* is the query language for the Semantic Web[a]. *SPARQL-DL* is variant of SPARQL that works with OWL-DL ontologies[b].
5. Ontology design

   There are no universally accepted design principles for the development of ontologies. There are best practices. The most popular ontology engineering methodologies include the methodology by Uschold and King (1995), the TOVE methodology (Fox and Grüninger 1994), METHONTOLOGY. (López et al. 1999), the On-To-Knowledge methodology (Sure et al. 2009), the NeOn methodology (Suarez-Figueroa et al. 2012) and others. The-state-of-the-art in ontology engineering in the biomedical domains is the set of recommendations by OBO (Open Biomedical Ontologies) Foundry. There are 19 OBO Foundry recommendations that include: the use of an upper-level ontology as a prototype to guide the development of an application ontology, the use of standardized relations, the avoidance of multiple inheritance, and the development of orthogonal resources (a class can be defined only in one ontology).
6. Ontology mapping

   There is no universal ontology that defines all principal entities. Computer applications often have to integrate several ontologies. Different ontologies may define the same entities, but with different URIs (e.g.,foaf:Agent and dtc:agent). Explicit mapping between such terms could be done automatically. The same entities may have different labels, or different entities may have the same labels. Sometimes complicated rules are required to reconcile differences in the intended semantics/meaning. Such mappings still cannot be done automatically. It is easier to integrate or construct a mapping for ontologies that use the same set of relations and upper-level classes.

[a] SPARQL: http://www.w3.org/TR/rdf-sparql-query/ (accessed 1 June 2014)
[b] SPARQL-DL: http://www.w3.org/2001/sw/wiki/SPARQL-DL (accessed 1 June 2014)

an upper-level ontology, the use of formal ontology relations, single inheritance, and the re-use of already existing ontological resources where possible (Smith et al. 2007). The application of these design principles enables cross-domain reasoning, facilitates wide re-usability of the developed ontology, and avoids duplication of ontology development efforts. Consequently, OntoDM-core imports the upper-level classes from the

Table 2 External ontologies that have been re-used in the development of OntoDM-core

Label	Ontology	Description
BFO	Basic formal ontology	An upper-level ontology that supports information retrieval, analysis and integration in scientific and other domains. It does not contain physical, chemical, biological or other terms which would properly fall within the coverage domains of the special sciences. Instead, it defines such principal entities as continuant and occurent. BFO is used by more than 100 ontologies (Grenon et al. 2004). URL: http://www.ifomis.org/bfo
RO	OBO relational ontology	A collection of relations for standardization across ontologies in the OBO Foundry. It incorporates core upper-level relations, such as part-of, as well as biology-specific relationships (Smith et al. 2005). URL: http://purl.org/obo/owl/OBO_REL
OBI	Ontology of biomedical investigations	An integrated ontology for the description of life-science and clinical investigations that defines such principal entities as investigation, assay, and data transformation (Brinkman et al. 2010). URL: http://purl.obolibrary.org/obo/obi
IAO	Information artifact ontology	An ontology that defines information entities, e.g. file, document, and relations between them, e.g. is-about. URL: http://goo.gl/4bSZr2
SWO	Software ontology	A resource for describing software tools, their types, tasks, versions, provenance and data associated. URL: http://theswo.sourceforge.net
OntoDT	Ontology of datatypes	The OntoDT ontology is based on the ISO standard (ISO 2007) for representing datatypes, datatype qualities, operations on datatypes, and a taxonomy of datatypes. The OntoDT taxonomy of datatypes consists of primitive datatypes, generated datatypes, subtypes, and defined datatypes. The data used for data mining can have arbitrarily complex structure, and OntoDT is currently the only ontology that can successfully support the characterization of structured data (Panov 2012). URL: http://www.ontodm.com

BFO version 1.1 and formal relations from the OBO RO and an extended set of RO relations.[16]

Following best practices in ontology development, the OntoDM-core ontology reuses appropriate classes from a set of ontologies, that act as mid-level ontologies for OntoDM-core (see Table 2). These include the Ontology for Biomedical Investigations (OBI), the Information Artifact Ontology (IAO), and the Software Ontology (SWO). For representing the mining of structured data, we import the OntoDT ontology of datatypes. Classes that are referenced and reused in OntoDM-core are imported into the ontology by using the Minimum Information to Reference an External Ontology

[16] In Table 7 from the Appendix, we list all relations used in OntoDM-core.

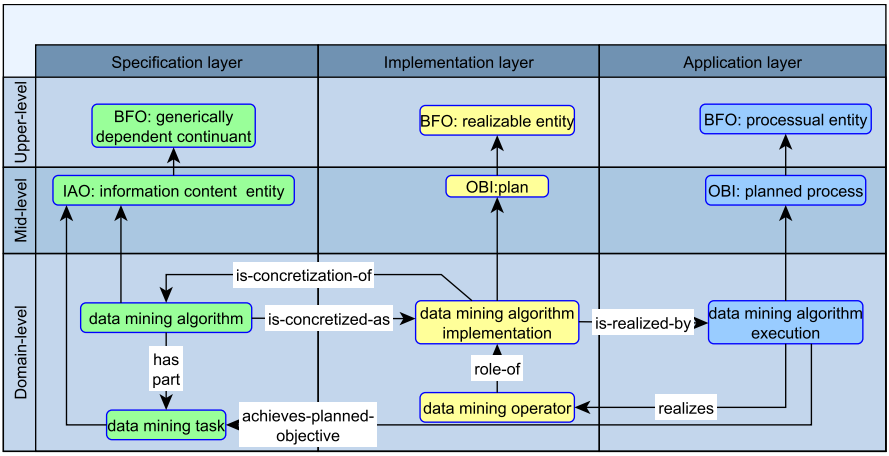


Fig. 1 The horizontal three layer description structure and vertical three level structure of OntoDM-core with example classes represented by rectangles with rounded corners. The ontological relations between classes are represented with directed *labeled arrows*. *Unlabeled arrows* represent IS- A relations

Term (MIREOT) principle (Courtot et al. 2011) and extracted using the OntoFox web service.[17]

3.2 The OntoDM-core design structure

For the domain of DM, we propose a horizontal description structure that includes three layers: a specification layer, an implementation layer, and an application layer. Having all three layers represented separately in the ontology will facilitate different uses of the ontology. For example, the specification layer can be used to reason about data mining algorithms; the implementation layer can be used for search over implementations of data mining algorithms and to compare various implementations; and the application layer can be used for searching through executions of data mining algorithms.

This description structure is based on the use of the upper-level ontology BFO and the extensive reuse of classes from the mid-level ontologies OBI and IAO. The proposed three layer description structure is orthogonal to the vertical ontology architecture which comprises an upper-level, a mid-level, and a domain level. This means that each vertical level contains all three description layers. In Fig. 1, we present examples of OntoDM-core classes from all three layers and all three vertical levels.

In Fig. 1, the specification layer contains *BFO: generically dependent continuants*[18] at the upper-level, and *IAO: information content entities* at the mid-level. In the domain of data mining, example classes are *data mining task* and *data mining algorithm*. The implementation layer describes *BFO: specifically dependent continuants*, such as *BFO: realizable entities* (entities that are executable in a process). At the domain level, this layer contains classes that describe the implementations of algorithms. The

[17] OntoFox: http://ontofox.hegroup.org (accessed 1 June 2014).

[18] In the remainder of the article, italic formatting denotes ontology class.

application layer contains classes that aim at representing processes, e.g., extensions of *BFO: processual entity*. Examples of (planned) process entities in the domain of data mining are the execution of a data mining algorithm and the application of a generalization on new data, among others.

The entities in each layer are connected using general relations, that are layer independent, and layer specific relations. Examples of general relations are IS- A and PART- OF: they can only be used to relate entities from the same description layer. For example, an information entity (member of the specification layer) can not have as parts processual entities (members of the application layer). Layer specific relations can be used only with entities from a specific layer. For example, the relation PRECEDES is only used to relate two processual entities. The description layers are connected using cross-layer relations. An entity from the specification layer IS- CONCRETIZED- AS an entity from the implementation layer. Next, an implementation entity IS- REALIZED- BY an application entity. Finally, an application entity, e.g., a planned process ACHIEVES-PLANNED- OBJECTIVE, which is a specification entity.

4 The key OntoDM-core classes

OntoDM-core is designed to answer a set of competency questions. Examples of such competency questions are: "Which data mining tasks can be formulated on data of datatype X?" and "What is the set of the possible generalization types that are given as output by solving a data mining task X on data of type Y?".[19] In order to support such competency questions, OntoDM-core includes information on data, DM tasks, generalizations, DM algorithms, implementations of algorithms, DM software, the processes of execution of algorithms and others.

In this section, we describe the representation of core data mining entities that appear in the mining of structured data. We discuss the most important representational issues that we encountered in the process of modeling these entities in OntoDM-core. Figure 2 shows for the structure of the key classes in OntoDM-core.

4.1 Data specification and dataset

The main ingredient in the process of data mining is the data. In OntoDM-core, we model the data with a *data specification* entity (see Fig. 2a) that describes the datatype of the underlying data. For this purpose, we import the mechanism for representing arbitrarily complex datatypes from OntoDT ontology (Panov 2012).

In OntoDM-core, we distinguish between a *descriptive data specification*, that specifies the data used for descriptive purposes (e.g., in the clustering and pattern discovery), and *output data specification*, that specifies the data used for output purposes (e.g., classes/targets in predictive modeling). A tuple of primitives or a graph with boolean edges and discrete nodes are examples of data specified only by a descriptive specification. Feature-based data with primitive output and feature-based data

[19] Table 8 in the Appendix lists the typical competency questions OntoDM-core is designed to answer.

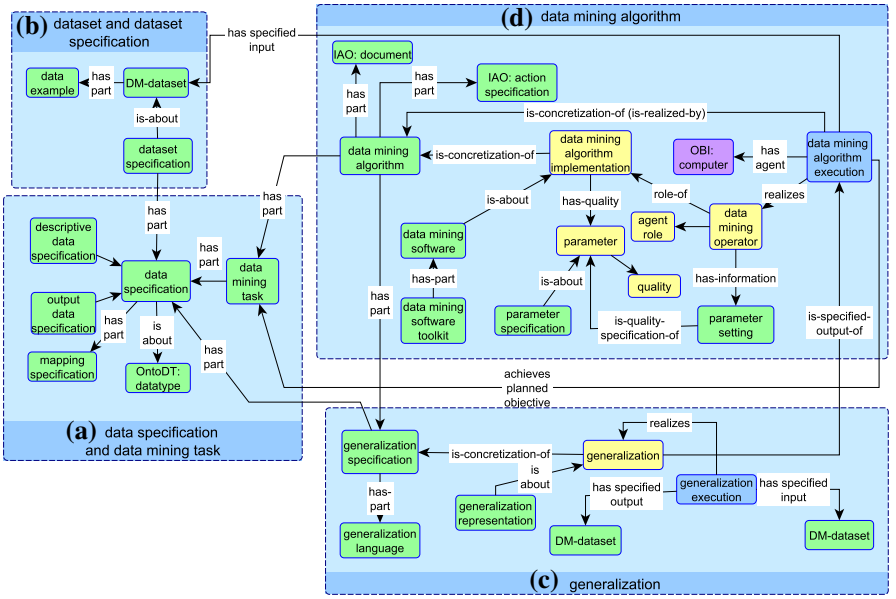


Fig. 2 Key data mining classes and their relations: **a** data specification and data mining task; **b** dataset and dataset specification; **c** generalization; and **d** data mining algorithm

with structured output are examples of data specified by both descriptive and output specifications.

OntoDM imports the IAO class dataset (defined as 'a data item that is an aggregate of other data items of the same type that have something in common') and extends it by further specifying that a *DM dataset* has part *data examples* (see Fig. 2b). OntoDM-core also defines the class *dataset specification* to enable reasoning about data and datasets. It specifies the type of the dataset based on the type of data it contains. Using data specifications and the taxonomy of datatypes from the OntoDT ontology, in OntoDM-core we build a taxonomy of datasets.

4.2 Data mining task

The task of data mining is to produce a generalization from given data. In OntoDM-core, we use the term generalization to denote the outcome of a data mining task. A *data mining task* is defined as sub-class of the IAO class *objective specification* (see Fig. 2a). It is an objective specification that specifies the objective that a data mining algorithm needs to achieve when executed on a dataset to produce as output a generalization.

The definition of a data mining task depends directly on the data specification, and indirectly on the datatype of the data at hand. This allows us to form a taxonomy of data mining tasks based on the type of data. Džeroski (2007) proposes four basic classes of data mining tasks based on the generalizations that are produced as output: *clustering*, *pattern discovery*, *probability distribution estimation*, and *predictive modeling*. These

classes of tasks are included as the first level of the OntoDM-core data mining task taxonomy. They are fundamental and can be defined on an arbitrary type of data. An exception is the predictive modeling task that is defined on a pair of datatypes (for the descriptive and output data separately).

