Ontological Assistance for Knowledge Discovery in Databases Process

FILIPE MOTA PINTO
Computer Science Department of School of technology and Management
Polytechnic Institute of Leiria
Morro do Lena – Leiria
PORTUGAL
fpinto@estg.ipleiria.pt

MANUEL FILIPE SANTOS
Department of Information Systems of Engineering School
University of Minho
Campus de Azurém – Guimarães
PORTUGAL
mfs@dsi.uminho.pt

Abstract: - The dramatically explosion of data and the growing number of different data sources are exposing researchers to a new challenge - how to acquire, maintain and share knowledge from large databases in the context of rapidly applied and evolving research. This paper describes a research of an ontological approach for leveraging the semantic content of ontologies to improve knowledge discovery in databases. We analyze how ontologies and knowledge discovery process may interoperate and present our efforts to bridge the two fields, knowledge discovery in databases and ontology learning for successful database usage projects.

Key-Words: - Ontologies, Knowledge Discovery, Databases, Data Mining, PMML, Web Semantic.

1 Introduction

In artificial intelligence, ontology is defined as a specification of a conceptualization [10]. Ontology specifies at a higher level, the classes of concepts that are relevant to the domain and the relations that exist between these classes. Indeed, ontology captures the intrinsic conceptual structure of a domain. For any given domain, its ontology forms the heart of the knowledge representation.

In spite of ontology-engineering tools development and maturity, ontology integration in knowledge discovery projects remains almost unrelated.

Knowledge Discovery in Databases (KDD) process is comprised of different phases, such as data selection, preparation, transformation or modeling. Each one of these phases in the life cycle might benefit from an ontology-driven approach which leverages the semantic power of ontologies in order to fully improve the entire process [8].

Our challenge is to combine ontological engineering and KDD process in order to improve it. Ontologies can help the KDD process introducing a new semantic layer to the process and moving it from a data driven approach to a knowledge driven approach [9]. Therefore, this document describes a research on a new approach to leveraging the semantic content of ontologies to improve KDD.

This paper is organized as follows: after this introductory part we present related background concepts. Then, we present related work on this area following the presentation and discussion of ontological assistance. The main contribution is presented in terms of a system prototype description and also system operation sections. Finally we draw some conclusions and address further research based on this research to future KDD data environment projects.

2. Background

2.1 Knowledge Discovery in Databases

Knowledge discovery in databases (KDD) is the result of an exploratory process in order to achieve domain defined objectives involving the application of various algorithmic procedures for manipulating data, building models from data, and manipulating the models. The Data Mining phase deserves more attention from the research community: processes comprise multiple algorithmic components, which interact in nontrivial ways..

We consider tools that will help data analysts to navigate the space of KDD processes systematically, and more effectively. In particular, this paper focuses on a subset of stages of the KDD —those stages for which there are multiple algorithmic components that can apply.

For most of this paper, we consider a prototypical KDD process template, similar to the one represented in Figure 1.
The sequence of KDD phases is not strict. Moving back and forth between different phases is always required. It depends on the outcome of each phase, which one, or which particular task of a phase has to be performed next.

We focus our attention on the three main macro components of KDD life cycle: data understanding (data selection); data pre processing (all related data preparation and transformation activities), and modeling (data mining and the application of induction algorithms). We have chosen this set of components because, individually, they are relatively well understood—and they can be applied to a wide variety of benchmark data sets.

2.2 Predictive Model Markup Language

Predictive model markup language (PMML) is an XML-based language that provides a way for applications to define statistical and data mining models and to share these models between PMML compliant applications (Data Mining Group). Furthermore, the language can describe some of the operations required for cleaning and transforming input data prior to modeling. Since PMML is an XML based standard, its specification comes in the form of an XML schema that defines language primitives as follows [3]:

- Data Dictionary;
- Mining schema;
- Transformations.
- Model statistics;
- Data mining model.

2.3 Ontology Web Language

Ontologies are used to capture knowledge about some domain of interest. Ontology describes the concepts in the domain and also the relationships that hold between those concepts. Different ontology languages provide different facilities. Ontology Web Language (OWL) is a standard ontology language from the World Wide Web Consortium (W3C). An OWL ontology consists of Individuals (represent domain objects), Properties (binary relations on individuals - i.e. properties link two individuals together), and Classes (interpreted as sets that contain individuals).

