TECHNICAL REPORT

RT - XX XX/ XX

MF-Ontology: an Ontology for the Text Mining Domain

Daniel de Oliveira
(danielc@cos.ufrj.br)

Fernanda Baião
(fernanda.baiao@uniriotec.br)

Marta Mattoso
(marta@cos.ufrj.br)

Programa de Engenharia de Sistemas e Computação

COPPE/UFRJ

Rio de Janeiro, January 2009

January 22, 2009

Text mining (TM) has emerged as a definitive technique for knowledge acquisition from text. The TM process is based on several phases that prepare the text for mining, process the text, and analyze the results. Effective and efficient use of the combination of TM algorithms and techniques is a challenge. Most of the research is focused on developing new data structures, algorithms and methods to achieve that. However, the TM process is still lacking of modeling support. The TM analyst faces many options when modeling a TM process. For instance, the analyst needs to choose the most effective solution to extract the desired knowledge. This is a complex decision involving choices for each one of the TM process phases where many algorithms and implementations are available for composition and several parameters must be tuned. This scenario tends to be chaotic and each time a new modeling starts, all this ad-hoc process is repeated. A first step towards this modeling is to add semantics to the TM process and register modeling results. The use of ontologies to describe the TM domain can help to structure the systematic composition of algorithms and techniques of the text mining process. By adopting the same structure, similar modeling can be identified and reuse of TM software components (web services, local applications) is facilitated. In this paper we describe the MF-Ontology, an ontology for the modeling of activity flow tailored to the TM domain. MF-Ontology that can be used to simplify the development of knowledge discovery applications based on texts. It represents a reference model to the different phases of text mining tasks, methodologies and software available in order to solve a problem. Thus, MF-Ontology offers semantic help for the TM analyst in finding the most appropriate solution. We describe the design of the MF-Ontology and analyze its different levels of abstraction to semantically represent the TM process. We also present an evaluation of MF-Ontology and show techniques for revising the ontology concepts based on interviews with specialists.

1. Introduction

The fast development of computational infra-structures [11, 12, 13] and the effective use of text mining tools (as well the continuous development of new text

mining techniques) to extract useful knowledge from a huge amount of data (semi-structured data) have increased the necessity to execute in-silico experiments in order to model complete text mining processes. These kinds of experiments can be usually modeled as scientific workflows, and can be enacted using workflow engines. There is a huge variety of workflows engines available on the web, such as Taverna of the myGrid project [30, 31, 37, 38] and Kepler [1, 2, 23, 24], which exploits grid and web services technologies to efficiently support applications of many domains (in case of myGrid, the focus is on bioinformatics applications) .

However, the development of formal data models to represent these experiments and their associate data is still characterized by the lack of semantic help (for modeling and data analysis). For example, in order to compose web services as a complete experiment, the text mining analyst must previously know which service is related to which step of the text mining process. In addition, the text mining analyst must know the compatibility among all the web services (or grid services) when modeling his experiment. Depending on the amount of available web services (or any computational component) it could be a hard task to search for equivalent or compatible services.

Another problem is relates to the nomenclature and the associated data and its relation to the experiment. The experiments could be annotated with free-text describing the techniques used, the methodology and so on. These kinds of annotations are essential for a more complete (post) analysis of the experiment, and also provide a way to reuse them in future text mining processes that will be modeled.

Several formats and formalisms have been used over the years to construct annotation databases. Free-text is still the most used formalism. The most important advantage of this approach is its expressiveness. The text mining analyst is able to express whatever he wants with this approach. However, the use of free-text limits search capabilities and automatic comparisons. A simple alternative to free-text would be controlled vocabulary. Nevertheless, this approach will reduce the expressiveness. The most adequate option is to use domain ontologies coupled with a tool that allows the construction of data models and their association with the ontologies.

Different domains of knowledge consider different definitions for ontologies [10, 14, 15, 16, 17]. The word ontology has been defined by philosophy for a long time and it is related to the subject of existence. Gruber [15] defines ontology as "a specification of a conceptualization". But specifically for Computer Science, the focus is on what ontology is designed for. In most cases ontologies are modeled in order to provide ways for sharing and reuse of knowledge. In addition, Guarino [17] proposes an ontology classification in four main categories: High-level Ontologies, Domain Ontologies, Task Ontologies and Application Ontologies. We focus on this paper on the development of domain ontology. In general, ontology can be defined as a declarative model of a domain that represents the concepts of that domain, their attributes and the relationships between them.

The outline of the paper is as follows. In the next section, we briefly introduce the text mining process and it particularities. In Section 3, we show related work and present the conceptual ideas of the MF-Ontology. In Section 4, we demonstrate the evaluation criteria for MF-Ontology and the techniques for achieving it. Finally, we discuss some future work and conclude in Section 5.

2. Text Mining Process

Text Mining aims to extract knowledge from texts in documents. Text Mining process can be modeled by composing several tasks necessary for the Text Mining cycle [8, 9]. Txt Mining cycle is composed of three main phases: preprocessing, mining, and visualization [19].

Text Mining is a very active research area and innumerous algorithms have been designed for these three phases [3, 20, 22, 28, 36, 40]. Several programs are also available, including open-source ones [26, 27, 41]. There are repositories of TM codes such as Weka [41] and RapidMiner (as previously known as YALE) [26, 27]. In addition, Text Mining analysts build their own codes through variants of well-known algorithms or variants for parallel and distributed computing.