At the next levels, the taxonomy of data mining task depends on the datatype of the descriptive data (in the case of predictive modeling also on the datatype of the output data). If we focus only on the predictive modeling task (Fig. 3) and using the output data specification as a criterion, we distinguish between the *primitive output prediction task* and the *structured output prediction task* (examples of this task can be found in Bakir et al. 2007). In the first case, the output datatype is primitive (e.g., discrete, boolean or real); in the second case, it is some structured datatype (such as a tuple, set, sequence or graph).

Primitive output prediction tasks can be feature-based or structure-based, depending on the datatype of the descriptive part. The *feature-based primitive output prediction tasks* have a tuple of primitives (a set of primitive features) on the description side and a primitive datatype on the output side. This is the most exploited data mining task in traditional single-table data mining, described in all major data mining textbooks (e.g., Hand et al. 2001). If we specify the output datatype in more detail, we have the *binary classification task*, the *multi-class classification task* and the *regression task*; where the output datatype is boolean, discrete or real, respectively. *Structure-based primitive output prediction tasks* operate on data that have some structured datatype (other than tuple of primitives) on the description side and a primitive datatype on the output side.

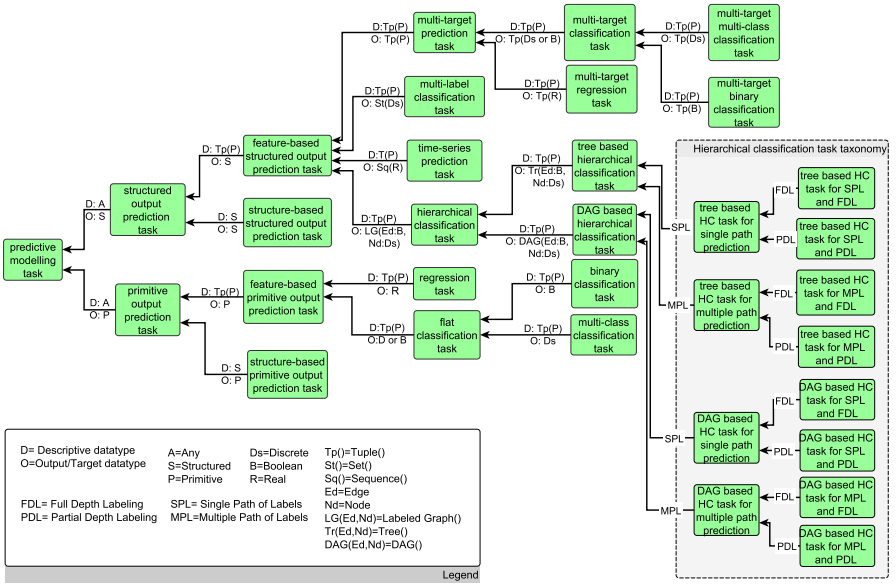


Fig. 3 Taxonomy of predictive modeling tasks. The full line arrows represent IS- A relations. The labels on the arrows are descriptive and output data specifications used as criteria for building the taxonomy

In a similar way, *structured output prediction tasks* can be feature-based or structure-based. *Feature-based structured output prediction tasks* operate on data that have a tuple of primitives on the description side and a structured datatype on the output side. *Structure-based structured output prediction tasks* operate on data that have structured datatypes both on the description side and the output side.

If we focus just on feature-based structured output tasks and further specify a structured output datatype, we can represent a variety of structured output prediction tasks. For example, we can represent the following tasks: *multi-target prediction* (Caruana 1997) (which has as output datatype *tuple of primitives*), *multi-label classification* (Tsoumakas and Katakis 2007) (having as output datatype *set of discrete*), *time-series prediction* (Slavkov et al. 2010) (having as output datatype *sequence of real*) and *hierarchical classification* (Silla and Freitas 2011) (having as output datatype *labeled graph with boolean edges and discrete nodes*). *Multi-target prediction* can be further divided into: *multi-target binary classification*, *multi-target multi-class classification* (Demšar et al. 2006), and *multi-target regression* (Kocev et al. 2009).

4.3 Generalization

We take generalization to denote the outcome of a data mining task. In OntoDM-core, we consider and model three different aspects of generalizations, each aligned with a different description layer (see Fig. 2c): the specification of a generalization, a generalization as a realizable entity, and the process of executing a generalization.

Many different types of generalizations have been considered in the data mining literature. The most fundamental types of generalizations, as proposed by Džeroski (2007) are in line with the data mining tasks. These include clusterings, patterns, probability distributions, and predictive models.

In OntoDM-core, the *generalization specification* class is a subclass of the OBI class *data representational model*. It specifies the type of the generalization and includes as part the *data specification* for the data used to produce the generalization, and the *generalization language*, for the language in which the generalization is expressed. Examples of generalization language formalisms for the case of a *predictive model* include the languages of: trees, rules, Bayesian networks, graphical models, neural networks, etc.

As in the case of datasets and data mining tasks, we can construct a taxonomy of generalizations. In OntoDM-core, at the first level, we distinguish between a *single generalization specification* and an *ensemble specification*. Ensembles of generalizations have as parts single generalizations. We can further extend this taxonomy by taking into account the data mining task and the generalization language.

Generalizations have a dual nature (Džeroski 2007). They can be treated as data structures and as such represented, stored and manipulated. On the other hand, they act as functions and are executed, taking as input data examples and giving as output the result of applying the function to a data example. In OntoDM-core, we define a *generalization* as a sub-class of the BFO class *realizable entity*. It is an output from a *data mining algorithm execution*. The dual nature of generalizations in OntoDM-core is represented with two classes that belong to two different description layers:

generalization representation, which is a sub-class of information content entity and belongs to the specification layer, and *generalization execution*, which is a subclass of planned process and belongs to the application layer (see Fig. 2c).

A *generalization representation* is a sub-class of the IAO class *information content entity*. It represents a formalized description of the generalization, for instance in the form of a formula or text. For example, the output of a decision tree algorithm execution in any data mining software usually includes a textual representation of the generated decision tree. A *generalization execution* is a sub-class of the OBI class *planned process* that has as input a *dataset* and has as output another *dataset*. The output dataset is a result of applying the *generalization* to the examples from the input dataset.

4.4 Data mining algorithm

A *data mining algorithm* is an algorithm (implemented in a computer program), designed to solve a data mining task. It takes as input a dataset of examples of a given datatype and produces as output a generalization (from a given class) on the given datatype. A specific data mining algorithm can typically handle examples of a limited set of datatypes: For example, a rule learning algorithm might handle only tuples of Boolean attributes and a boolean class. In the OntoDM-core ontological framework, we consider three aspects of the DM algorithm entity: a DM algorithm (as a specification), a DM algorithm implementation, and a DM algorithm execution.

Data mining algorithm as a specification is a subclass of the IAO class *plan specification* having as parts a *data mining task*, an *action specification* (reused from IAO), a *generalization specification*, and a *document* (reused from IAO) (see Fig. 2d). The *data mining task* defines the objective that the realized plan should fulfill at the end giving as output a generalization, while the *action specification* describes the actions of the data mining algorithm realized in the process of execution. The *generalization specification* denotes the type of generalization produced by executing the algorithm. Finally, having a *document* class as a part allows us to connect the algorithm to the annotations of documents (journal articles, workshop articles, technical reports) that publish knowledge about the algorithm. In analogy with the taxonomy of datasets, data mining tasks and generalizations, in OntoDM-core we also construct a taxonomy of data mining algorithms. As criteria, we use the data mining task and the generalization produced as the output of the execution of the algorithm.

Data mining algorithm implementation is defined as a sub-class of the BFO class *realizable entity*. It is a concretization of a *data mining algorithm*, in the form of a runnable computer program, and has as qualities *parameters*. The parameters of the algorithm affect its behavior when the algorithm implementation is used as an operator. A parameter itself is specified by a *parameter specification* that includes its name and description.

In OntoDM-core, we define *data mining software* as a sub-class of *directive information entity* (reused from IAO). It represents a specification of a *data mining algorithm implementation*. It has as parts all the meta-information entities about the software implementation such as: *source code*, *software version specification*, *programming language*, *software compiler specification*, *software manufacturer*, the *data min-*

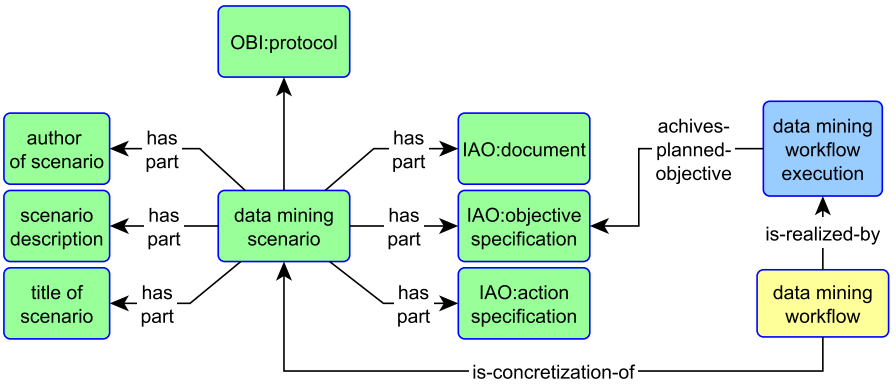


Fig. 4 Scenarios and workflows in OntoDM

ing software toolkit it belongs to, etc. Finally, a *data mining software toolkit* is a specification entity that contains as parts *data mining software* entities.

Data mining operator is defined as sub-class of the BFO class *role*. In that context, it is a role of a *data mining algorithm implementation* that is realized (executed) by a *data mining algorithm execution* process. The *data mining operator* has information about the specific *parameter setting* of the algorithm, in the context of the realization of the operator in the process of execution. The *parameter setting* is a subclass of *data item* (reused from IAO), which is a quality specification of a *parameter*. In OntoDM-core, we define *data mining algorithm execution* as a sub-class of *planned process* (reused from the OBI ontology). A *data mining algorithm execution* realizes (executes) a *data mining operator*, has as input a *dataset*, has as output a *generalization*, has as agent a *computer*, and achieves as a planned objective a *data mining task*.

4.5 Data mining scenario

A scenario is "a postulated sequence or development of events".[20] Therefore, a data mining scenario comprises a logical sequence of actions to infer some type of generalization from a dataset, a sequence of actions for applying a generalization on a new dataset, and a sequence of actions for evaluating the obtained generalizations. OntoDM-core represents a data mining scenario in three different description layers in the ontology: data mining scenario (as a specification), data mining workflow (as an implementation), and data mining workflow execution (as an application).

In OntoDM-core (see Fig. 4), a *data mining scenario* is an extension of the OBI class *protocol*. It includes as parts other information entities such as: *title of scenario*, *scenario description*, *author of scenario*, and *document*. From the protocol class it also inherits as parts *objective specification* and *action specification*. A *data mining workflow* is a concretization of a data mining scenario, and extends the *plan* entity

[20] Oxford dictionary: http://oxforddictionaries.com/definition/scenario (accessed 1 June 2014).

(defined by OBI). Finally, a data mining workflow is realized (executed) through a *data mining workflow execution* process.

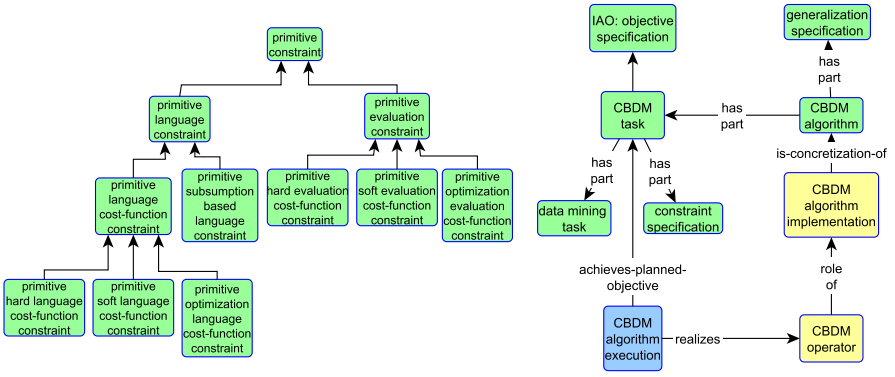
OntoDM-core does not represent scenarios and workflows that belong to other phases of the Knowledge Discovery process, such as application understanding, data understanding, data preprocessing, data mining process evaluation, and deployment. These are the subjects of representation in the OntoDM-KDD ontology (Panov et al. 2013). Because both OntoDM-core and OntoDM-KDD are built by using the same design principles, the same upper-level ontology, and same type of relations they can be used together to represent the complete knowledge discovery process.

4.6 Constraints and constraint-based data mining tasks and algorithms

Constraints play a central role in data mining and constraint-based data mining (CBDM) is now growing in importance (Bayardo 2002). A general statement of the problem involves the specification of a language of generalization and a set of constraints that a generalization needs to satisfy (Mannila and Toivonen 1997). In CBDM, constraints are propositions or statements about generalizations. They can be classified along three dimensions Džeroski (2007): (1) primitive and composite constraints; (2) language and evaluation constraints; and (3) hard (Boolean) constraints, soft constraints and optimization constraints.

A *constraint specification* is defined in OntoDM-core as a sub-class of OBI *data representational model* and is the top-level class of a taxonomy of constraints that we propose. At the first level of the taxonomy, we have the *primitive* and *complex* constraints. Primitive constraints are based on atomic and complex constraints on non-atomic propositions. *Complex constraints* have as parts *primitive constraints* and a *combination function specification* that defines how the primitive constraints are combined to form a complex constraint.

At the second level, if we focus on the primitive constraints (see Fig. 5a), we have *primitive language constraints* and *primitive evaluation constraints*. Language



(a) Taxonomy of primitive constraints. **(b)** CBDM tasks and algorithms.

Fig. 5 Constraints and CBDM tasks in OntoDM-core. **a** Taxonomy of primitive constraints, **b** CBDM tasks and algorithms

constraints concern the representation of a generalization and only refer to its form. Commonly used types of language constraints are subsumption constraints (e.g., all itemsets must contain the item 'bread') and language cost constraints (e.g., itemsets should contain at most three items). Evaluation constraints concern the semantics of a generalization when applied to a dataset. They usually include evaluation functions, where the evaluation functions measure the validity of a generalization on a given dataset (e.g., classification accuracy).

At the last level the *primitive language cost-function constraint* is extended with three sub-classes that include: *primitive hard language cost-function constraint*, *primitive soft language cost-function constraint*, and *primitive optimization language cost-function constraint*. Hard constraints represent boolean functions on generalizations and the constraint can be either satisfied or not satisfied. Soft constraints do not dismiss a generalization that violates a constraint, but rather penalize it for violating a constraint. Optimization constraints ask for a fixed-size set of generalizations that have some extreme values for a given cost or evaluation function. In a similar way, we define the sub-classes of the *primitive evaluation constraint* class.

The task of CBDM is to find a set of generalizations that satisfy a set of constraints, given a dataset that consists of examples of a specific datatype, a data mining task, a generalization specification and a specifications of the set of constraints. In the OntoDM-core ontology, we represent a *CBDM task* (see Fig. 5b) as a sub-class of the *objective specification* class (reused from IAO). It has as parts a *data mining task* and a set of *constraint specifications*. We further define a *CBDM algorithm* as an algorithm that solves a CBDM task and represent it in the same manner as discussed in Sect. 4.4. Finally, this structure allows us to form a taxonomy of CBDM tasks, where at the first level of the taxonomy the basic CBDM task classes are aligned with the fundamental data mining tasks, and then at the next levels depend on the data specification and the type of constraints.

For example, we can represent an algorithm for learning multi-target regression PCTs (Predictive Clustering Trees) under constraints (Struyf and Dzeroski 2005) (see Fig. 6). *Algorithm for learning multi-target regression PCTs with constraints* is an instance of *constraint-based multi-target regression algorithm*. It has as parts *learning multi-target regression PCTs with constraints* task and *multi-target regression PCT*

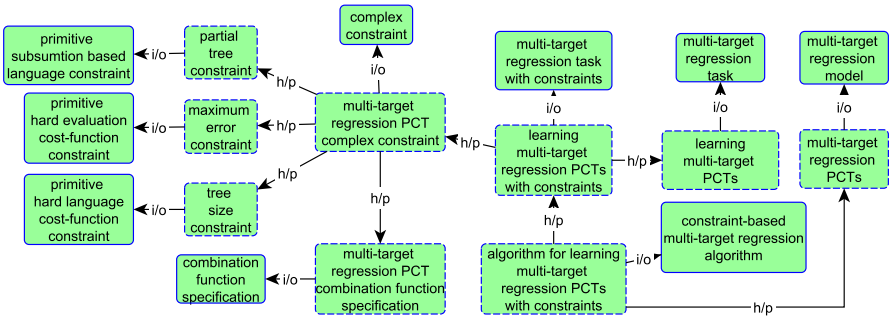


Fig. 6 Example representation of a constraint-based data mining task and an algorithm instance in the specification layer of OntoDM-core. Note that h/p means has-part and i/o instance-of. The *dash lined rectangles* represent instances while the *full-lined rectangles* represent classes

as a generalization specification instance. The CBDM task instance, solved by this algorithm, includes as parts the definition of the basic data mining task instance, in this case *learning multi-target regression PCTs*, and definition of the complex constraint instance, in this case *multi-target regression PCT complex constraint*.

The complex constraint instance contains three primitive constraint instances used in the CBDM task and a combination function specification instance. Each primitive constraint is an instance of a different class of primitive constraints: (1) *partial tree constraint* is instance of *primitive subsumption based language constraint*; (2) *maximum error constraint* is an instance of *primitive hard evaluation cost-function constraint*; and (3) *tree size constraint* is an instance of *primitive hard language cost-function constraint*.

5 Ontology evaluation

We assess the quality of OntoDM-core from three different evaluation aspects. We analyze a set of ontology metrics; assess how well the ontology meets a set of predefined design criteria and ontology best practices; and assess the ontology toward a set of competency questions.

A variety of ontology metrics is available for assessing ontologies (Garcia et al. 2010). In this paper, we use the statistical ontology metrics from the Protégé software and the BioPortal web service. This includes metrics such as the number of classes and individuals, maximum depth, average number of siblings, maximum number of siblings, sub-class axioms count, disjoint classes axioms count, and annotation assertion axioms count. The values of these statistical ontology metrics for OntoDM-core are presented in Table 3.

In Sect. 3, we introduced and presented a set of predefined ontology best practices and design criteria that we intended to use for the development of the OntoDM-core ontology. After the ontology was constructed, we assessed it against these principles in order to see how the finalized ontology fits them. The set of principles (in total 29) are divided into four groups: scope and structural assessment (12 principles); naming and vocabulary assessment; documentation and collaboration assessment; and availability, maintenance, and use assessment.

Table 3 Statistical metrics for the OntoDM-core ontology

Number of classes	856
Number of individuals	480
Maximum depth	20
Avg. number of siblings	9
Max. number of siblings	28
Sub-class axioms count	1,585
Disjoint class axioms	367
Class assertion axioms	506
Annotation axioms	5,826

The scope and structure assessment of OntoDM-core checks its coverage, the use of upper-level ontology, relations, reuse of other ontologies, ontology modularity, use of disjoint classes, use of single inheritance, is-a completeness, established domains and ranges of the relations, use of inverse relations, orthogonality with other ontologies, and instantiability. The naming and vocabulary assessment checks the ontology language, the use of annotation properties, annotation of labels of classes and relations, the definition of the ontology namespace, naming of ontology terms, multi-lingual capabilities, the use of naming conventions, and referencing of external classes. The documentation and collaboration assessment checks the provided definitions of classes and relations, evaluates the documentation of the ontology, and the listed collaboration efforts. The availability, maintenance and use assessment checks the use of reasoners, openness and availability, versioning, the established users of the ontology, maintenance, and handling of obsolete classes. The results of the evaluation are summarized in Tables 9, 10, 11, and 12 of the Appendix. In sum, the pre-defined set of design principles was closely followed during the development of OntoDM-core.

Grüninger and Fox (1995) proposed a methodology for the design and evaluation of ontologies. The methodology proposes to first define the ontology's requirements in the form of questions (or queries) that the ontology must be able to answer, named ontology competency questions. Next, the terminology of the ontology and its classes and relations should be defined. The competency questions in this phase represent informal competency questions, since they are still not expressed in the language of the ontology. Once the informal competency questions have been posed for the new ontology, then the terminology of the ontology (its classes and relations) is specified using some first order logical language (e.g., description logics). This language must provide the necessary terminology to formally restate the informal competency questions.

For every informal competency question, there must be classes, instances and relations in the ontology, which are intuitively required to answer the question. Furthermore, having defined the language of the ontology, the competency questions are defined formally as an entailment queries with respect to the axioms in the ontology. Every newly developed ontology must be accompanied by a set of formal competency questions. In this way, one can evaluate the ontology and claim that it is adequate. Finally, ontologies can be distinguished by the competency questions they can answer.

For the case of the OntoDM-core ontology, in the design phase we established a list of informal competency questions. Having built the ontology and expressed it in a formal ontology language based on description logics (OWL-DL), we can formulate the formal competency questions and show that the ontology is capable of solving and answering the questions it is designed to answer. For that purpose, we formulate the questions using the SPARQL-DL query language (Sirin and Parsia 2007), where the SPARQL-DL is a query language for querying OWL-DL ontologies. SPARQL-DL is a subset of the SPARQL language. In Table 13 of the Appendix, we list all informal competency questions for OntoDM-core with their SPARQL-DL counterparts, while in the next section we present the results of execution of an example query on a populated specification layer segment of OntoDM-core.

6 Querying OntoDM-core

OntoDM-core provides logically defined semantic descriptors to annotate an arbitrary data mining algorithm. Such semantic descriptors would enrich the recording of data mining algorithms, ensure the clarity of their descriptions, promote sharing and reuse of data mining knowledge, and support automatic reasoning over knowledge about algorithms. In this section, we demonstrate the use of the OntoDM-core ontology for this purpose by populating a segment of the specification layer of the ontology with instances from the Clus software system.[21] Furthermore, we perform reasoning using a reasoner and show examples of different queries that can be posed and answered by this populated segment of OntoDM-core. Finally, we show the results of execution of one complex SPARQL query on the inferred segment of OntoDM-core.

6.1 The Clus system

Clus is a decision tree and rule learning system that works in the predictive clustering framework (Blockeel et al. 1998). Clus contains predictive modeling algorithms that solve both primitive and structured output prediction tasks, and produces predictive clustering trees (PCTs) or predictive clustering rules (PCRs) as output generalizations (Ženko and Džeroski 2008). The primitive output prediction tasks include classification and regression, while the structured output prediction tasks include multi-target prediction (multi-target classification and multi-target regression) (Struyf and Dzeroski 2005), hierarchical classification (Vens et al. 2008), multi-label classification (Madjarov et al. 2012), and time-series prediction (Slavkov et al. 2010). Finally, the Clus system was extended with ensemble learning algorithms for all these tasks (Kocev et al. 2013).

6.2 Populating OntoDM-core with Clus specific instances

In order to demonstrate the use of the ontology for automatic reasoning and answering queries, we populated a segment of the specification layer of the ontology with instances (see Fig. 7). This included instances of DM-dataset, dataset specification, data specification (both descriptive and output), predictive modeling task, predictive model (and ensemble) specification, generalization language specification, and predictive modeling (single generalization and ensemble) algorithm. The instances were asserted in a separate OWL file[22] that imports the main ontology and is available on the ontology web page.

In order to provide annotation of data mining algorithms (see Sect. 4), we first specified what are the datatypes that characterize the data that can be used as input to Clus algorithms. For this purpose, we populated the *descriptive data specification* and *output data specification* with instances. Descriptive data specification instances

[21] Clus: http://sourceforge.net/projects/clus/ (accessed 1 June 2014).

[22] Clus OntoDM-core instances: http://ontodm.com/lib/exe/fetch.php?media=clus_instances.owl (accessed 1 June 2014).

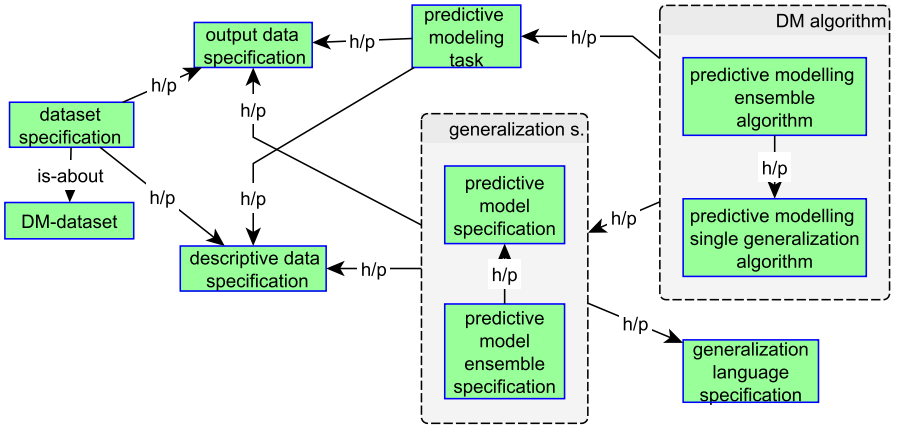


Fig. 7 The class schema of the OntoDM-core specification layer segment populated with Clus instances. Note that h/p means has-part

include only *tuple of primitives*, because all Clus algorithms work on data that have a tuple of primitive features on the descriptive side. Output data specification instances include all primitives (boolean, discrete and real), tuple of primitives (tuple of boolean, discrete, real), set of discrete, sequence of real, tree with boolean edges and discrete nodes, and Directed Acyclic Graph (DAG) with boolean edges and discrete nodes. These are further used to define instances of tasks, generalizations, and datasets.

We populated OntoDM-core with 29 Clus specific data mining task instances. They are all instances of classes from the taxonomy of predictive modeling tasks. For example, the *multi-target classification task* class is populated with the following instances: *learning ensembles of multi-target classification PCRs*, *learning ensembles of multi-target classification PCTs*, *learning multi-target classification PCRs*, and *learning multi-target classification PCTs*.

The generalizations that are output of the Clus system are expressed in the language of predictive clustering trees (PCTs), and predictive clustering rules (PCRs). We populated OntoDM-core with 22 Clus specific generalization specification instances. They are all instances of classes from the taxonomy of generalizations (more specifically *single predictive model specification* class and *ensemble of predictive models specification* class). For example, the *multi-target classification model* class is populated with the following instances: *multi-target classification PCT*, and *multi-target classification PCR*.

We populated OntoDM-core with 48 Clus specific data mining algorithm instances. We included instances of both feature-based predictive modeling algorithms for primitive and structured output, and instances of predictive modeling ensemble algorithms. For example, the *multi-target classification algorithm* class is populated with the following instances: *clus-MTC-PCTs* (algorithm that learns multi-target classification PCTs, i.e., produces a *multi-target classification PCT* as output), and *clus-MTC-PCRs* (algorithm that learns multi-target classification PCRs, i.e., produces *multi-target classification PCRs* as output).

Finally, in order to reason with the ontology and pose queries, we populated OntoDM-core with 40 classes and 79 instances of datasets that were used in real

experiments with the algorithms from the Clus system published in several publications (Kocev et al. 2013; Madjarov et al. 2012). Each dataset instance is specified with a dataset specification, containing the specification of the descriptive and output (or target) datatypes of the data examples in the dataset. Dataset instances that originate from the same source are organized under dataset classes. For example, the *dataset:EDM* class (Karalic and Bratko 1997) class contains 4 instances: *EDM_MCT* (specified with a tuple of real valued targets), *EDM_MDT* (specified with a tuple of discrete valued targets), *EDM_SCT* (specified with a single real valued target), and *EDM_SDT* (specified with a single discrete valued target).

6.3 Types of queries and reasoning

After populating a segment of OntoDM-core with instances, we classified the ontology using the HermiT 1.3.8 reasoner,[23] that is available as a plug-in in the Protégé ontology editor. The inferred ontology was then joined with the asserted ontology, exported as a separate file,[24] and available on the ontology web page.

In general, OntoDM-core can be queried using the SPARQL endpoint on BioPortal.[25] This service allows queries on all asserted ontologies deposited at BioPortal, and in addition allows for execution of federated SPARQL queries. For the purpose of this paper, we queried the OntoDM-core segment using the OWL2Query Protégé plug-in[26] (Kremen and Sirin 2008) that uses SPARQL-DL, and also the built-in SPARQL Protégé engine. OWL2Query is a conjunctive query, meta-query and a visualization engine that facilitates the creation of SPARQL queries using an intuitive graph based syntax and evaluates them by using an OWL-API compliant reasoner. The inferred ontology segment can be also queried using a general purpose SPARQL endpoint named SPARLer,[27] by providing the location of the inferred ontology.

With this setting, we can ask three types of queries: TBox queries (queries that concern only classes), ABox queries (queries that concern only instances), and mixed ABox/TBox queries. Examples of such queries are as follows:

1. TBox queries
   - Find all subclasses of *DM-dataset*.
   - Find all subclasses of *predictive modeling single generalization algorithm* that has as part *structured output prediction task*.
2. ABox queries
   - Find all instances of the Emotions dataset with their data specifications.
   - Find all data mining algorithms that can be applied on the Yeast dataset having as a result a generalizion that is expressed in the language of PCRs.
3. Mixed ABox/TBox queries

[23] Hermit reasoner: http://www.hermit-reasoner.com/ (accessed 1 June 2014).

[24] OntoDM-core inferred segment: http://ontodm.com/lib/exe/fetch.php?media=clus_inferred.owl (accessed 1 June 2014).

[25] BioPortal SPARQL endpoint: http://sparql.bioontology.org (accessed 1 June 2014).

[26] OWL2Query: http://krizik.felk.cvut.cz/km/owl2query/index.html (accessed 1 June 2014).

[27] SPARQLer: http://www.sparql.org/sparql.html (accessed 1 June 2014).

– Find all datasets to which the bagging of multi-target classification PCTs algorithm can be applied.
– Find all datasets on which a hierarchical-classification task can be defined.

6.4 Example query

In this subsection, we analyze one query in more detail and present the results that are obtained by executing the query on the inferred ontology segment. The query is as follows:

> "Find all algorithms that solve a structured output prediction task, produce a generalization expressed in the language of PCTs as output, and are applicable to the EDM dataset".

This query represents a typical ABox query. In Fig. 8, we present the graph representation of the query, the mapping of class/instances/relations labels to URIs, and the SPARQL representation of the query.

The query graph is composed of 6 type query atoms and 10 property value query atoms. Type atoms are presented as type arrows from an ABox node to a TBox node. Property value query atoms are represented with two ABox nodes (subject and object) connected with a property arrow.

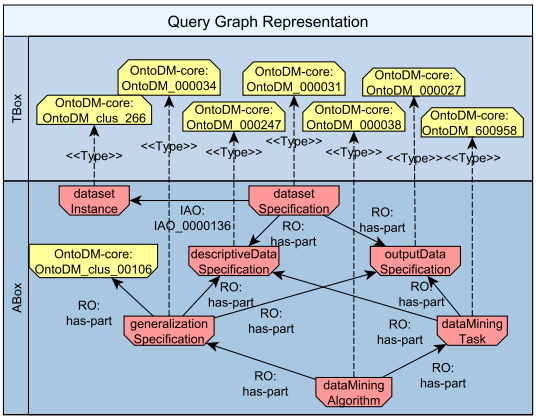
In Table 4, we present results obtained by executing the SPARQL query on the inferred segment of OntoDM-core. The answer to the query are 10 algorithm instances that include algorithms for learning multi-target regression and classification PCTs, and 4 different ensemble algorithms (Bagging, Random Subspace, Random Forest, and SubBag) for each base level algorithm.

7 Use cases

The OntoDM-core ontology can be used to facilitate different applications. These include applications in the automation of the scientific method (representing and reasoning about data mining scenarios, workflows and experiments), semantic annotation of data mining algorithms and software, semantic annotation of articles describing data mining investigations, use of text mining with the ontology to harvest information about the used data mining methods from articles, etc. One of the key features of OntoDM-core is that, due to its design principles, it can be used in cross-domain applications, such as representing investigations within the automated drug discovery pipeline.

7.1 OntoDM-core as a mid-level ontology for Exposé

There is a pressing need for storing information about machine learning experiments and their results in public databases. Currently, thousands of machine learning research papers contain extensive experimental results, but details of those experiments are often



Label	Part of URI/Entity type
language of PCTs	OntoDM_clus_00106 instance
is-about	IAO_0000136 relation
dataset:EDM	OntoDM_clus_00266 class
output data specification	OntoDM_000027 class
dataset specification	OntoDM_000031 class
descriptive data specification	OntoDM_000247 class
data mining task	OntoDM_600958 class
data mining algorithm	OntoDM_000038 class