Web ontology language for services (OWL-S) [12] consists of several interrelated OWL ontologies that provide a set of well defined terms for use in service applications. OWL-S uses OWL to define a set of classes and their properties specific to the description of Web services. The class Service is at the top of this ontology, which provides three essential types of knowledge about a service represented as classes:

- ServiceProfile;
- ServiceModel;
- ServiceGrounding;

Moreover, OWL-S enables inclusion of some expressions to represent logical formulas in Semantic Web rule language (SWRL) [11]. SWRL is a rule language that combines OWL with the rule markup language providing a rule language compatible with OWL.

2.4 Semantic Web Language Rule

At the best of our knowledge there are no standard OWL-based query languages. Several RDF-based query languages exist but they do not capture the full semantic richness of OWL. To tackle this problem, it was developed a set of built-in libraries for Semantic Web Rule Language (SWRL) that allow it to be used as a query language.

The OWL is a very useful means for capturing the basic classes and properties relevant to a domain. However, these domain ontologies establish a language of discourse for eliciting more complex domain knowledge from subject specialists. Due to the nature of OWL, these more complex knowledge structures are either not easily represented in OWL or, in many cases, are not representable in OWL at all. The classic example of such a case is the relationship uncleOf \((X, Y)\). This relation, and many others like it, requires the ability to constrain the value of a property \((brotherOf)\) of one term \((X)\) to be the value of a property \((childOf)\) of the other term \((Y)\); in other words, the siblingOf property applied to \(X\) (i.e., \(brotherOf(X, Z)\)) must produce a result \(Z\) that is also a value of the childOf property when applied to \(Y\) (i.e., \(childOf(Y, Z)\)). This “joining” of relations is outside of the representation power of OWL.

![Figure 1: Knowledge discovery process framework adapted from [5]](image-url)
One way to represent knowledge requiring joins of this sort is through the use of the implication (\(\rightarrow\)) and conjunction (AND) operators found in rule-based languages (e.g., SWRL). The rule for the \textit{uncleOf} relationship appears as follows:

\[
\text{brotherOf}(X, Z) \land \text{childOf}(Y, Z) \rightarrow \text{uncleOf}(X, Y)
\]

3. Related work

A KDD assistance through ontologies should provide user with nontrivial, personalized “catalogs” of valid KDD-processes, tailored to their task at hand, and helps them to choose among these processes in order to analyze their data.

In spite of the increase investigation in the integration of domain knowledge, by means of ontologies and KDD, most approaches focus mainly in the DM phase of the KDD process [1] [2] [4] while apparently the role of ontologies in other phases of the KDD has been relegated.

Currently there are others approaches being investigated in the ontology and KDD integration, like ONTO4KDD\(^1\) or AXIS\(^2\). Both of them are focusing the application of ontologies in order to improve overall KDD process regarding DM models optimization and sophistication.

In the literature there are several knowledge discovery life cycles, mostly reflect the background of their proponent’s community, such as database, artificial intelligence, decision support, or information systems [7]. Although scientific community is addressing ontologies and KDD improvement, at the best of our knowledge, there isn’t at the moment any fully successful integration of them.

This research encompasses an overall perspective, from business to knowledge acquisition and evaluation. To this end we use the Data Mining Ontology (DMO), integrated in KDD process to propose a general framework. Moreover, this research focuses the KDD process regarding the best fit modeling strategy selection supported by ontology.

Therefore, at this work we focus the role of ontology in order to assist the KDD in different stages of the process: data understand; data preparation and modeling. Indeed, to select the appropriate an adequate tasks sequence to support the KDD work becomes an important decision. This work proposes a computational model based on ontologies to assist the KDD planning process.

4. Ontological Assistance

To achieve the goals presented in the introductory section, we have designed a specialized tool that fulfils the role of the KDD shown in Figure 1. For simplicity reasons, trough Eclipse software\(^3\) it is implemented an application able to navigate a user in the phases DBM process and KDD.

Our main is to assist the user to carry out the KDD process. Our solution provides support to choose particular knowledge extraction objectives and manage the entire process, from data to output models evaluation.