This scenario poses some challenges observed in other scientific laboratories such as bioinformatics [30, 31, 42]. The designer is faced with several alternative software to be used and evaluated in each phase of the Text Mining cycle. These alternatives are usually neither organized nor structured. Once the software is chosen, the designer is concerned with parameters definition, data sets to test the software upon, among other design decisions. Such decisions can be complex depending on the implementation of the algorithms. Many software systems perform more than one activity of the Text Mining life-cycle and the covered activities (by the implementation) are often not documented. Thus, it is a hard task to know which programs/services cover which activities of the Text Mining cycle.

Once everything is set, it is time to run those programs. Usually they are executed isolated one from another. The development of a Text Mining experiment, that reflects a complete Text Mining cycle, may involve different platforms associated to the various steps of an experiment of this nature, such as preprocessing data, removing stop words in a parallel cluster machine, stemming, text mining analysis in a different cluster and visualization of the results using a visualization environment.

Some of these issues have been addressed by scientific workflow management systems (WfMS). The Text Mining steps can benefit from such a system where the steps would be linked and the data associated along the workflow definition. However, most of these systems [23, 32, 42] are concerned with the execution of the workflow, thus focusing on efficiency, scheduling, fault-tolerance, data movement, and so on. Some of them include a workflow editor to help on the composition of the tasks, but it is also an operational tool. WfMS with semantics are focused on a specific scientific domain.

Once the workflow definition is done, the Text Mining designer experiments several variations of this workflow through fine tuning the parameters. However, after such several executions it is difficult to relate all these executions to the "same" workflow. For example, cross-validation [9, 19] is a very common technique applied in the Text Mining domain. This means that the same workflow will be executed many times, varying the value of the input parameters in order to analyze the

variation of the results. Thus, it is very important to the Text Mining specialist to be able to relate all these executions to the "same" designed workflow. All WfMS analyzed do not support that, since the workflows are executed "isolated" one from another.

A Text Mining environment has some specific characteristics, and thus presents specific requirements that are not available in current solutions. An example of one of those characteristics is the need for one generic workflow definition reflecting the entire Text Mining cycle, that is, a workflow that applies to almost all experiments. This characteristic motivated the development of well known tools in the area, such as Weka [41] and RapidMiner [26, 27]. However, in their current stage, they may be seen as just a repository of programs with some semantic on them and no workflow support.

As we can observe, available WfMS do not support all TM needs. Some systems do offer some provenance support, but they are either disconnected from the WfMS [24, 42] or they are specific to an application domain, such as bioinformatics in Taverna [38].

Based on these facts, our proposal is to couple a domain ontology (MF-Ontology) to a WfMS environment. The goal of MF-Ontology is to offer semantic support for Text Mining resource discovery, experiment design (workflow composition), modeling experiments, execution and visualization. In the next sections we present a Text Mining Ontology (MF-Ontology) extended from [4] to describe concepts and relationships between key aspects of the Text Mining domain.

3. MF-Ontology: an Ontology for the Text Mining Process

This section presents in details the development of MF-Ontology, an ontology for Text Mining domain. This section is focused on detail about the abstract specification of the ontology and its implementation in concrete artifact in OWL [25, 33], a W3C recommended language for ontology representation. This section briefly explains about languages for representing ontologies and related work.

3.1. General-purpose languages for information representation

To be considered concrete artifacts and then be used in computational components, domain ontologies need to be represented in an ontology representation language. However, as occurs in other research areas in computer science, there is no rigid standard for ontologies representation. Many ontology languages have been proposed in the last few years, most of them based on XML [7], creating specific tags. However, a few of these languages are recommended by W3C. In the following paragraphs, we briefly talk about these languages, before explain why we have chosen OWL [25, 33] as our representation language.

RDF or Resource Description Framework Schema Language [35] is one of the many kinds of languages for representing information in the Web. This language is mainly based on the idea of creating statements in the form of subject-predicate-object expressions, called triples in the terminology of RDF language. The subject is strictly related to the resource we are dealing with, and the predicate denotes aspects of the resource and represents a relationship between the subject and the object.

OIL or Ontology Interchange Language proposed by Fensel [10] is a web-based representation and inference layer for ontologies and it is compatible with RDF Schema (RDFS), and includes a precise semantics for describing term meanings. OIL presents a layered approach to a standard ontology language.

DAML+OIL (or DARPA) is an ontology representation language proposed by Horrocks et al. [21]. DAML+OIL is an extension of OIL language. This language is being developed as an extension to XML and the RDF. It provides a rich set of constructs with which to create ontologies and to markup information so that it is machine readable and understandable.

OWL or Web Ontology Language [25, 33] is an ontology representation language recommended by W3C that is based on OIL and DAML+OIL. These kinds of ontology representation language have been carefully designed to provide both expressiveness and computational efficiency as stressed by Guizzardi [18]. OWL

facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics. OWL has three derived sub-languages: OWL Lite, OWL DL and OWL Full. Each one of these variations has different expressiveness.

Along with the advantages stressed on the previous paragraph, the OWL was our choice for this work, since it is recommended by the W3C and has a huge number of applications available for handling this kind of language, including the most comprehensive among those tested, protégé [34], as will be detailed in the following sub-section.

3.2. Ontologies for Data Mining

In all literature there are some proposals of ontologies focused on the of data mining process as [4, 5, 6]. Meanwhile, was not found in the literature any ontology describing the process of text mining. Our approach then was to search for data mining ontologies that could be used as solid foundation of our text mining ontology.

In this subsection we show and explain about some of data mining ontologies that were found in the literature and which were the basis for the construction of our text mining ontology that we propose follow in this work.

WekaOntology

This ontology for data mining was proposed by Cannataro and Veltri [6] to be used in the MS Analyzer (Mass Spectra Analyzer) tool proposed in the same work.

The MS-Analyzer is an application to analyze mass spectral, or a large body of data on measures (intensity, for example) that represent the abundance of biomolecules that have some mass. This application has an editor of workflows based on ontologies using an ontology called WekaOntology.

The WekaOntology is an ontology that models the tools for the data mining computational package called Weka [41] enriched with the description of concepts of

algorithms for pre-processing of mass spectral (main purpose of the application MS-Analyzer) and ProtOntology, that models concepts, methods, algorithms, tools, and databases relevant to the proteomic field and provides base to date analysis in the future. Unfortunately this ontology was not available for consultation, which made it impossible to use them to me as a basis for work or as a reference for it.

DAMON

This data mining ontology was proposed by Cannataro and Comito [4] as a solution to offer semantic support to the user when his data mining workflow was being defined within the Knowledge Grid [5]

This work proposes an ontology for data mining tasks, not including the phases of pre-processing, or post-processing. This work proposes to model the data mining process around four key-concepts, as follows:

1. **Task**: represents the data mining tasks available for the user.
2. **Method**: represents the available data mining methods (for example: a decision tree).
3. **Algorithm**: represents the proposals of algorithms that are based on methods already modeled.
4. **Software**: represents the implementations of certain algorithms.

DAMON was used as the basis for our ontology proposal for text mining as it was available for analysis and modeled a general process of data mining, without incorporating particular characteristics of any domain (such as WekaOntology incorporates the concepts of mass spectral ontology).

3.3. Development of MF-Ontology

The development of the mentioned ontology (MF-Ontology) was made entirely manually. These concepts were created and inserted through literature research in the text mining area, especially to Feldman and Sanger [9] and the specialists in the text

mining field. Thus, there was no tool used for extraction of concepts for inserting the data into the ontology, the only tool used was Protégé [34] for modeling and of data (creation of ontology instances).

During the development stage, the methodology used as a basis for modeling ontologies proposed by Noy and McGuiness [29] (consisting of questions and activities to be implemented during the construction of an ontology) was used to extend and adapt the proposal ontology DAMON by Cannataro and Comito [4], generating a new ontology that takes account the text mining process requirements and covers the text mining field.

The **first step** of the ontology development is the **definition of the field that this proposal ontology is intended to cover**: in our case the field is the text mining domain. In addition, we need to know what types of questions we should be able to answer using the developed ontology:

1. **Identify equivalent activities in the process**: mapping the text mining process by means of ontology concepts, we can identify which activity of the process defined is equivalent to another, just checking if each one refers to the same concept in ontology. For example: If an activity is designated to perform a function of pre-processing within the workflow and the user does not know which third-part service to choose to execute this activity, the ontology must be able to offer a list of services that are available to run the activity which was determined by the concept involved.

2. **Identify data types**: through inference on the ontology, we should be able to identify which data type is related to each software component input and output (it could be a service, local program, service on the grid, and so on), this way we can run a check type consistency validation of data types in the definition of our text mining process, thus avoiding more common errors. For example, if an activity for pre-processing generates a text file as output and the input of service following read the

data as an excel spreadsheet, these services are incompatible and ontology must be able to examine this type of problem, avoiding future problems.

3. **Identify the sequence of activities within the modeled process**: the ontology must be able to identify whether an activity depends on another to be defined in the process. For example, if the user defines a text mining activity will be executed before the activity of pre-processing, the ontology must be able to verify that the order of execution of the activities does not make sense (hence pre-processing activity always comes before and text mining activity).

4. **Relate the generated data with ontology concepts**: through the use of ontology to support the text mining process, we can also associate their concepts to the generated data that are stored after the executions of process activities. Thus, the data stored will have semantics associated. For instance, the result data of a pre-processing activity called "Stop Word Removal" is a flat file. Using ontologies to model the entire process, we can associate concepts do the generated data, associating the flat file to a data type (modeled in the ontology) and to the activity that generated it ("Stop Word Removal" that is a pre-processing activity).

The **second step** of the methodology is to define what are the key concepts and the properties of the ontology. We have used a top-down approach, where more general concepts of the ontology have been defined and then specialized.

Based on the definition of the text mining process, we have extracted and defined the main key concepts: `TM Step`, `Task`, `Function`, `Algorithm`, `Method`, `Measure`, `Software` and `Data Type`.

A `TM Step` represents the macro-activities of the text mining life cycle, and is divided into: `Pre-processing`, `text mining` and `Post-Processing`.

A `Task` is a technique of Text Mining that is used to extract patterns and models of a set of unstructured data. In other words, the text mining tasks are the main goal of the life cycle of an text mining process.

A `Function` is a technique of preparing the data for the text mining activity or for evaluating models generated. Thus the functions were divided into `pre-processing functions` and `post-processing functions` in accordance with its objective.

An `algorithm` is how a `Task` or a `Function` is performed.

A `Method` is a methodology used as a basis for an algorithm to discover knowledge in texts. Different methods are used on the process for different purposes.

A `Measure` is a representation of how an algorithm of the text mining will perform. Each text mining algorithm can be performed with several different measures of distance, for example. The choice of the measure to the algorithm may cause impact on the final result.

A `Software` is the implementation of a particular `algorithm`. This is an important concept to our proposal, as one of the uses of the MF-Ontology will be guiding the user during the definition process of a text mining workflow. The computational environment (SGWf) in what the workflow definition is manipulated essentially deal with logical sequences of software for text mining. In this context, it is essential that the MF-Ontology can guide the environment to answer the following questions: What text mining task, or function of pre / post-processing, a certain `software` runs? What kind of methodology is used by a certain `software`? What types of data does this `software` handle with? What algorithm does this `software` implement?

A `Data Type` is the semantic representation of an input or output of any `Software`.

Once we have defined the base concepts of the ontology, we must define what are the relationships and properties of these concepts. For example: the property `Available` of `software` indicates whether it is available for use or not. Other examples we can cite are: the relationship `Execute Task` defines the relation between `Algorithm` and `Text Mining Task`, and the `Preceeds` relationship that indicates which `TM Step` preceeds another in a logical order. For example, in the proposed ontology, `Pre-processing` should always preceeds `text mining`.

The **third step** of the ontology development process is to define the hierarchy of concepts by means of taxonomies. Taxonomies are used to organize ontological concepts in a Hierarchical way and is composed by two types of relationships:

1. "*is-a*": this kind of relationship generalizes or specializes a concept in the ontology. For instance: `Classification` "is-a" `Text Mining Task`.

2. "*part-of*": this kind of relationship defines partitions of a class. For example: `Text Mining` is "part-of" `TM Step`.

The ontology was built using a variety of small specialized taxonomies and interrelated. Display below some of taxonomies (part of them) that were defined in the construction process of the text mining ontology.

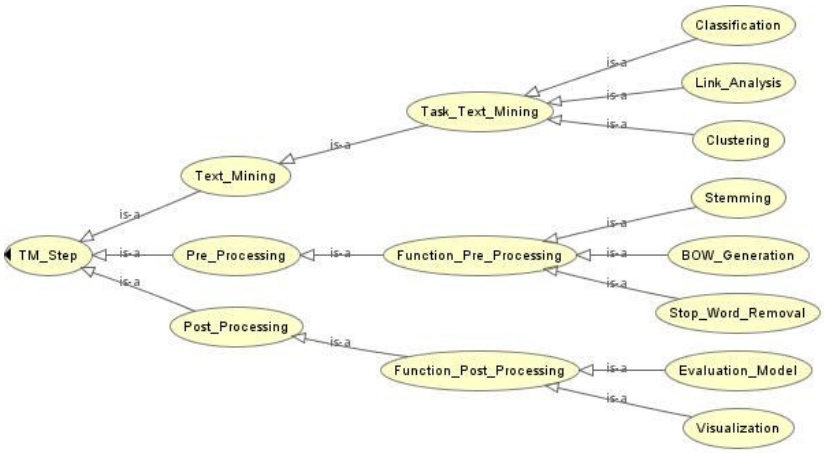

Figure 1: The TM Step taxonomy

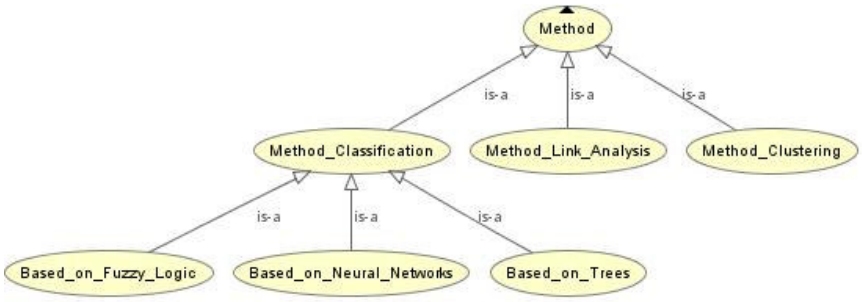

Figure 2: The Method taxonomy

The **fourth step** of the construction of ontology is to define the axioms (statements that are always true) to link the taxonomies that have been raised to the concepts of `TM Step`, `Method`, `Algorithm`, `Software`, `Measure` and `Data Type`. In the text mining field, we use axioms to represent statements as represented below:

- A `Software` ALWAYS implements at least one `algorithm`.
- An `algorithm` ALWAYS uses at least one `method`.
- If a `Software` implements an `algorithm`, and this `software` is a `classification software`, it must implement an `Classification Algorithm` (restriction of integrity of ontology).
- A `Method` ALWAYS specifies a `task` or a `function`.

Thus, we define relations between the taxonomies, creating a network of concepts that can now be used to fulfill the needs previously raised. Figure xx shows us part of the developed ontology. In this figure we show the ontology referring only to fictional `classification software` X, which is based on the `algorithm` X1, which uses the `method based on trees`.

It is important to emphasize that the MF-Ontology, on its present stage, covers only the sub-field of algorithms, software and methods of text mining that are applied in

geometric space. Other sub-fields of text mining, for example where the algorithms or are based experiments in the space of sets are not covered.

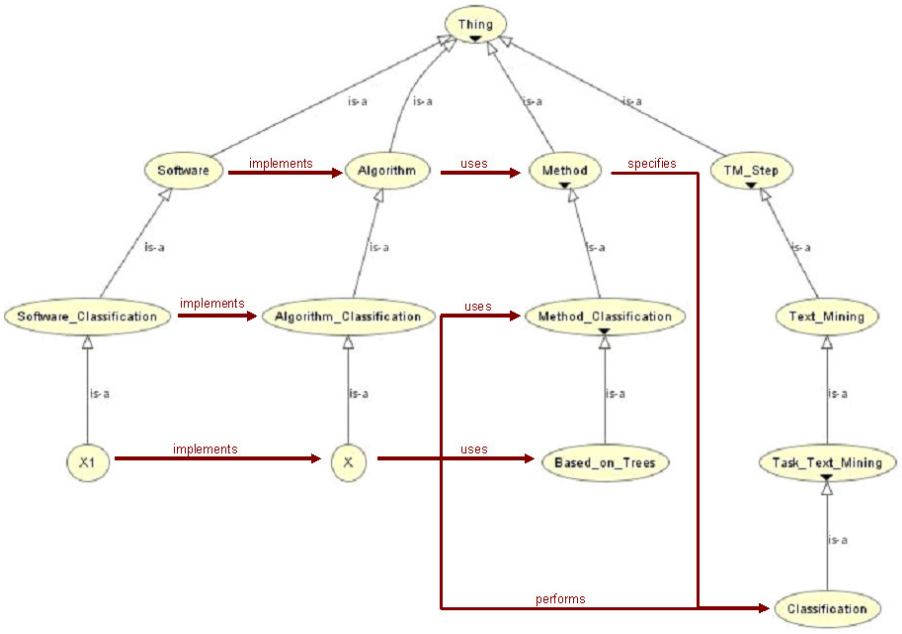

Figure 3: Part of the MF-Ontology

4. Evaluation of MF-Ontology

4.1. A Methodology for evaluation

The process of modeling and building an ontology is, despite the many efforts of the scientific community, almost a craft work. It is a well-known statement by the scientific community for the same domain there may be several different ontologies, each with its modeling particularity. Precisely because of this variety, it is extremely complex to define whether it is "certain" or "wrong" since the same concept can be modeled properly in different ways.

To be considered useful artifacts, ontologies in general need to offer a high quality. To ensure high quality, ontologies must be submitted to an evaluate process

according to different criteria [14]. Despite the need for high quality ontologies only very few approaches for ontology evaluation exist so far.

In literature, there are some proposals for ontology evaluations. One of the most prominent is OntoClean, proposed by Guarino and Welty [16]. Typically, this approach has served to prove the usability of an ontology which has been developed for any domain of knowledge.

In this work, the evaluation of MF-Ontology was performed using both OntoClean methodology and an additional evaluation through systematized interviews with domain experts that were related to the meta-properties of OntoClean methodology.

The OntoClean methodology is the most prominent methodology for ontology evaluation. In a few words, OntoClean methodology enables structural analysis of concepts and relationships based on the philosophical notions rigidity, unity, dependency and identity (commonly called meta-properties).

Identity is fundamental notion for ontologies. Identity is well known in database conceptual modeling. According to database modeling is very common to specify a primary key for rows in a specific table. If two different rows have identical primary keys, they are considered the same row. The same notion can be applied to ontologies. The idea of this notion is finding the conditions under which a proposed entity would be both the same and different.

Unity is the meta-property (commonly symbolized by +U) of classes in which all individuals related to the class are wholes under the same relation. Intuitively, a class has unity if all instances related to the class are the same type of whole. This is typically true for classes of natural objects. Non-unity (symbolized by –U) is the meta-property of classes whose instances are not all wholes, or at least not all wholes by the same relation.

The Concept of Rigidity is defined as “a property is rigid if and only if it is essential for all instances of classes”. Take, for example a class "person", if there was a

property "Being Person" it would be rigid. For instance, "Being tough" may be essential to the class hammer, but not hard, because in the case of sponge it is not essential. The property "Be Hard" is called a semi-rigid, it is essential to a class and not essential to another. All property of ontology should be classified into rigid, semi-rigid and anti-rigid. Thus, to ensure that the ontology is correct, the authors propose some restrictions to evaluate it based on the meta-properties.

Dependence is a varied notion. A property is dependent of if each instance of this property is directly related to the existence of another instance. The property “son”, for example, is dependent, since to be a son there must be a father. In other words, for every instance of son there is at least one instance of father.

By defining the OntoClean meta-properties, the ontology designer can define and verify the ontology in a formal way. For instance, according to [39] “by stating if a certain concept is rigid or not helps to understand if this concept is meant to be used as a role, that applies to an individual, or rather as an essential type”. It means that, if the ontology designer is able to answer these kind of questions, his ontology will be more “clean” and easier to be (re)used consistently.

The application of OntoClean methodology is composed by two main steps: the ontology tagging step and an ontology checking step. In the first step, all ontological concepts that were modeled ate tagged with an OntoClean meta-property. In our implementation of the MF-Ontology, we used an implementation of OntoClean in Protégé. Although Protégé offers an environment integrated to the OntoClean, ontology tagging can be considered a manual process yet. In our case, we spent a lot of time tagging all concepts of the ontology according the meta-properties. After all concepts are tagged, the methodology specifies that a set of defined constraints that must be checked in order to conclude that a potential modeling error exists.

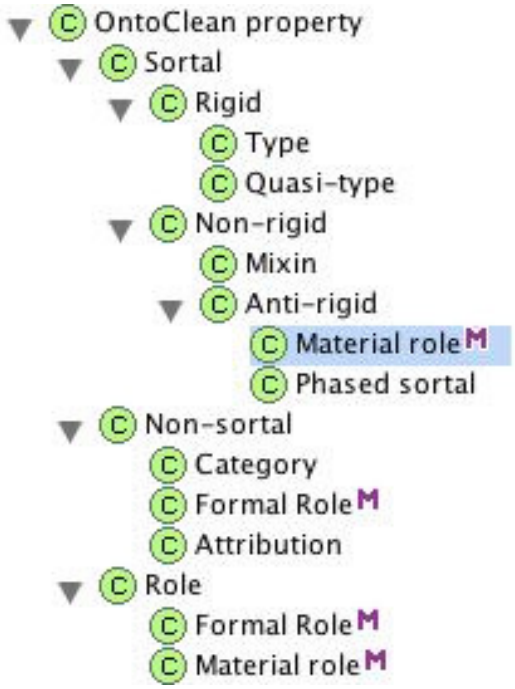

Figure 4: Implementation of the OntoClean ontology in Protégé

Despite the OntoClean methodology be considered a reference among the proposed techniques for ontology evaluation, it can not evaluate an ontology in all of its aspects. OntoClean is a domain-independent ontology evaluation methodology, and because of that OntoClean is not proposing to assess whether an ontology is appropriate for a specific domain or expertise area, indicating whether any concept of the domain was not modeled, for example. The technique used to evaluate the ontology in relation to these problems was to evaluate modeling through interviews with specialists (in our case, the text mining specialists).

Although interviews are considered a classic way to extract knowledge from specialists in order to model it into ontology, it is completely independent to the meta-properties evaluation. A domain specialist could validate the existing concepts but the ontology can still have misconceptualisations. On the other hand, the ontology designer could modeled a concept in a wrong way, and despite OntoClean evaluated that, it could be considered wrong by a specialist.

To relate the interviews and OntoClean, we propose a interview script that links the questions to the meta-properties of OntoClean. This way, we try to guide the specialist to evaluate the modeling (if there is a missing concept, for instance) and guarantee the ontology integrity based on the OntoClean methodology.

The script was drawn up so that questions could translate and evaluate a meta-property used by OntoClean and thus could show encountered problems even before MF-Ontology submitted to the method. In addition, the questionnaire make us able to check whether any concept has not been previously modeled and this way we could incorporate it in ontology.

Let us take, for example, two classes of ontology, A and B, where B is subclass of A. Contextualizing the MF-Ontology can assume that A is a class Method and B the class Method Based on Trees. One question in the questionnaire of this evaluation is: "Is it possible that a method based on trees is not a method?" Or "Is the method based on trees always a text mining method?". With these questions we are evaluating the axioms of ontology and if the answer is “no” for the first case, and “yes” to the second, we would have already endorsed the meta-property identity. In the assessment form were created to validate the classes, properties and relationships, relating a meta-property to a question being evaluated.

4.2. Results

The ontology evaluation was performed in two main steps: the OntoClean structural evaluation and the systematic interviews with domains specialists.

In first step of the evaluation process, the ontology was submitted to specialists for a concept evaluation. In this step were interviewed three specialists of the domain (text mining), all with qualified academic training and experience. The specialists are university professors, with several publications in national and international conferences and academic journals.

At this evaluation stage no errors were detected in the definitions of existing concepts. However some new concepts were added to the ontology, since the specialists noticed the absence of some concepts, for example:

1. Add the Lemmatization function in the pre-processing step: lemmatization is the process of grouping together the different inflected forms of a word so they can be analyzed as a single item. In computing, lemmatization is the algorithmic process of determining the lemma for a specific word. Lemmatization is closely related to stemming. The difference is that a stemmer operates on a single word without knowledge of the context, and therefore cannot discriminate between words which have different meanings depending on part of speech. In this context, we remodeled the ontology, including the new concept of Lemmatization Function. We also included a new restriction in the ontology to represent incompatible functions. In this case, it does not make sense to use a Lemmatization Function and a Stemming Function in the same Text Mining Process.

2. Add Summarization (defined as the creation of a shortened version of a text by a computer program) and Link Analysis (defined as a subset of network analysis, exploring associations between objects, in this case, text documents) Tasks: these tasks are very common to the development of a text mining process. The first version of MF-Ontology was developed for a Master Thesis case study, so that the ontology was not completed at that time.

3. Add Model Evaluation and Result Comparison as post-processing Functions: for the same reason explained before, in the first version of the ontology, these concepts were not modeled. However, they are key concepts and could not be forgotten.

Other observations were made related to the way of the concept method was modeled in the ontology. The text mining methods are essentially derived from data mining methods. Thus, all methods (decision trees, neural networks, and so on) are also data mining methods. However, for our evaluation, this concept can be correctly modeled since according to the responses we ensure the meta-properties Identity and Unit, for the validation later via Protégé.

The completed form obtained from the interviews can be downloaded (XLS file) at www.cos.ufrj.br/ ~ danielc.

The second phase of evaluation process was to submit the MF-Ontology to the OntoClean methodology. Through Protégé OntoClean implementation, each class and

relationship of the MF-Ontology was analyzed by OntoClean, generating a report with the inconsistencies found.

The main inconsistency found by Protégé was related to the Class Data. Initially we had modeled two subclasses of Data: Data Input and Data Output. However, these two classes have not been evaluated in the identity meta-property as both have the same instances (PDF document, for example).

Analyzing this inconsistency, we could observe that data type is the same concept whether if it is associated with inputs or outputs. Because of this inconsistency, we have remodeled the ontology, withdrawing of the classes Data Input and Data Output and creating a Class named Data. To differentiate a given input of a given output, were created two properties in each instance of Software: Input Type and Output type that reference to instances of Data.

In the current version of the ontology 49 classes have been modeled, 13 relationships between the classes (and therefore among the instances of the class), 21 properties of classes and instances, and for the case study version, the ontology was populated with 63 instances of software.

5. Conclusion and Future Work

To offer support for the modeling process of a scientific workflow for the text mining domain, this work proposes the MF-Ontology, a domain ontology for the text mining field, adapted and extended from a more generic ontology dedicated to the data mining process. The MF-Ontology can be easily extended to other contexts in the same area (Web mining, for example).

This ontology can be easily incorporated into SGWf in order to offer semantic support to guide the user in the construction of his scientific workflow of text mining. The design and construction of MF-Ontology provides subsidies for the construction of

a tool that fulfill all the requirements of the text mining applications, besides of that, the ontology modeling helped us a lot in understanding the process of text mining in details.

MF-Ontology can be used to help text mining specialists in the search for available services, and in the selection of adequate services for each step of the Knowledge Discovery in Texts cycle, during workflow definition. Both the search and the composition of services are enhanced through the use of this domain ontology, which provides richer semantics and a validation mechanism to reduce the incidence of errors in the workflow definition.

MF-Ontology per se can be considered a contribution. However in this work we proposed a systematic evaluation process that complements the OntoClean methodology. Ontology evaluation is a critical task for ontology specialists. OntoClean is the most prominent approach that checks the taxonomic structure of the ontology. First of all, applying OntoClean helps ontology specialists to better understand the ontology proposed. Besides of that, OntoClean allows for an evaluation of the formal properties of an ontology to detect misconceptualisations. However, OntoClean methodology is not able to check modeling problems like concepts that were not taken account when the ontology was modeled. Our approach complements OntoClean proposing systematic interviews with domain specialists. These interviews are composed by several questions about the domain. Which question of the questionnaire is related to a meta-property of OntoClean. Applying this technique we assure that the modeled ontology is correct either structurally and conceptually.

Future work includes an extension of MF-Ontology to model missing concepts. The main idea is to develop an internet crawler that capture new concepts related to the Text Mining domain on the web and incorporates them into the MF-Ontology. Besides, the application of the systematic interviews along OntoClean on other ontologies of other domains is one of our plans.

Acknowledgments

Research reported in this paper has been partially financed by CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) through a M.Sc. scholarship. Special thanks to Professor Geraldo Xexéo and Professor Alexandre Evsukoff, for their useful ideas and comments.

6. References

1. Altintas, I., et al. "Kepler: An Extensible System for Design and Execution of Scientific Workflows", Proceeding of 16th Intl. Conf. on Scientific and Statistical Database Management (SSDBM'04), 21-23 June 2004, Santorini Island, Greece.

2. Altintas, I., Barney, O., Jaeger-Frank, E., 2006, "Provenance Collection Support in the Kepler Scientific Workflow System". IPAW2006, Chicago, Illinois, May 2006.

3. Cai, D., He, X. and Han, J. "Document Clustering using locality preserving indexing". In IEEE Trans. Knowledge and Data Engineering, 1624-1637, 2005.

4. Cannataro, M., Comito, C. 2003, "A Data Mining Ontology for Grid Programming", in Workshop on Semantics in Peer-to-Peer and Grid Computing (in conj. with WWW2003), march 2003.

5. Cannataro, M., Talia, D. 2003, "KNOWLEDGE GRID: an architecture for distributed knowledge discovery", Communications of ACM, CACM, Vol. 46, N.1, pp. 89-96, january 2003.

6. Cannataro, M., Veltri, P. 2006, "MS-Analyzer: Composing and Executing Preprocessing and Data Mining Services for Proteomics Applications", in Concurrency and Computation: practice and experience, volume 19, issue 15, pages 2047-2066, december 2006.

7. Extensible Markup Language (XML), World Wide Web Consortium, URL: http://www.w3.org/XML/, visitado em 15/12/2007.

8. Feldman, R; Dagan, I., 1995, "Knowledge discovery in textual databases (KDT)". In proceedings of The First International Conference on Knowledge Discovery and Data Mining (KDD-95), Montreal Canada, August 20-21, AAAI Press, 112-117.

9. Feldman, R., Sanger, J., 2006, The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data, Cambridge University Press, 2006.

10. Fensel, D. et al., 2000, "OIL in a nutshell In: Knowledge Acquisition, Modeling, and Management", Proceedings of the European Knowledge Acquisition Conference (EKAW-2000), R. Dieng et al. (eds.), Lecture Notes in Artificial Intelligence, LNAI, Springer-Verlag, October.

11. Foster, I.; Kesselman, C. The Grid: Blueprint for a new computing infrastructure. Morgan Kaufmann. 1998.

12. Foster, I. A Globus Primer. Available at http://www.globus.org/toolkit/docs/4.0/key/. 2005.

13. Globus Toolkit. Available at http://www.globus.org/toolkit/.

14. Gómez-Pérez, A. (2004). Ontology Evaluation. In Handbook on Ontologies in Information Systems. Berlin: Springer.

15. Gruber, T.R., 1993, A translation approach to portable ontology specifications – Knowledge Acquisition – 5: 199-220

16. Guarino, N., Welty, C., 2002 "Evaluating Ontological Decisions with OntoClean", Communications of the ACM. 2(45):61–65, 2002.

17. Guarino, N. Formal Ontology and Information Systems. In: International Conference on Formal Ontologies in Information Systems (FOIS). Trento, Italy, June 1998. pp. 3-15.

18. Guizzardi, G., 2005 "Ontological Foundations for Structural Conceptual Models", PhD Thesis series, No 05 – 74, Telematica Instituut Fundamental Research Series, Enschede, The Netherlands, 2005.

19. Han, J., Kamber, M. "Data Mining: Concepts and Techniques". San Francisco, USA: Morgan Kaufmann, 2001. pp. 614-628, 364-365, 374

20. Hoffman, T. "Probabilistic latent semantic indexing". In Proc. 1999 Int. ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR`99), 50-57, Berkley, CA, Aug. 1998

21. Horrocks, I., Patel-Schneider, P. F., Van Harmelen, F., 2002, "Reviewing the Design of {DAML+OIL}: An Ontology Language for the Semantic Web". Presented at 18th Nat. Conf. on Artificial Intelligence (AAAI2002).

22. Joachims, T. "A statistical learning model of text classification with support vector machines". In Proc. Int. 2001 ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR`01), 128-136, New Orleans, LA, Sept. 2001.

23. Kepler Project, 2007, Available on : http://kepler-project.org/

24. Kepler Provenance Framework. 2007; Available from: http://kepler-project.org/Wiki.jsp?page=KeplerProvenanceFramework.

25. McGuiness, D. L , Van Harmelen, F, "OWL Web Ontology Language Overview. W3C Recommendation - World Wide Web Consortium, 2004

26. Mierswa, I., et al., 2006 "YALE: Rapid Prototyping for Complex Data Mining Tasks". The 12th Annual SIGKDD International Conference on Knowledge Discovery and Data Mining. Philadelphia, USA, 2006.

27. Mierswa, I., Wurst, M., Klinkenberg, R. and Scholz, M. "YALE: rapid prototype for complex data mining tasks", Proceedings of the 12th ACM SIGKDD International conference on knowledge discovery and Data Mining, Philadelphia, USA. 2006. pp. 935-940.

28. Nigam, K., McCallum, A., Thrun, S. and Mitchel, T. "Text classification from labeled and unlabeled documents using EM". Machine Learning, 103-134, 2000.

29. Noy, N. F., McGuiness, D. L., 2001, "Ontology Development 101: A Guide to Creating Your First Ontology". Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001.

30. Oinn, Tom; Greenwood, Mark; Addis, Matthew; Alpdemir, M. Nedim; Ferris, Justin; Glover, Kevin; Goble, Carole; Goderis, Antoon; Hull, Duncan; Marvin, Darren; Li, Peter; Lord, Phillip; Pocock, Matthew; Senger, Martin; Stevens, Robert; Wipat, Anil; Wroe, Chris. Taverna: Lessons in creating a workflow environment for the life sciences. In: Concurrency and Computation: Practice and Experience, pp.2. 2002.

31. Oinn, Tom; Addis, Matthew; Ferris, Justin; Marvin, Darren; Senger, Martin; Greenwood, Mark; Carver, Tim; Glover, Kevin; Pocock, Matthew R.; Wipat, Anil; Li, Peter. Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics Journal, 20(17), pp. 3045-3054. 2004.

32. Open Source Problem Solving Environment. 2007; Available from: http://www.trianacode.org/index.html.

33. OWL Ontology Web Language Reference, World Wide Web Consortium, URL: http://www.w3.org/TR/owl-ref/, visitado em 15/12/2007.

34. Protégé Ontology Editor and Knowledge Acquisition System, URL: http://protege.stanford.edu/

35. RDF Vocabulary Description Language 1.0: RDF Schema, World Wide Web Consortium, URL: http://www.w3.org/TR/rdf-schema/, visitado em 15/12/2007.

36. Sebastiani, F. "Machine learning in automated text categorization". ACM computing surveys, 34: 1-47, 2002.

37. Stevens, R.; Robinson, A.; Goble, C. myGrid: Personalized bioinformatics on the information grid. Bioinformatics, 19(1), pp. 302-304. 2003.

38. Taverna Project Website. 2006. Available at http://taverna.sourceforge.net/.

39. Völker, J, Vrandecic, D, Sure, Y and Hotho, A, AEON – An approach to the automatic evaluation of ontologies, 2008, Applied Ontologies

40. Wang, K., Zhou, S. and Liew, S. C. "Building hierarchical classifiers using class proximity" In Proc.1999 Int. Conf. Very Large Data Bases (VLDB`99), pp. 363-374, Edinburgh, UK, Sept. 1999

41. Witten, I. H. and Frank, E. "Data Mining: Practical machine learning tools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

42. Wroe, C.; Lord, P.; Miles, S., Papay, J., Moreau, L.; Goble, C. Recycling Services and Workflows through Discovery and Reuse. Proc UK e-Science All Hands Meeting 2004, pp. 622-629. 2004.