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX RO: <http://www.obofoundry.org/ro/ro.owl#>
PREFIX OntoDM-clus: <http://www.ontodm.com/clus/>
PREFIX OntoDM-core: <http://www.ontodm.com/OntoDM-core/>
PREFIX OBO: <http://purl.obolibrary.org/obo/>
SELECT DISTINCT ?dataMiningAlgorithm
WHERE { ?dataMiningAlgorithm
   RO:has_part  ?dataMiningTask .
    ?datasetSpecification
    RO:has_part  ?outputDataSpecification .
    ?generalizationSpecification
    RO:has_part  OntoDM-clus:OntoDM_clus_00106 .
    ?datasetSpecification
    RO:has_part  ?descriptiveDataSpecification .
    ?dataMiningTask
    RO:has_part  ?outputDataSpecification .
    ?generalizationSpecification
    RO:has_part  ?descriptiveDataSpecification .
    ?dataMiningAlgorithm
    RO:has_part  ?generalizationSpecification .
    ?generalizationSpecification
    RO:has_part  ?outputDataSpecification .
    ?dataMiningTask
    RO:has_part  ?descriptiveDataSpecification .
    ?datasetSpecification
    OBO:IAO_0000136  ?datasetInstance .
    ?datasetInstance
    rdf:type  OntoDM-clus:OntoDM_clus_00266 .
    ?outputDataSpecification
    rdf:type  OntoDM-core:OntoDM_000027 .
    ?datasetSpecification
    rdf:type  OntoDM-core:OntoDM_000031 .
    ?descriptiveDataSpecification
    rdf:type  OntoDM-core:OntoDM_000247 .
    ?dataMiningTask
    rdf:type  OntoDM-core:OntoDM_600958 .
    ?dataMiningAlgorithm
    rdf:type  OntoDM-core:OntoDM_000038 .
  }
```

Fig. 8 Query graph, mapping of labels and URIs, and the SPARQL query for the example query

Table 4 Results obtained by executing the example SPARQL query on an inferred segment of OntoDM-core by using the Protégé built-in SPARQL engine

dataMiningAlgorithm
algorithm_s:clus-Bagging-MTC-PCTs
algorithm_s:clus-RandomSubspace-MTC-PCTs
algorithm_s:clus-SubBag-MTC-PCTs
algorithm_s:clus-RandomForest-MTC-PCTs
algorithm_s:clus-MTC-PCTs
algorithm_s:clus-MTR-PCTs
algorithm_s:clus-Bagging-MTR-PCTs
algorithm_s:clus-RandomSubspace-MTR-PCTs
algorithm_s:clus-RandomForest-MTR-PCTs
algorithm_s:clus-SubBag-MTR-PCTs

lost after publication. This makes it impossible to reproduce, reuse, or compare such experiments.

Vanschoren et al. (2012) have addressed this need by the implementation of a machine learning experiment database that at present holds over 650,000 experiments: "Experiments are automatically transcribed in a common language" and "are uploaded to pre-designed databases where they are stored in an organized fashion: the results of every experiment are linked to the exact underlying components (such as algorithm, parameter settings and dataset used)". This common language (ExpML), the database design and an overall framework are based on the ontology Exposé (Vanschoren et al. 2012). Exposé, in turn, uses OntoDM-core as a mid-level ontology.

Just like OntoDM-core, Exposé uses the BFO as an upper-level ontology. It relies heavily on using OntoDM-core as a mid-level ontology, re-using both its representational layers and many classes related to algorithms, as described below. Exposé also uses classes form other data mining ontologies (such as DMOP). The use of other ontologies and classes from these ontologies is illustrated in Fig. 9.

Exposé and OntoDM-core follow the same design principles and are therefore fully interoperable. This has enabled Exposé to reuse the OntoDM-core three representational layers (specification, implementation, and application). Moreover, it reuses and further extends the OntoDM-core classes related to algorithms: *algorithm specification*, *algorithm implementation* and *algorithm execution* (or application) for the tasks of classification and regression. In addition, it reuses the OntoDM-core classes representing primitive datatypes, constraints, predictive models, mathematical functions, and evaluation measures.

The machine learning experiments database collects all the details on machine learning experiments, performed and shared by many researchers. It also enables a wide range of new interesting research queries to be answered in future studies. For example, it enables queries related to meta-learning research.

OntoDM-core plays an important role in the provision of the experiment databasese services to the research community. Given its scope, OntoDM-core can facilitate the further development of ontologies like Exposé and experiment databases. The direction of covering a wider range of data mining tasks, such as structured-output prediction, seems especially relevant.

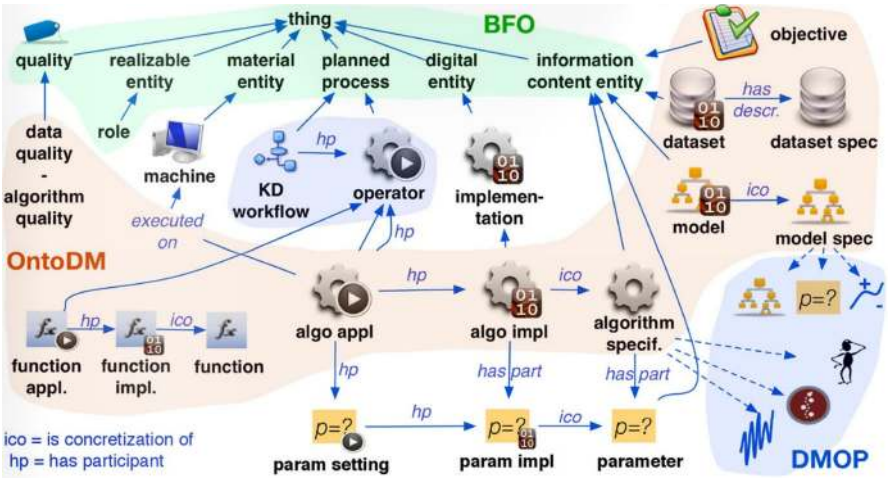


Fig. 9 The Exposé ontology used BFO as a upper-level ontology and OntoDM-core as a mid-level ontology. It reuses many classes from OntoDM-core and some classes from DMOP. The figure has been taken from Vanschoren and Soldatova (2010) with permission of the author

For example, a large body of research is currently conducted on the task of multi-label classification (MLC). This includes the development of different algorithms and their evaluation on a variety of datasets: Comparative studies that investigate the performance of MLC algorithms on different datasets using a plethora of metrics are starting to appear (Madjarov et al. 2012). The development of experiment databases in this direction thus seams well motivated. Since OntoDM-core covers the topic of hierarchical multi-label classification in detail (as can be seen in Sect. 4), we can expect it to prove useful in the context of such developments.

7.2 The annotation of QSAR studies

Quantitative Structure-Activity Relationship (QSAR) modeling is one of the key components of the drug discovery pipeline. QSAR models are used for the rapid prediction and virtual pre-screening of compound activity. A QSAR modeling algorithm is usually a DM algorithm. It receives as input a description of a set of compounds with associated pharmacological activities and outputs a predictive model of activity, i.e., a mapping from the structure of compounds to their activity.

In this use case, we first present how OntoDM-core can be used in combination with the Drug Discovery Investigations (DDI) ontology (Qi et al. 2010) to represent the QSAR modeling process. We then describe a proposal for an annotation schema for annotating QSAR studies and discuss how this schema can be used in context of the Meta-QSAR project.

7.2.1 Representing QSAR modeling with the OntoDM-core and DDI ontologies

The OntoDM-core ontology was designed to support, among others, the representation of the QSAR modeling process for the Robot Scientist project (King et al.

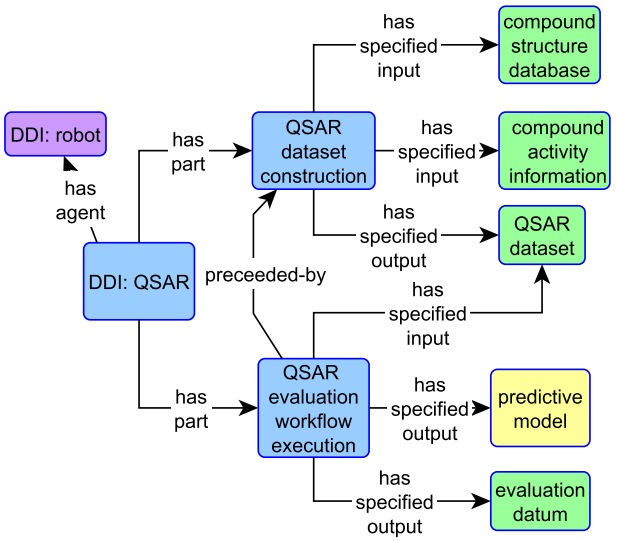


Fig. 10 QSAR modeling for drug discovery investigations with OntoDM-core and DDI

2009).[28] A Robot Scientist "Eve" is an automated laboratory designed to carry out autonomous drug discovery investigations. It is important to note that Eve does not only perform measurements of compound activity, but also performs QSAR modeling in an autonomous fashion. To accurately record Eve's QSAR modeling efforts, appropriate representation of QSAR modeling is a must. Eve works not only with the conventional representation of chemical compounds as a table of features, but also with an innovative relational representation of chemical structure (King et al. 1996) and thus requires descriptions of structured data and building predictive models from such data.

The DDI ontology has been developed to support the recording of data and metadata generated by Eve in a formally defined semantic form. OntoDM-core and DDI were built upon the same design principles, share the same set of relations, and are based on the same upper-level classes, and therefore are fully interoperable. These ontologies can be downloaded together, e.g., into Protégé, and used to represent the principal entities necessary for the recording of QSAR modeling. The OntoDM-core ontology enables accurate recording of all key aspects of the use of data mining for QSAR modeling. It provides logically defined semantic representations of datasets, complex datatypes (propositional and graph-like), QSAR algorithms, predictive modeling tasks, and QSAR model evaluations within the context of a drug discovery investigation.

In Fig. 10, we present an upper-level overview of the representation of the QSAR process using both OntoDM-core (unmarked boxes) and DDI (boxes with the mark DDI:). The class *QSAR* process is modeled in DDI as a part of the class *investigation* process, and HAS- AGENT a *robot*. The QSAR process has two sub-processes: *QSAR dataset construction* and *QSAR evaluation workflow execution* process.

[28] Robot Scientist Project: http://goo.gl/6wazqw and http://goo.gl/Iq6WGS (accessed 1 June 2014).

The *QSAR dataset construction* process is performed by an autonomous agent—the robot scientist Eve. The robot collects input data about compounds from a *compound structure database* and *compound activity information* from the mass screening assays. The output of this process is a *QSAR dataset*. A *QSAR dataset* IS-A *dataset* and it inherits all the properties of the OntoDM-core *dataset* class. The power of the OntoDM-core representation of structured data allows accurate recording of molecular structures by structured datatypes.

The process of *QSAR evaluation workflow execution* is a complex process that involves building of a QSAR model on training data, executing the model on test data and calculation of evaluation functions. OntoDM-core allows the precise recording of the evaluation process and its specifications (e.g., parameter settings of operators). Finally, OntoDM-core can be further used for automated selection of QSAR algorithms based on selected tasks and types of generalizations (linear models, decision rules, regression trees, neural networks) and available datasets.

7.2.2 A schema for annotating QSAR modeling

To facilitate the automated selection of QSAR algorithms, the approach of meta-learning can be applied. Examples of successful and less successful QSAR studies exist in the literature. Many QSAR models are not capable of making accurate predictions for the test sets due to many reasons (Golbraikh and Tropsha 2002). These include errors in the training dataset (structures and activities), the training dataset being too small, an incorrect selection of a training (external) dataset, an incorrect division of the dataset into training and test sets, incorrect measurement of prediction accuracy, and a lack of an external validation (Tropsha 2010).

Experimental validation of QSAR predictive models is very rare (Golbraikh and Tropsha 2002). Young et al. (2008) pointed out the importance of cleaning data as a part of the QSAR model building. Small structural errors within a dataset could lead to significant losses of predictive accuracy of QSAR models that have been built using those erroneous input data (Young et al. 2008). Unfortunately, vital information about the QSAR modeling process, such as quality of the input datasets, provenance of the used datasets, a validation method of the constructed model is often missing. We thus argue for systematic and detailed annotation of QSAR modeling efforts.

From examples of successful and less sucessful QSAR modeling studies, one could learn to recommend QSAR algorithms appropriate to the task at hand. This would require precise descriptions (annotations) both of the QSAR tasks and the algorithms applied to them, as well as the outcome (success) of the modeling effort.

OntoDM-core (along with other ontologies) provides semantic descriptors for the annotation of QSAR studies to ensure that all essential information for the analysis and re-use has been recorded. OntoDM-core is of particular importance for the recording of information about the validation of predictive QSAR models. In addition, the structure of OntoDM-core provides semantic descriptors to annotate not only propositional datasets and algorithms that are most frequently used in QSAR studies, but also structured representations of datasets and algorithms that work on such representations, e.g., representation used by King et al. (1996).

The proposed QSAR annotation schema consists of the following semantic descriptors:

- From BAO (the Bio Assay Ontology):[29] assay provider, assay biosafety level, assay measurement throughput quality, bioassay type (i.e. admet, biding, functional).
- From DMOP: algorithm parameter (i.e. KNN parameter, kernel type parameter, tree pruning parameter), dataset characteristic for the case of propositional datasets (i.e. noise signal ratio, number of outliers, number of missing features).
- From DDI (an ontology for Drug Discovery Investigations) (Qi et al. 2010), imported from BODO (the Blue Obelisk Descriptors Ontology) (Guha et al. 2006) and QSAR-ML (Spjuth et al. 2010): molecular descriptors such as element count, bond count, XlogP, connectivity index, largest chain, etc.
- From OntoDM-KDD (ontology for representing the Knowledge Discovery Process) (Panov et al. 2013): data cleaning, data selection, data preparation.
- From OntoDM-core: predictive model (e.g., regression model, a decision tree), predictive model evaluation, prediction error (e.g., mean squared error), single generalization predictive modeling algorithm, predictive modeling ensemble algorithm (e.g., random forest, bagging, boosting).

We suggest that QSAR datasets deposited into public repositories should be annotated with the proposed QSAR annotation schema to help researchers to effectively evaluate QSAR models and to re-use valuable data.

7.2.3 An example annotation of a QSAR study

We demonstrate the advantages of the proposed QSAR annotation scheme on the example of the Guha Artemisinin QSAR dataset from the Chemoinformatics repository[30] and one of the QSAR models, constructed with the use of this dataset—a linear regression model (Guha and Jurs 2004). In Table 5, we present the example annotation of this QSAR study.

The dataset is available in an MDL Molfile[31] format with activity data in a separate text file. It is not a trivial task to evaluate the quality of this dataset, despite the stated trust level "High—Original Author Data". The dataset (Guha and Jurs 2004) has been derived from the dataset provided by Avery et al. (2002) by the removal of 32 data items (chemical structures): the dataset is deposited to CHEMBLE repository.[32]

An interested reader would notice that the dataset reported in Avery et al. (2002) contains 100 new data items and 111 are data items from different sources: "100 analogues from our laboratory and a number from the literature to assemble the 211 artemisinin analogues for the database." Only when the interested reader would get hold of the report by Avery et al. (1999), it would become clear how the whole dataset

[29] BAO: http://bioassayontology.org (accessed 1 June 2014).

[30] QSAR Chemoinformatics repository: http://cheminformatics.org/datasets/#qsar (accessed 1 June 2014).

[31] Example MDL Molfile http://mychem.sourceforge.net/doc/apes06.html (accessed 1 June 2014).

[32] CHEMBLE repository: https://www.ebi.ac.uk/chembl/doc/inspect/CHEMBL1135798 (accessed 1 June 2014).

Table 5 An example annotation of a QSAR study (a fragment)

OntoDM-core: data set	Guha Artemisinin QSAR dataset
BAO:assay provider	Rajarshi Guha, 152 Davey Laboratory—Chemistry, Penn State University, University Park, Pennsylvania 16802.
OntoDM-core: derived from dataset	Avery et al. Artemisinin dataset
OntoDM-KDD: data preparation	Removal of 32 examples (one of the members of each enantiomer pair with the lowest log RA)
OntoDM-KDD: data cleaning	Removal of 3 outliers
BAO: assay measurement throughput quality	Not known
BAO: bioassay type	Functional
DMOP: number of instances	179 (molecule structures—artemisinin analogues).
DMOP: number of outliers	3 (molecules with the same log RA of −4.0, a least trimmed squares regression algorithm was employed)
DMOP: number of features	65
DMOP: feature correlation	>0.8
OntoDM-core: predictive model	Linear regression model
DDI(BODO): molecular descriptor	N7CH—number of seventh-order chains; NSB-12—number of single bonds; WTPT-2—the molecular ID number considering only carbon atoms; MDE-14—the molecular distance edge vector, considering only primary and quaternary atoms.
OntoDM-core: train set	144
OntoDM-core: cv set	17
OntoDM-core: test set	18
Onto-DM-core: training set evaluation process	Ranking (ranking the molecules according to their dependent variable value)
OntoDM-core: predictive model evaluation	A leave-one-out cross-validation
OntoDM-core: prediction error	The RMSE for the training set: 0.77; The RMSE for the prediction set: 0.77; The R2 value for the training is 0.74.
OntoDM-core: experimental validation	Not known

has been constructed. (Note that none of the articles mentioned in this section are open access and are therefore not immediately available to researchers.)

Interestingly, Avery et al. (1999) point out: "Prediction of the in vitro activity for the racemic pair of compounds leads to a less clear interpretation of the data. Is only one enantiomer active? If so, should the biological activity value be 'doubled' assuming only half of the assayed material is active? If the enantiomers are unequally active, how should the data be treated?" The Guha dataset has been derived from the Avery et al. dataset by the removal of one of the members of each enantiomer pair[33] with lowest logRA (Relative Activity).

The activity values in the Avery et al. assay are open to various interpretations. The removal of 32 data items could potentially decrease the quality of the Guha dataset. Normally, it is beneficial to use all the available information. Therefore it is important

[33] A pair of molecules consisting of one chiral molecule and the mirror image of this molecule. The molecules making up an enantiomeric pair rotate the plane of polarized light in equal, but opposite, directions.

to record in an easily accessible form how this dataset has been produced to enable informed decisions on how to use this data and the produced QSAR models. The suggested ontology-based annotation would ease the analysis of this dataset and the linear QSAR model reported in (Guha and Jurs 2004) (see Table 5).

Several QSAR models have been built with the use of the described Artemisinin datasets, e.g., linear regression models, neural networks, 3D QSAR and a hologram QSAR. However, currently it is difficult to compare models: "A direct comparison with the original work is not feasible as the model development process in this study was different" (Guha and Jurs 2004). The proposed annotation scheme would ease the comparison of QSAR models. One would argue that the predictive power of the models is a decisive factor, while other would prefer experimentally validated models. For example, the reported 3D QSAR model (Avery et al. 1999) has higher prediction error then the considered model ($R^2 = 0.63$ for the training set), but it has been experimentally verified.

7.2.4 Meta-QSAR

The meta-QSAR project EP/K030469/1, recently funded by the UK Engineering and Physical Sciences Research Council (EPSRC), aims to systematically run comparative QSAR studies to determine what combinations of datasets, DM algorithms, and drug targets work best. The project aims to output a knowledge base containing recommendations about what DM algorithms with what parameters to use for specified drug targets, representations of molecular structures, and quality of datasets (see Fig. 11a). The project has two main stages: the base-level QSAR learning, where molecular descriptors provided by BODO/ DDI will be used to run millions of QSAR studies, and the meta level QSAR, learning where the proposed QSAR annotation schema composed by the descriptors provided by BAO, DMOP, OntoDM-core and OntoDM-KDD.

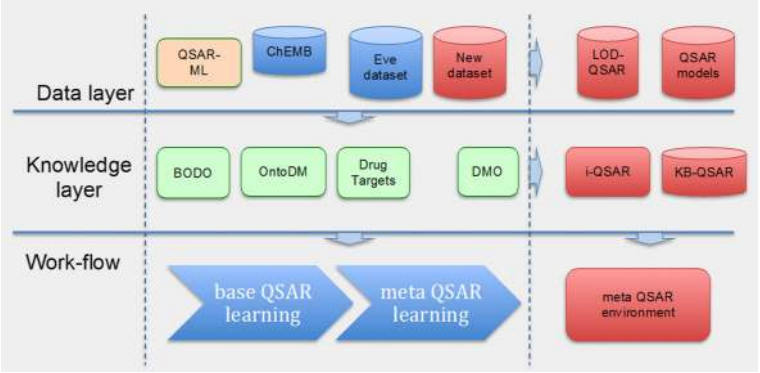
Descriptions of the base-level learning process and its outcomes described in the representation provided by OntoDM-KDD and OntoDM-core will be used to learn about the performance of different QSAR models (see Fig. 11b). Thus the QSAR annotation schema is essential for the achieving of a new goal: the systematic comparative analysis of QSAR studies. Potentially, it may lead to significant societal benefits: new drugs could be delivered to the market faster and cheaper.

7.3 Use of OntoDM-core for disambiguation of terms in text mining

Ontologies are widely used in text mining for a number of tasks, e.g., named entity recognition (NER), disambiguation of terms, event extraction, and others. OntoDM-core can assist in text mining of DM articles. It can do so by providing the terminology to describe DM investigations.

We will demonstrate how OntoDM-core could help in text mining by annotating an example paper by Ford et al. (2004) from the ART Corpus of Biochemistry papers.[34]

[34] ART: http://www.aber.ac.uk/en/cs/research/cb/projects/art/art-corpus (accessed 1 June 2014).



(a) Stages of the meta-QSAR project.

Representation of molecular structure	BAO: measurement throughput quality	...	OntoDM-core: prediction error	OntoDM-core: predictive model	DMOP: algorithm parameter	Drug target
relational	high		0.01	N. Bayes		Single protein
relational	high		0.01	NN	3	Chimeric protein
tabular	high		0.05	N. Bayes		Single protein
tabular	high		0.05	N. Bayes		Single protein
tabular	medium		0.01	NN	4	Chimeric protein
tabular	medium		0.05	N. Bayes		Chimeric protein
tabular	high		0.05	NN	4	Single protein
relational	high		0.01	NN	3	Single protein
tabular	unknown		0.01	NN	3	Chimeric protein
tabular	unknown		0.01	NN	3	Chimeric protein
tabular	high		0.01	NN	3	Chimeric protein

(b) Meta-QSAR learning.

Fig. 11 The meta-QSAR project. **a** Stages of the meta-QSAR project, **b** meta-QSAR learning

The Chemistry using Text Annotations (CheTA) project by the UK National Centre for Text Mining[35] used the Open-Source Chemistry Analysis Routines (OSCAR) toolkit for NER in chemistry publications to annotate papers from the ART Corpus. OSCAR employs domain ontologies to support annotation and three OBO ontologies, ChEBI,[36] FIX,[37] and REX,[38] were used in the CheTA project. However, depending on the goals

[35] NACTEM centre: http://www.nactem.ac.uk/cheta (accessed 1 June 2014).

[36] ChEBI: http://www.ebi.ac.uk/chebi (accessed 1 June 2014).

[37] FIX: http://obofoundry.org/cgi-bin/detail.cgi?fix (accessed 1 June 2014).

[38] REX: http://www.obofoundry.org/cgi-bin/detail.cgi?id=rex (accessed 1 June 2014).

of annotation, i.e., identification of most popular methods for prediction of biological activity of compounds and extraction of information about such methods, or reasoning about molecular descriptors and chemical diversity, it may be desirable to use other domain specific ontologies (such as OntoDM-core).

The considered paper, Ford et al. (2004), reports on QSAR studies with a new molecular descriptor EVA. OntoDM-core could assist in NER of DM terms used in QSAR studies, such as *training/test set*, *regression model*, *cross-validation*, while the OBO ontologies would fail to do so. The paper has three instances of the class *cross-validation*: Leave-One-Out Cross-Validation, Leave-n-Out Cross-Validation, and Leave-Group-Out Cross-Validation. It also has two instances of the class *regression model*: PLS regression model and QSAR regression model;

The paper further concerns three instances of the class *dataset*: the Calcium Channel Agonist data set, 36 compound data set, and proprietary data sets (Shell Research Ltd., the former Sittingbourne Research Centre). It also has several statements about *training/test set*: "test set of 76 compounds" and "the training set is comprised of the same number of well-defined components". OntoDM could also assist in the recognition of other DM relevant terms, i.e., "robustness of the model" is a *generalization quality* and "model validity" is a *generalization evaluation*. OntoDM may be used for reasoning over recognized entities through the defined relations between those entities.

Furthermore, OntoDM-core may assist in the disambiguation of terms. For example, the term EVA was classified by OSCAR as a chemical compound with a likelihood 0.22. OntoDM, on the other hand, recognizes "biological activity EVA" as an instance of the class *feature specification* via the synonym "molecular descriptor".

8 Discussion

A standardized community-agreed formally defined representation of DM entities would have a significant influence on the whole area of data mining. It would support efficient data and knowledge exchange, and also the integration of DM knowledge with the applications domain knowledge. Unfortunately, ontology engineering research in the area of data mining still lags behind, compared with the advances of ontological research in biomedicine. Several lessons can be learned from this more advanced research. In this section, we summarize our experience in the ontology development in the from of lessons learned and present a mapping of DM core terms in different ontologies in order to facilitate knowledge sharing and interoperability.

Lesson 1 *There is a pressing need for a DM domain-specific upper-level ontology that would define the most essential DM entities and a set of relations between them and would act as a mid-level ontology for other ontology development efforts.*

Recently, the DMOP ontology (Keet et al. 2013) has aligned its representations with the upper-level ontology DOLCE (Gangemi et al. 2002). DOLCE works best with natural language representations and applications, but is difficult to use for other applications, for example, the representation of biomedical experiments. BFO is widely used within biomedical representations (Grenon et al. 2004), and this seriously complicates

an integration of DOLCE-driven resources with BFO-compliant ones. There is a need for semantic consistency (Smith and Ceusters 2010), and domain-specific upper-level ontologies can ensure such a consistency within the relevant domains.

For example, a domain-specific upper-level ontology for the area of medicine, i.e., Ontology for General Medical Science[39] (OGMS), has been recently proposed. It defines entities commonly used in medicine, such as *disease, symptom, diagnosis.* Previously, these terms have been defined in different resources inconsistently, and it has been difficult to integrate medical data across different areas and applications. OGMS uses BFO as an upper-level ontology and extends its classes with domain-specific subclasses. Currently, several OBO Foundry ontologies are being re-designed to align its representations with OGMS to ensure data and knowledge interoperability within the area of biomedicine.

There are similar efforts in other areas of research. The Core Enterprise Ontology[40] (CEO) project aims to develop a Core Reference Ontology for the representation of the key entities and processes in enterprises. The Web Service Modeling Ontology [41] (WSMO) working group aligns the research and development efforts in the areas of Semantic Web Services.

Lesson 2 *The development of an ontology is a time-costly endeavour. The OntoDM-core ontology representation, reported in this paper is a result of five years of intensive research. This is comparable with other ongoing ontology development projects. For example, the OBI project exists for seven years and combines researchers from over 20 international research communities, the IAO research has been carried out for five years, and FMA is the result of ten years work in the area.*

While there are automated data-driven approaches to ontology development, the quality of their outputs mostly remains unsatisfactory for the purposes described in this paper. Text mining tools are capable of analysing thousands of texts to extract frequently used terminology and even group these terms thematically. For example, the Taxonomy for rehabilitation of knee conditions (TRAK) ontology has been developed with the use of such techniques (Button et al. 2013). However, significant manual modeling is still required to achieve an accurate, logically consistent representation. OntoDM-core has been developed manually, but it would have benefited from complementary data-driven approaches. A combination of top-down and bottom-up design approaches could make the development process more efficient. When a key representational structure is in place, data-driven approaches could be used to further populate an ontology.

Lesson 3 *Currently, the main benefit of ontological representations is the integration of different, sometimes heterogeneous, data resources. Bio-ontologies are used to define semantic mappings between different bioinformatics databases and for federated queries across several domains. While such applications are also of benefit to the*

[39] OGMS: www.obofoundry.org/cgi-bin/detail.cgi?id=OGMS (accessed 1 June 2014).

[40] CEO: http://goo.gl/AUktCK (accessed 1 June 2014).

[41] WSMO: http://www.wsmo.org (accessed 1 June 2014).

DM domain, we argue that potentially the greatest benefit of ontological representations is in the support of knowledge discovery process.

Machine-processable logical representations of DM knowledge, integrated with applications' knowledge and data representations, would lead to more intelligent systems and new discoveries. The power of deductive and abductive inference with the use of ontologies has been little exploited so far, leaving plenty of room for further development. The dominating form of ontological representations is description logic, which is an excellent mechanism for classification tasks (in terms of reasoning): Ontologies in OWL-DL should be able to greatly facilitate DM classification algorithms. Finally, we envisage the use of ontologies along with knowledge bases defining inference rules for the entities formalized in the ontologies to enhance data mining and KDD research.

A mapping of DM core terms. In the absence of a community agreed DM-core ontology, it is not surprising that terms essential to data mining, such as *dataset, task, algorithm*, are defined in several DM resources. Unfortunately, sometimes the intended semantics/meaning of these terms differ between these resources. For example, the DMOP ontology (version 5.4) defines the class *DM-task* as: "any task that needs to be addressed in the data mining process", while the OntoDM-core ontology defines the class *data mining task* as: "an objective specification that specifies the objective that a data mining algorithm needs to achieve when executed on a dataset to produce as output a generalization".

In order to facilitate knowledge sharing and interoperability of computer applications the research community has to resolve such differences in the semantics of all core data mining terms. In the absence of a community agreed DM-core resource, we provide an explicit machine processable mapping of the key DM terms between some of the DM ontologies that are available. The origin of this mapping is the Data Mining Ontology Jamboree (2010)[42] held in Ljubljana, Slovenia, where a mapping between OntoDM, DMOP and Exposé classes was discussed.

We do not resolve the differences in the meanings of the key DM terms, but instead, we evaluate if those differences are such that the terms could be mapped by the OWL construct *same as*. For example we judge that the semantic meanings of the classes *DMOP: DM-Task* and *OntoDM-core: data mining task* are close enough to justify such a mapping (see Table 6). The proposed mapping of the key DM terms could be used by computer applications that require terms from several DM ontologies, for example by an application supporting the annotation of QSAR studies (see Sect. 7.2). The proposed mapping is a step towards a community agreed DM-core standard.

9 Conclusions and further work

Summary. Data mining has been used for a large variety of applications. There is a pressing need for an interoperable representation of both data mining and application

[42] DMO Jamboree 2010: http://kt.ijs.si/janez_kranjc/dmo_jamboree/ (accessed 1 June 2014).

Table 6 Mapping of classes representing data mining core entities between different ontologies

OntoDM-core	DMOP Hilario et al. (2011)	Exposé Vanschoren et al. (2012)	KD Ontology Žáková et al. (2010)	KDDOnto Diamantini and Potena (2008)
DM-dataset	DataSet	Dataset	Dataset	Dataset
Data mining task	DM-Task	Data mining task	Data mining task	Task
Generalization	DM-Model and DM-Pattern Set	Model and pattern set	Model	Model
Data mining algorithm	DM-Algorithm	Algorithm specification	Algorithm	Algorithm
Data mining algorithm implementation	DM-Operator	Algorithm implementation	N/A	N/A
Data mining algorithm execution	DM-Operation	Algorithm application	Algorithm execution	N/A

domains knowledge to support interdisciplinary research. To address this need, we proposed an ontology of core data mining entities OntoDM-core that is fully interoperable with many domain resources, is agile, and easily extensible. The strength and advantage of the OntoDM-core ontology is in the generic representation used to describe the mining of structured data (e.g., structured output prediction) and constraint based data mining. It is easily extendable to cover new data mining tasks and algorithms that operate on data of arbitrarily complex datatypes.

The OntoDM-core ontology is designed and implemented by following ontology best practices and design principles. It uses an upper-level ontology (BFO) as a template, includes formally defined relations (from RO and other state-of-the-art ontologies), and reuses classes and relations from other ontologies for representing scientific investigations (such as OBI, IAO, and SWO). In addition, it uses a three-layer description structure (specification, implementation, application) for representing entities from the domain of DM, where each description layer represents classes with fundamentally different nature. Finally, the ontology is developed in a general fashion in order to be used as a mid-level ontology, and can be easily extended further by other ontologies that focus on a specific part of the DM domain.

The OntoDM-core ontology has been designed to support a large variety of applications, as demonstrated by the use cases presented. OntoDM-core supports the annotation and representation of data mining algorithms, data mining scenarios, and data mining investigations. Furthermore, it provides support for the annotation and comparison of QSAR studies in the context of drug design investigations. Combined with text mining, it can support the annotation of articles containing data mining terms. Finally, the OntoDM ontology can be used as a mid-level ontology (as demonstrated in the case of the Exposé ontology).

Future work. In future developments of the OntoDM-core ontology, we plan to focus on several aspects. First, we plan to align and map our ontology to other upper-level ontologies, for example to YAMATO (Mizoguchi 2010). A mapping to YAMATO would additionally provide OntoDM-core with the possibility of representing change

(e.g., in the case of dynamic datasets), more expressiveness for representing events, and a well developed framework for dealing with representations.

In the context of data mining entities, we plan to extend OntoDM-core with components of data mining algorithms, such as distance functions and kernel functions. This would enable the formulation of a taxonomy of distance and kernel functions based on the type of data. Next, we want to populate the ontology downward with instances. In that context, we can first import terms from other ontologies representing data mining entities developed for the case of single-table data mining, such as DMOP (Hilario et al. 2011) and Exposé (Vanschoren et al. 2012). Finally, we want to use the OntoDM-core framework to support the representation of dynamical systems and their behavior, as well as predictive models thereof, such as systems of ordinary differential equations.

Acknowledgments We would like to acknowledge the support of the European Commission through the project MAESTRA—Learning from Massive, Incompletely annotated, and Structured Data (Grant Number ICT-2013-612944).

Appendix

Table 7 Relations used in the OntoDM-core ontology

Origin	Relation	Inverse relation (if defined)
Fundamental	IS- A	HAS- SUBCLASS
	HAS- INSTANCE	INSTANCE- OF
BFO 2.0 & OBO RO	HAS- PART	PART- OF
	HAS- PARTICIPANT	PARTICIPATES- IN
	HAS- ACTIVE- PARTICIPANT	IS- ACTIVE- PARTICIPANT- OF
	PRECEEDES	PRECEEDED- BY
	INHERES- IN	BEARER- OF
	HAS- QUALITY	IS- QUALITY- OF
	HAS- ROLE	IS- ROLE- OF
	IS- CONCRETIZED- AS	IS- CONCRETIZATION- OF
	REALIZES	IS- REALIZED- BY
IAO	IS- ABOUT	
	DENOTES	
	QUALITY- IS- SPECIFIED- AS	IS- QUALITY- SPECIFICATION- OF
OBI	ACHIEVES- PLANNED- OBJECTIVE	OBJECTIVE- ACHIEVED- BY
	HAS- SPECIFIED- INPUT	IS- SPECIFIED- INPUT- OF
	HAS- SPECIFIED- OUTPUT	IS- SPECIFIED- OUTPUT- OF
	IS- MANUFACTURED- BY	
EXACT	HAS- INFORMATION	
LABORS	HAS- REPRESENTATION	IS- REPRESENTATION- OF

Table 8 Examples of OntoDM-core competency questions

1. Which datasets have data belonging to a datatype X?	7. Which data mining tasks can be formulated on data of datatype X?
2. Which data mining tasks can be formulated on a dataset X?	8. Which data mining algorithms solve a data mining task X?
3. Which data mining algorithms are applicable to data of datatype X?	9. What is the set of the possible generalization types that are given as output by solving a data mining task X on data of type Y?
4. What is the set of implementations for a DM algorithm X that have a parameter Y?	10. What is the set of all generated generalizations on a dataset X?
5. Which DM software toolkit contains an implementation of a DM algorithm X?	11. On which computer a specific data mining algorithm X has been executed?
6. What is the parameter setting of a given execution of a data mining algorithm X on a computer Y, having as input a dataset Z?	12. What is the set of datasets on which a given generalization X has been executed?

Table 9 Scope and structure assessment

No.	Principle	Assessment
1	Coverage	OntoDM-core provides a representation of core data mining entities, and it is general enough to represent the mining of structured data.
2	Upper-ontology	OntoDM-core uses the BFO ontology as an upper-level ontology.
3	Relations	OntoDM-core uses relations defined in RO, IAO, EXACT and OBI. The relations defined in IAO and OBI are candidates for inclusion into RO.
4	Ontology reuse	OntoDM-core reuses classes and relations from OBI, IAO, SWO and EXACT.
5	Modularity	OntoDM-core is part of the OntoDM ontology, that contains also OntoDT and OntoDM-KDD. It can also be used independently.
6	Use of disjoint classes	In OntoDM- core, we extensively use disjoint class axioms.
7	Use of single inheritance	In OntoDM-core, each class has only one superclass. This reduces the potential inconsistency and errors in reasoning processes.
8	IS- A completeness	All the OntoDM-core classes are connected via the IS- A relation. There are no orphan classes.
9	Domains and ranges for relations	Imported relations from RO, IAO and OBI have defined ranges and domains.
10	Inverse relations	Most of the imported relations from RO, OBI, and IAO have defined inverse relations.
11	Orthogonality with other ontologies	OntoDM-core is orthogonal to other ontologies already lodged within OBO.
12	Instantability	Some classes from OntoDM-core have defined instances. In future work, more extensive population of the ontology with instances is planned.

Table 10 Naming and vocabulary assessment

No.	Principle	Assessment
1	Ontology language	OntoDM-core is expressed in the W3C standard Web Ontology Language OWL-DL.
2	Use of annotation properties	We reuse the OBI consortium defined meta-data (http://obi-ontology.org/page/OBI_Minimal_metadata) to provide additional semantic annotation of the classes and relations.
3	Label annotations	We use label annotations to provide human readable names of classes and relations in the ontology.
4	Ontology namespace	OntoDM-core has its own namespace (http://www.ontodm.com/OntoDM-core/). The classes and relations that are imported from other ontologies have kept their source ontology namespace and ID.
5	Ontology term IDs	The IDs of the ontology terms include a combination of an ontology module ID and a five digit code. For the OntoDM-core module we use `OntoDM_xxxxx`.
6	Multi-lingual capabilities	At this moment the OntoDM-core ontology does not provide multi-lingual capabilities.
7	Naming conventions	The ontology uses set of naming conventions provided by the OBO Foundry.
8	Referencing external classes	The external classes are referenced by using the MIREOT principle.

Table 11 Documentation and collaboration assessment

No.	Principle	Assessment
1	Definitions	Most of the OntoDM-core classes have textual definitions. They are regularly updated and revised.
2	Documentation	The ontology is documented on its dedicated web page and in a doctoral disertation.
3	Collaboration efforts	OntoDM-core participates in the Data Mining Ontology (DMO) Foundry with the aim of developing a core data mining mid-level ontology.

Table 12 Availability, maintenance and use assessment

No.	Principle	Assessment
1	Use of reasoners	To test the class and relations consistency we use the Pellet and Hermit reasoners.
2	Openness and availability	OntoDM-core is open and is available in its web page http://www.ontodm.com and additionally at BioPortal (http://bioportal.bioontology.org/).
3	Versioning	For tracking the changes in the ontology we use the industry standard Subversion tool.
4	Users of the ontology	The ontology has established a set of users. The ontology is reused by Expose ontology.
5	Maintenance	The ontology has a dedicated person that cares about its maintenance.
6	Handling of obsolete classes	Deleted classes in the OntoDM-core class hierarchy are listed under the *obsolete* class, so that applications based on them can still use the terms. The domain terms that have been collected so far but are still not represented in the ontology are listed under the *non-curated* class in the OntoDM class hierarchy.

Table 13 Formalization of the OntoDM-core competency questions using the SPARQL-DL language

1. Which datasets have data belonging to a datatype X? *Q1(d):-Type(?d,dataset),* *PropertyValue(?ds,is-about,?d),* *PropertyValue(?ds,has-part,?datas),* *PropertyValue(?datas,is-about,x),* *Type(x,datatype).*	7. What is the set of implementations for a DM algorithm X that have a parameter Y? *Q7(dmi):-* *Type(?dmi,data_mining_algorithm_implementation),* *PropertyValue(?dmi,is-concretization-of,x),* *Type(x,data_mining_algorithm),* *PropertyValue(?dmi,has-quality,y),* *Type(y,parameter).*
2. Which data mining tasks can be formulated on data of datatype X? *Q2(dmt):-Type(?dmt,data_mining_task),* *PropertyValue(?dmt,has-part,?datas),* *PropertyValue(?datas,is-about,x),* *Type(x,datatype).*	8. What is the set of all generated generalizations on a dataset X? *Q8(g):-Type(?g,generalization),* *PropertyValue(?g,is-specified-output-of,?dmae),* *PropertyValue(?dmae,has-specified-input,x),* *Type(x,dataset).*
3. Which data mining tasks can be formulated on a dataset X? *Q3(dmt):-Type(?dmt,data_mining_task),* *PropertyValue(?dmt,has-part,?datas),* *PropertyValue(?ds,has-part,?datas),* *PropertyValue(?datas,is-about,x),* *Type(x,dataset).*	9. Which DM software toolkit contains an implementation of a DM algorithm X? *Q9(dmst):-Type(?dmst,data_mining_software_toolkit),* *PropertyValue(?dmst,has-part,?dms),* *PropertyValue(?dms,is-about,?dmi),* *PropertyValue(?dmi,is-concretization-of,x),* *Type(x,data_mining_algorithm).*
4. Which data mining algorithms solve a data mining task X? *Q4(dma):-Type(?dma,data_mining_algorithm),* *PropertyValue(?dma,has-part,x),* *Type(x,data_mining_task).*	10. On which computer has a specific data mining algorithm X been executed? *Q10(c):-Type(?c,computer),* *PropertyValue(?c,is-agent-of,?dmae),* *PropertyValue(?dmae,is-realized-by,?dmo),* *PropertyValue(?dmo,is-concretization-of,x),* *Type(x,data_mining_algorithm).*
5. Which data mining algorithms are applicable to data of datatype X? *Q5(dma):-Type(?dma,data_mining_algorithm),* *PropertyValue(?dma,has-part,?dmt),* *PropertyValue(?dmt,has-part,?datas),* *PropertyValue(?datas,is-about,x),* *Type(x,datatype).*	11. What is the parameter setting of a given execution of a data mining algorithm X on a computer Y, having as input a dataset Z? *Q11(ps):-Type(?ps,parameter_setting),* *PropertyValue(?dmo,has-information,?ps),* *PropertyValue(x,realizes,?dmo),* *Type(x,data_mining_algorithm_execution),* *PropertyValue(?dmae,has-agent,Y),* *Type(y,data_mining_algorithm),* *PropertyValue(?dmae,has-specified-input,z),* *Type(z,dataset).*
6. What is the set of the possible generalization types that are given as output by solving a data mining task X on data of type Y? *Q6(gs):-Type(?gs, generalization_specification),* *PropertyValue(?gs,has-part,?datas),* *PropertyValue(x,has-part,?datas),* *Type(x,data_mining_task),* *PropertyValue(?datas,is-about,y),* *Type(y,datatype).*	12. What is the set of datasets on which a given generalization X has been executed? *Q12(dmst):-Type(?d,dataset),* *PropertyValue(?ge,has_specified_input,?d),* *PropertyValue(?ge,realizes,?g),* *Type(g,generalization).*

References

Avery MA, Alvim-Gaston M, Woolfrey JR (1999) Sythesis and structure-activity relationships of peroxidic antimalarials based on artemisinin. Adv Med Chem 4:125–217. doi:10.1016/S1067-5698(99)80005-4

Avery MA, Alvim-Gaston M, Rodrigues CR, Barreiro EJ, Cohen FE, Sabnis YA, Woolfrey JR (2002) Structure activity relationships of the antimalarial agent artemisinin: the development of predictive in vitro potency models using CoMFA and HQSAR methodologies. J Med Chem 45:292–303. doi:10.1021/jm0100234

Bakir GH, Hofmann T, Schölkopf B, Smola AJ, Taskar B, Vishwanathan SVN (2007) Predicting structured data. Neural information processing. The MIT Press, Cambridge, MA

Bayardo RJ (2002) The many roles of constraints in data mining: letter from the guest editor (special issue on constraints in data mining). SIGKDD Explorations 4(1):i–ii

Bernstein A, Provost F, Hill S (2005) Toward intelligent assistance for a data mining process: an ontology-based approach for cost-sensitive classification. IEEE Trans Knowl Data Eng 17(4):503–518. doi:10.1109/TKDE.2005.67

Blockeel H, DeRaedt L, Ramon J (1998) Top-down induction of clustering trees. In: Proceedings of the 15th international conference on machine learning, Morgan Kaufmann, pp 55–63

Brezany P, Janciak I, Tjoa AM (2007) Ontology-based construction of grid data mining workflows. In: Data mining with ontologies: implementations, findings and frameworks, IGI Global, pp 182–210. doi:10.4018/978-1-59904-618-1.ch010

Brinkman RR et al (2010) Modeling of biomedical experimental processes with OBI. J Biomed Semant 1(Suppl 1):S7. doi:10.1186/2041-1480-1-S1-S7

Button K, Deursen RW, Soldatova L, Spasić I (2013) TRAK ontology: defining standard care for the rehabilitation of knee conditions. J Biomed Inf 46(4):615–625. doi:10.1016/j.jbi.2013.04.009

Cannataro M, Comito C (2003) A data mining ontology for GRID programming. In: Proceedings of 1st international workshop on semantics in peer-to-peer and grid computing, pp 113–134

Caruana R (1997) Multitask learning. Mach Learn 28:41–75. doi:10.1023/A:1007379606734

Chapman P, et al. (1999) The CRISP-DM process model. Discussion paper. http://www.crisp-dm.org

Courtot M et al (2011) MIREOT: the minimum information to reference an external ontology term. Appl Ontol 6(1):23–33. doi:10.3233/AO-2011-0087

Demšar D et al (2006) Using multi-objective classification to model communities of soil. Ecol Model 191(1):131–143. doi:10.1016/j.ecolmodel.2005.08.017

Diamantini C, Potena D (2008) Semantic annotation and services for KDD tools sharing and reuse. In: ICDMW '08: proceedings of the 2008 IEEE ICDM workshops, IEEE computer society, pp 761–770. doi:10.1109/ICDMW.2008.43

Dietterich T et al (2008) Structured machine learning: the next ten years. Mach Learn 73:3–23. doi:10.1007/s10994-008-5079-1

Džeroski S (2007) Towards a general framework for data mining. In: KDID 2006—revised selected and invited papers, LNCS, vol 4747, Springer, pp 259–300. doi:10.1007/978-3-540-75549-4_16

Ford M, Philips L, Ste A (2004) Optimising the EVA descriptor for prediction of biological activity. Organ Biomol Chem 2:3301–3311. doi:10.1039/B410053K

Fox MS, Grüninger M (1994) Ontologies for enterprise integration. In: CoopIS, pp 82–89

Gangemi A, Guarino N, Masolo C, Oltramari A, Schneider L (2002) Sweetening ontologies with DOLCE. In: Proceedings of 13th international conference on knowledge engineering and knowledge management. Ontologies and the semantic web, pp 166–181. doi:10.1007/3-540-45810-7_18

Garcia J, Garcia-Penalvo FJ, Theron R (2010) A survey on ontology metrics. In: Communications in computer and information science, vol 111, Springer, Berlin, pp 22–27. doi:10.1007/978-3-642-16318-0_4

Golbraikh A, Tropsha A (2002) Beware of q^2!. J Mol Gr Mod 20:269–276. doi:10.1016/S1093-3263(01)00123-1

Grenon P, Smith B, Goldberg L (2004) Biodynamic ontology: applying BFO in the biomedical domain. In: Pisanelli D, (ed) Ontologies in medicine, vol 102. IOS, Amsterdam, pp 20–38. doi:10.3233/978-1-60750-945-5-20

Gruber T (2009) Ontology. In: Ling L, Tamer Özsu M (eds) The encyclopedia of database systems. Springer, pp 1963–1965. doi:10.1007/978-0-387-39940-9_1318

Grüninger M, Fox M (1995) Methodology for the design and evaluation of ontologies. In: IJCAI'95, workshop on basic ontological issues in knowledge sharing

Guha R, Jurs PC (2004) Development of QSAR models to predict and interpret the biological activity of artemisinin analogues. J Chem Inf Comput Sci 44:1440–1449. doi:10.1021/ci0499469
Guha R, Howard MT, Hutchison GR, Murray-Rust P, Rzepa H, Steinbeck C, Wegner J, Willighagen EL (2006) The blue obelisk-interoperability in chemical informatics. J Chem Inf Model 46(3):991–998. doi:10.1021/ci050400b
Hand DJ, Smyth P, Mannila H (2001) Principles of data mining. MIT Press, Cambridge, MA
Hilario M, Nguyen P, Do H, Woznica A, Kalousis A (2011) Ontology-based meta-mining of knowledge discovery workflows. In: Meta-learning in computational intelligence, studies in computational intelligence, vol 358, Springer, Berlin, pp 273–315. doi:10.1007/978-3-642-20980-2_9
ISO (2007) ISO/IEC 11404:2007—Information Technology—General-Purpose datatypes (GPD). Tech. rep, International Organization for Standardization
Karalic A, Bratko I (1997) First order regression. Mach Learn 26:147–176. doi:10.1023/A:1007365207130
Keet CM, Lawrynowicz A, d'Amato C, Hilario M (2013) Modeling issues and choices in the data mining optimisation ontology. In: 8th workshop on OWL: experiences and directions (OWLED-13), 26–27 May 2013, Montpellier
Kietz JU, F Serban AB, Fischer S (2010) Data mining workflow templates for intelligent discovery assistance and Auto-Experimentation. In: ECML/PKDD 2010 workshop on third generation data mining: towards service-oriented knowledge discovery (SoKD-10), pp 1–12
King RD, Muggleton SH, Srinivasan A, Sternberg MJ (1996) Structure-activity relationships derived by machine learning: the use of atoms and their bond connectivities to predict mutagenicity by inductive logic programming. Proc Natl Acad Sci 93(1):438–442. doi:10.1073/pnas.93.1.438
King RD et al (2009) The automation of science. Science 324(5923):85–89. doi:10.1126/science.1165620
Kocev D, Džeroski S, White M, Newell G, Griffioen P (2009) Using single and multi-target regression trees and ensembles to model a compound index of vegetation condition. Ecol Model 220(8):1159–1168. doi:10.1016/j.ecolmodel.2009.01.037
Kocev D, Vens C, Struyf J, Džeroski S (2013) Tree ensembles for predicting structured outputs. Pattern Recognit 46(3):817–833. doi:10.1016/j.patcog.2012.09.023
Kremen P, Sirin E (2008) SPARQL-DL implementation experience. In: Proceedings of the fourth OWLED workshop on OWL: experiences and directions volume 496 of CEUR workshop proceedings
Kriegel HP et al (2007) Future trends in data mining. Data Min Knowl Discov 15:87–97. doi:10.1007/s10618-007-0067-9
López MF, Gómez-Pérez A, Sierra JP, Sierra AP (1999) Building a chemical ontology using methontology and the ontology design environment. IEEE Intell Syst 14:37–46. doi:10.1109/5254.747904
Madjarov G, Kocev D, Gjorghevikj D, Džeroski S (2012) An extensive experimental comparison of methods for multi-label learning. Pattern Recognit 45(9):3084–3104. doi:10.1016/j.patcog.2012.03.004
Malone J, Parkinson H (2010) Reference and spplication ontologies. Ontogenesis. http://ontogenesis.knowledgeblog.org/295
Mannila H, Toivonen H (1997) Levelwise search and borders of theories in knowledge discovery. Data Min Knowl Discov 1(3):241–258. doi:10.1023/A:1009796218281
Mizoguchi R (2010) Yamato: yet another more advanced top-level ontology. http://www.ei.sanken.osaka-u.ac.jp/hozo/onto_library/YAMATO101216
Panov P (2012) A modular ontology of data mining. PhD thesis, Jožef Stefan Iternational Postgraduate School, Ljubljana, Slovenia
Panov P, Džeroski S, Soldatova LN (2008) OntoDM: an ontology of data mining. In: ICDMW '08: proceedings of the 2008 IEEE ICDM workshops. IEEE Computer Society, pp 752–760
Panov P, Soldatova L, Džeroski S (2010) Representing entities in the OntoDM data mining ontology. In: Inductive databases and constraint-based data mining, Springer, New York, pp 27–58. doi:10.1007/978-1-4419-7738-0_2
Panov P, Soldatova L, Džeroski S (2013) OntoDM-KDD: ontology for representing the knowledge discovery process. In: DS 2013, LNAI 8140, Springer, Berlin, pp 126–140. doi:10.1007/978-3-642-40897-7_9
Podpečan V, Zemenova M, Lavrač N (2012) Orange4WS environment for service-oriented data mining. Comput J 55(1):82–98. doi:10.1093/comjnl/bxr077
Qi D, King RD, Hopkins AL, Bickerton GRJ, Soldatova LN (2010) An ontology for description of drug discovery investigations. J Integr Bioinf 7(3):126. doi:10.2390/biecoll-jib-2010-126
Robinson P, Bauer S (2011) Introduction to bio-ontologies. Chapman & Hall, London
Serban F, Vanschoren J, Kietz J, Bernstein A (2013) A survey of intelligent assistants for data analysis. ACM Comput Surv 45(3):31.1–31.35. doi:10.1145/2480741.2480748

Silla C, Freitas A (2011) A survey of hierarchical classification across different application domains. Data Min Know Discov 22:31–72. doi:10.1007/s10618-010-0175-9
Sirin E, Parsia B (2007) SPARQL-DL: SPARQL query for OWL-DL. In: 3rd OWL experiences and directions workshop (OWLED-2007)
Slavkov I, Gjorgjioski V, Struyf J, Džeroski S (2010) Finding explained groups of time-course gene expression profiles with predictive clustering trees. Mol BioSyst 6:729–740. doi:10.1039/b913690h
Smith B et al (2005) Relations in biomedical ontologies. Genome Biol 6(5):R46. doi:10.1186/gb-2005-6-5-r46
Smith B et al (2007) The OBO foundry: coordinated evolution of ontologies to support biomedical data integration. Nat Biotech 25(11):1251–1255. doi:10.1038/nbt1346
Smith B, Ceusters W (2010) Ontological realism: a methodology for coordinated evolution of scientific ontologies. Appl Ontol 5(3–4):139–188. doi:10.3233/AO-2010-0079
Soldatova LN, Lord Ph, Sansone SA, Stephens SM, Shah NH (2010) Selected papers from the 12th annual bio-ontologies meeting. J Biomed Semant 1(Suppl 1):I1
Spjuth O, Willighagen EL, Guha R, Eklund M, Wikberg JES (2010) Towards interoperable and reproducible QSAR analyses: exchange of data sets. J Cheminf 2:5. doi:10.1186/1758-2946-2-5
Struyf J, Dzeroski S (2005) Constraint based induction of multi-objective regression trees. In: KDID 2005. Lecture notes in computer science, vol 3933, Springer, pp 222–233. doi:10.1007/11733492_13
Suarez-Figueroa M C, Gomez-Perez A, Motta E, Gangemi A (2012) The NeOn methodology for ontology engineering. In: Ontology engineering in a networked world, pp 9–34. doi:10.1007/978-3-642-24794-1_2
Sure Y, Staab S, Struder R (2009) Ontology engineering methodology. In: Staab S, Struder R (eds) Handbook on ontologies, 2nd edn. International Handbooks on Information Systems. Springer, Berlin, Heidelberg, pp 135–152. doi:10.1007/978-3-540-92673-3_6
Tropsha A (2010) Best practices for developing predictive QSAR models. Oral presentation. http://infochim.u-strasbg.fr/CS3_2010/OralPDF/Tropsha_CS3_2010
Tsoumakas G, Katakis I (2007) Multi label classification: an overview. Int J Data Wareh Min 3(3):1–13. doi:10.4018/978-1-60566-058-5.ch021
Uschold M, King M (1995) Towards a methodology for building ontologies. In: Workshop on basic ontological issues in knowledge sharing, held in conjunction with IJCAI-95
Vanschoren J, Blockeel H, Pfahringer B, Holmes G (2012) Experiment databases—-a new way to share, organize and learn from experiments. Mach Learn 87(2):127–158. doi:10.1007/s10994-011-5277-0
Vanschoren J, Soldatova L (2010) Exposé: an ontology for machine learning experimentation. Presentation at the Data Mining Jamboree, Ljubljana 2010. http://kt.ijs.si/janez_kranjc/dmo_jamboree/Expose
Vens C, Struyf J, Schietgat L, Džeroski S, Blockeel H (2008) Decision trees for hierarchical multi-label classification. Mach Learn 73(2):185–214. doi:10.1007/s10994-008-5077-3
Yang Q, Wu X (2006) 10 challenging problems in data mining research. Int J Inf Technol Decis Mak 5(4):597–604. doi:10.1142/S0219622006002258
Young D, Martin T, Venkatapathy R, Harten P (2008) Are the chemical structures in your QSAR correct? QSAR Comb Sci 27(11–12):1337–1345. doi:10.1002/qsar.200810084
Žáková M, Kremen P, Železný F, Lavrač N (2010) Automating knowledge discovery workflow composition through ontology-based planning. IEEE Trans Autom Sci Eng 8(2):253–264. doi:10.1109/TASE.2010.2070838
Ženko B, Džeroski S (2008) Learning classification rules for multiple target attributes. In: PAKDD. Lecture notes in computer science, vol 5012. Springer, pp 454–465. doi:10.1007/978-3-540-68125-0_40