The system dynamically modifies the task set composition depending on knowledge extraction objectives, entered data, defined preconditions and effects, and existing description of services available in the knowledge base:

\[
\text{getObjectiveTypeData}[(\text{Database}(?db) \land \text{hasDataType}(?db,?dt1,…,?dtn)) \land (\text{ObjectiveType}(?objT))) \rightarrow \text{DataSet}(?ds)
\]

This SWRL get expression gets the set of data that have datatypes (\(dt1, dt2,..., dtm\)). Where database corresponds to all data available to be used with KDD. \textit{ObjectiveType} is the general objective to be achieved at the final of KDD process. \textit{hasDataType}, is a OWL property that links the objective type and data types previously used and registered at knowledge base. \textit{DataSet} is the expression output with workable set of data. Each variable is preceded by a question mark.

5. System Prototype

A general overview of the main components of the system is shown in Figure 2. Our system has four main components:

- **Knowledge base**: developed over Protégé\(^4\) OWL editor it is used to create and maintain the ontology. Protégé stores information the OWL format file. The knowledge base is formed by

---

\(^1\) [http://olp.dfki.de/pkdd04/cfp.htm](http://olp.dfki.de/pkdd04/cfp.htm)


\(^3\) [www.eclipse.org](http://www.eclipse.org)

\(^4\) [http://Protege.stanford.edu](http://Protege.stanford.edu)
two main components: domain knowledge ontology and DMO ontology – here we have reused an ontology for modeling purposes: Data Mining Ontology [14];

- **Rule engine bridge**: performs inference tasks through SWRL knowledge base. It extracts SWRL rules and relevant OWL knowledge, using the rule engine and system knowledge base. To infer about knowledge in the knowledge base, we build SWRL expressions to perform queries over the knowledge base and invoke the Pellet reasoner [15]. We need implement engine or map to the existing rule engine, here the bridge;

- **System rule engine**: is based on SWRL API supported by Jena Toolkit [13] and is able to interact with a user to assemble the required information. Jena is a Java framework for building Semantic Web applications. It provides a programming environment for RDF, RDFS, OWL and SPARQL and includes a rule based inference engine. Jena is available to Protégé through an API – JessTab [6];

- **GUI**: developed through Eclipse java software to develop it supports the system user interface.

Keeping it straightforward, the assistant system communicates over the rule engine bridge with the Pellet reasoner, which is able to answer a subset of SWRL/SPARQL queries [16]. Also the inference system queries knowledge base every time it needs to enumerate some parameters or find a DM task, algorithm, service, and so forth. Moreover, the DBMI system also updates the knowledge base with instances of DMO classes and values of their properties.

6. General system operation

Our system prototype operation follows general KDD framework (Figure 1) and uses the DMO ontology to infer at each user interaction. To carry out this we have developed an initial set of SWRL rules. Since KDD is an interactive process, these rules deal at both levels: user and ontological levels. The logic captured by these rules is this section using an abstract SWRL representation, in which variables are prefaced with question marks.

6.1. Objectives definition

At user level, our system uses the ontology to assist at objective type selection. This task is performed throughout the following SWRL code:

```
Objective(?obj)-> query:user input
ObjectiveType(?objT)
```

6.2. Data understanding and selection

Data understanding stands for data description and evaluation. Prior to be used at KDD process, each attribute need to be evaluated by a set of analysis tasks, e.g., data completeness (missing values); data description (e.g., range values, units, granularity), among others.

Prior working data selection it is required a previous mining model selection. Therefore, the system will carry a `sqwrl:orderBy(?met)` instruction in order to present at user choice a performance ordered list knowledge base of available modeling techniques (?met).

**DM-TASK**(?db, ?objT)^

```
(sqwrl:orderBy(?met))
```

`swrl:selecthasModeling(?met)`

Where, DM-TASK is a DMO class which has `hasModeling` property. To a given initial database (`db`) and objective type (`objT`), the SQWRL code used will prompt the user for one of the listed modeling techniques (`met`). Thereafter the system proceed with data set selection.

```
hasDatabase(?db)^ hasObjectiveType(?objT)^ hasModeling(?met)^
```

```
swrl:select
hasDataSelectionTask(?wd)
```

The system is able to perform the data property `hasDataSelectionTask` perform produce as output the selected working data (`wd`).

6.3. Data Preprocessing

Since we have selected working data (`wd`) we need to proceed with its preparation regarding algorithm’s data format requirements:

```
hasModeling(?met)^ hasWorkingData(?wd)
```

```
->sqwrl:select
DataPreProcessingTask(?dpp)
```

`DataPreProcessingTask` is a data property that selects and displays the data preprocessing task to be performed over the working data (`wd`).

6.2.4 Modeling

In order to optimize efforts we have re-used a data mining ontology (DMO) [14], which has similar knowledge base taxonomy. Here we take advantage of an explicit ontology of data mining techniques and standards (as presented in the above sections) using
the OWL concepts to describe an abstract semantic service for DM and its main operations.

The DMO uses a service named Abstract Data Mining Service that simplifies its architecture as the realization of the OWL service with a detailed description of its profile and model. DMO has three essential types of data mining components involved in the assisted KDD process: DM-element, DM-task and DM-service.

The DMO is built through the description of the DM-tasks, DM-elements and involved DM-services. The DM-elements are represented by OWL classes together with variations of their representations in XML (allowing information interchange with PMML DM models). It means that a concept described by an OWL class can have one or more related XML schemas that define its concrete representation in XML.

In the DMO, for simplicity reasons, there are two defined types of DM-elements: settings and results. The settings represent inputs for the DM-tasks, and on the other hand, the results represent outputs produced by these tasks. There is no difference between inputs and outputs because it is obvious that an output from one process can be used, at the same time, as an input for another process.

The settings are built through enumeration of properties of the DM algorithms and characterization of their input parameters. Based on the concrete Java interfaces, as presented in the Weka software API [17] and JDM API, it was constructed a set of OWL classes and their instances that handle input parameters of the algorithms and their default values.

The ontology describes available DM tasks, methods, algorithms, their inputs and results they produce.

All these concepts are not strictly separated but are rather used in conjunction forming a consistent ontology

\[
\text{DataSet}(\text{?ds}) \quad \text{hasObjectiveType}(\text{?eot}) \\
\text{ModelSelection}(\text{?ds}, \text{?eot}) \rightarrow \text{AlgorithmSelector}(\text{?alg}) \\
\text{hasModeling}(\text{?met}) \\
\text{hasData}(\text{?dpp}) \\
\text{hasAlgorithm}(\text{?alg}) \rightarrow \text{DM-TASK}(\text{?m})
\]

6.5. Deployment

Each running KDD process must be evaluated according to the results, in order to be updated and reused in latter projects.

Firstly we need to perform a model evaluation, through an evaluation method. Such evaluation will depend of which type of model we have, which algorithm was used and of course, which model do we have. Then evaluate it through evaluation algorithms available, e.g., AUC (area under curve) or PCC (principal components analysis):

\[
\text{getEvaluation}(\text{DM-TASK}(\text{?m})) \\
\text{hasModeling}(\text{?met}) \\
\text{hasAlgorithm}(\text{?alg}) \\
\text{hasEvaluation}(\text{?m}, \text{?met}, \text{?alg}) ightarrow \text{Evaluation}(\text{?m}, \text{?ev})
\]

Once performed the evaluation, the system automatically updates the knowledge base with a new record. The registered information will serve for future use – knowledge sharing and reuse:

\[
\text{Model}(\text{?m}) \\
\text{ObjectiveType}(\text{?objT}) \\
\text{hasData}(\text{?dpp}) \\
\text{Evaluation}(\text{?ev}) ightarrow \text{KnowledgeBaseIndex}(\text{?m}, \text{?objT}, \text{?dpp}, \text{?ev})
\]

4 Conclusions and further research

We use general domain ontology to assist its knowledge extraction from databases with KDD process.

This research focuses the KDD development assisted by ontologies. Moreover we use ontologies to simplify and structure the development of knowledge discovery applications offering to a domain expert a reference model for the different kind of DM tasks, methodologies to solve a given problem, and helping to find the appropriate solution.

There are four main operations of KDD that can take advantage of domain knowledge embedded in ontologies:

i. During the data preparation phase, ontology can facilitate the integration of heterogeneous data and guide the selection of relevant data to be mined;

ii. During the mining step, domain knowledge allows the specification of constraints for guiding DM algorithms by, e.g. narrowing the search space;

iii. During the deployment phase, domain knowledge helps experts to validate extracted units and ranking them.

With knowledge base ontology may help analyst to choose the best modeling approach based on knowledge base ranking index.

Future work will be devoted to expand the use of KDD ontology through knowledge base population with more relevant concepts about the process. Another interesting direction to investigate is to represent the whole knowledge base in order to allow its automatic reuse.
References:


