Subgroup Discovery as a Method for Generating Test Ontologies

Daniel Knoell[1], Constantin Rieder[1], Martin Atzmueller[2], and Klaus Peter Scherer[1]

[1] Karlsruhe Institute of Technology
D-76344, Eggenstein-Leopoldshafen, Germany
firstname.lastname@kit.edu

[2] University of Kassel, Research Center for Information System Design
Knowledge and Data Engineering Group
Wilhelmshöher Allee 73, 34121 Kassel, Germany
atzmueller@cs.uni-kassel.de

Abstract. For the validation of ontologies, test cases can be applied. In order to ease the acquisition of such test cases, automatic methods can be used. This paper introduces such an approach to unit testing of ontologies using subgroup discovery methods. The tests are given by small ontologies, which will be compared to an existing ontology. The generation of the test ontologies uses subgroup discovery methods to derive concepts and relations from real data which refers to the ontology. For each concept of the ontology, we apply subgroup discovery in order to identify subgroup patterns consisting of concepts that are (un-)correlated with the target concept. Then, subgroups with very high (or low) confidence will be selected for the testing ontology. We exemplify the approach using real-world data from the ophthalmology domain, and present first results and experiences of a case study demonstrating the test case generation process.

1 Introduction

Not only ontology learning [9], but also the evaluation [15] of the learned ontologies is getting more and more important. In particular, this relates to the automatic evaluation of ontologies [17]. In this area, there exist different approaches like OntoClean[8] and AEON (Automatic Evaluation of Ontologies)[13]: OntoClean, for example aims for validating an ontology with the taxonomic relationships while AEON continues this approach with an automatically tagging of the OntoClean meta-properties.

One prominent method for the validation of knowledge systems considers the utilization of test cases, e. g., [14, 7]. Below, we sketch a method for ontology validation using automatically generated testing ontologies. For obtaining the appropriate test axioms automatically, we apply subgroup discovery – a versatile method for data analytics. Subgroup discovery [16, 3] aims at identifying interesting subsets of a dataset with

In: R. Bergmann, S. Görg, G. Müller (Eds.): Proceedings of the LWA 2015 Workshops: KDML, FGWM, IR, and FGDB. Trier, Germany, 7.-9. October 2015, published at http://ceur-ws.org

respect to a certain property of interest. Each subset that can be characterized by a certain descriptive *pattern* is then called a *subgroup*. For the construction of test cases, we generate patterns that are correlated with the target concept, or those that exclude the target concept.

Intuitively, we identify candidates for derivable axioms using patterns that have a high share of a certain target concept, i. e., the concepts making up the description of the pattern show a (strong) correlation with the target concept. Likewise, we obtain candidates for non-derivable axioms by constructing relations for which we are sure that they are not observed in the data. That is, for the validation of an ontology O, we apply two test ontologies as in [14]: T^+ and T^-, that specificy all axioms that should be derivable from the given ontology O, and the axioms that should not be derivable given O.

Below, we first introduce some background before we sketch our novel approach for the generation of automatic test cases. After that, we provide a case study using real-world data from the ophthalmology domain. The data is extracted from anonymized surgery sheets which are filled in by the eye specialist itself. These sheets include structured data like which eye was medicated or which medicine was used. We demonstrate the test case generation process, and discuss first experiences and implications.

2 Background

In this section, we briefly summarize necessary notation and basic concepts on subgroup discovery and the proposed approach for ontology validation using test ontologies.

2.1 Patterns and Subgroups

Formally, a *database* $DB = (I, A)$ is given by a set of individuals I and a set of attributes A. A *selector* or *basic pattern* $sel_{a_i = v_j}$ is a Boolean function $I \rightarrow \{0, 1\}$ that is true if the value of attribute $a_i \in A$ is equal to v_j for the respective individual. The set of all basic patterns is denoted by S. For a numeric attribute a_{num} selectors $sel_{a_{num} \in [min_j; max_j]}$ can be defined analogously for each interval $[min_j; max_j]$ in the domain of a_{num}. The Boolean function is then set to true if the value of attribute a_{num} is within the respective range.

Definition 1. *A* subgroup description *or (complex)* pattern *sd is given by a set of basic patterns $sd = \{sel_1, \ldots, sel_l\}$, where $sel_i \in S$, which is interpreted as a conjunction, i.e., $sd(I) = sel_1 \wedge \ldots \wedge sel_l$, with $length(sd) = l$.*

Without loss of generality, we focus on a conjunctive pattern language using nominal attribute–value pairs as defined above in this paper; internal disjunctions can also be generated by appropriate attribute–value construction methods, if necessary.

Definition 2. *A* subgroup (extension)

$$sg_{sd} := ext(sd) := \{i \in I | sd(i) = true\}$$

is the set of all individuals which are covered by the pattern sd.

As search space for subgroup discovery the set of all possible patterns 2^S is used, that is, all combinations of the basic patterns contained in S. For the practical implementation, we utilize the VIKAMINE platform [5] that provides efficient algorithms e. g., [10, 3] and according interestingness measure described below.

2.2 Interestingness of a Pattern

The interestingness of a pattern is determined by a quality function, that is selected according to the analysis task.

Definition 3. *A* quality function $q\colon 2^S \to \mathbb{R}$ *maps every pattern in the search space to a real number that reflects the interestingness of a pattern (or the extension of the pattern, respectively).*

While a large number of quality functions has been proposed in literature, many quality functions for a single target concept, e. g., in the binary or numerical case, trade-off the size $n = |ext(sd)|$ of a subgroup and the deviation $t_{sd} - t_0$, where t_{sd} is the average value of a given target concept in the subgroup identified by the pattern sd and t_0 the average value of the target concept in the general population. In the binary case, the averages relate to the *share* of the target concept. Then, this is equivalent to the concept of *confidence* commonly used in the field of association rule mining, e. g., [1]. In our context, we focus on binary target concepts only. Then, the subgroup patterns introduced above are equivalent to *class association rules* [11].

For subgroup discovery, typical quality functions are of the form

$$q_a(sd) = n^a \cdot (t_{sd} - t_0),\ a \in [0; 1]\,.$$

For binary target concepts, this includes, for example, the *weighted relative accuracy* for the size parameter $a = 1$ or a simplified binomial function, for $a = 0.5$. An extension to a target concept defined by a set of variables can be defined similarly, by extending common statistical tests.

The result of a subgroup discovery task is then the set of k (subgroup) patterns $sd_1, \ldots, sd_k$, where $sd_i \in 2^S$ with the highest interestingness according to the quality function. For our experiments in test ontology generation, we applied the simplified binomial function $q_{0.5}$ described above that conventiently balances the size of the subgroup with a high target share of the target concept.

2.3 Formalization of Test Ontologies

For the unit testing of ontologies, we adopt the approach proposed by Vrandecic and Gangemi [14]: We consider two testing ontologies. The ontology T^+ contains the axioms that should be derivable, and T^- contains the axioms that essentially should not be derivable. Thus, according to [14], for ontology O we have

$$O \models A_i^+ \,\forall A_i^+ \in T+\,,$$

for all axioms $A_i^+ \in T+, i = 1, \ldots, n$, and for all axioms $A_i^- \in T-, i = 1, \ldots, m$:

$$O \models A_i^- \,\forall A_i^- \in T-\ .$$

In this framework, the specific axioms themselves can be considered as atomic test cases for the ontology that need to be formalized accordingly. In the next section, we describe our approach for generating these axioms in an automatic fashion.

3 Method: Generating Test Ontologies

In the following, we first provide an overview on the proposed approach, before we discuss the test ontology generation in detail and present an algorithm for this task.

3.1 Overview

The basic idea of test ontology generation using subgroup discovery is the following: We start with all relevant test concepts contained in ontology O and map these to our test instances contained in the dataset. Then, for each target concept $t \in S$, we discover subgroups according to the total selector set S using a specific quality function (we used $q_{0.5}$ in our experiments). After that, we map these subgroup patterns to axioms that are integrated in the testing ontologies $T+$ and $T-$, respectively. For that step, we create complex classes for each selected subgroup pattern.

3.2 Generating Test Ontologies

For generating test ontologies from the subgroup discovery results, we map *(un-)correlated* patterns to the axioms of $T+$ and $T-$, respectively. Intuitively, when generating the according axioms using the discovered subgroup patterns, there must be a value which decides which subgroup is used and which not. There are several strategies for selecting the respective subgroup patterns used for creating the axioms. Below, we consider two variants: A confidence-based approach, and a test-coverage-based one.

1. A simple variant is a confidence-based approach. According to the share of the target concept in the subgroup, i. e., its *confidence*, we use the respective confidence of a subgroup to decide whether it should be used for the test ontology or not. If the confidence is higher than a specific threshold, it is used as for the T^{+} ontology. If the confidence is below a specific other threshold, it is used for the T^{-} ontology. The threshold for T^{-} has to be very low or zero, because otherwise the test will also fail for the same dataset. Therefore it is also possible to define that not all of the test cases in the T^{-} ontology have to pass. It would be suitable if for example eight out of ten test cases pass to let the whole test pass.
2. An alternative approach is test-coverage-based and will be described using an example. For a specific target value two subgroups were detected. These subgroups only contain three different selectors. Subgroup one, for example, contains selector A and B and subgroup two contains selector A and C. In our example we add the two subgroups to the test ontology. In reality we would use a quality function to rate them and only add for example the five best subgroups to the test ontology. Back to our example with the two subgroups: If an ontology contains the target value and is connected with two out of three selectors it would be a coverage of $\frac{2}{3}$. With this approach we also need a threshold which decides if the test will pass or not. If this threshold would be $\frac{1}{3}$, our test in the example would have passed.

In our algorithm presented below, we use the described confidence-based approach, because it is easier to implement in this initial stage. In the future, we also plan a test-coverage-based approach to compare the results.

3.3 Algorithm

For the automatic generation of the described test ontologies we developed the algorithm shown in Listing 1.1. It uses subgroup discovery to derive interesting correlations and transforms them to small test ontologies consisting of complex classes.

Listing 1.1: Algorithm in pseudo code

```
for each attribute{
  subgroupset = findSubgroups(attribute)
  candidatesTplus = [], candidatesTminus = []
  for each subgroup in subgroupset{
    if subgroup.confidence >= thresholdPlus
      candidatesTplus.add(subgroup)
    if subgroup.confidence <= thresholdMinus
      candidatesTminus.add(subgroup)
  }
  makeComplexClasses(candidatesTplus, candidatesTminus)
}
```

In the first step, subgroup discovery is performed for each attribute of the dataset. Therefore, the current attribute $t \in S$ is used as target value. For each target variable a set of subgroups is calculated. Each set contains all subgroups with the appropriate target variable.

Each subgroup in the set that exceeds a certain confidence threshold is added as a test candidate to the positive test candidate set (candidatesTplus). If the confidence is under a specific different threshold. In this case the subgroup is added to the negative candidate set (candidatesTminus). Thus each attribute has its own two test candidate sets, which contain all test cases for the specific attribute. A test candidate consists of any number of selectors that are associated by the AND function. This combination of selectors is characteristic for the target variable related to a high confidence and therefore will be added to the positive test candidate set. If we consider the exemplary “vehicle”-domain, for example, then the target variable “vehicle type = tricycle” has the significant describing combination of the selectors “number of wheels = 3” and “engine = false”.

For the generation, we use the confidence as a threshold. From the test candidate set a complex class is created for each attribute. This set is composed of the individual test candidates concatenated by the OR function. Thus generated complex classes will be compared or tested against an existing ontology in the next step.

4 Case Study

This section demonstrates the proposed approach using real-world data and presents first results. It covers the ontology structure and the working on the generation of T^+ ontologies.

4.1 Overview

In order to verify our approach we used anonymised operative reports from the cataract surgery domain to set up a dataset for testing. We have extracted the structured data from the data sheets. All records were reduced, encoded, typed and converted to a suitable format. Finally, after the data wrangling the latest version of the test dataset for this work contained more than 500 instances described by 80 attributes. As output format "ARFF"(Attribute-Relation File Format) was used, which is an input format for data mining tools like WEKA or VIKAMINE. For subgroup discovery, the VIKAMINE platform was used, because WEKA does not support subgroup discovery out of the box. The VIKAMINE system offers a rich-client environment for subgroup discovery and analytics [5]. We iterated over every attribute and performed a subgroup discovery with VIKAMINE for each attribute as target value. With the results of the subgroup discovery we create test ontologies for each attribute using the confidence based approach. These test ontologies are written in a KnowWE(Knowledge Wiki Environment) Wiki. KnowWE has export functions for the ontology which supports various popular formats like Turtle, RDF/XML or RDF/JSON, cf. [2].

4.2 Structure of the basic ontology

The output ontology follows a specific order. A cataract ontology[12] was designed and the structure of the attribute is displayed in Figure 1. Each real attribute is a subclass of the "Attribute"-class and each class has its own wiki page.

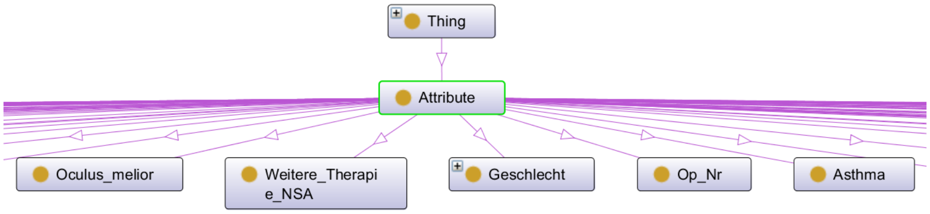

Fig. 1: Overview of the attribute classes of the ontology

The attribute value is stored in the instances of the specific attribute. A visualisation of the the instances of the attribute "Pupille" (German for pupil) is shown in Figure 2. This attribute has three possible values: "weit", "eng" and "mittel". For each value there is a corresponding instance of the attribute with the matching data property. For the data property the relation "hasValue" is used. As an example the instance "PupilleMittel" of type "Pupille" has a data property "hasValue" with the value "mittel".

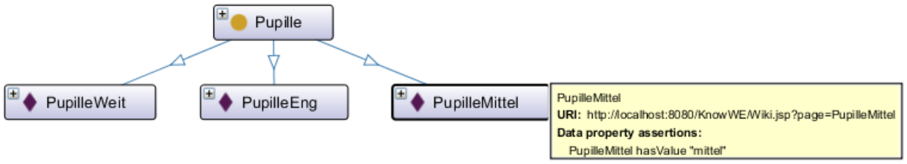
Pupille
PupilleWeit
PupilleEng
PupilleMittel
PupilleMittel
URI: http://localhost:8080/KnowWE/Wiki.jsp?page=PupilleMittel
Data property assertions:
PupilleMittel hasValue "mittel"

Fig. 2: Pupil class with the corresponding individuals

4.3 Automatic generation of the test ontologies

Below, we describe first exemplary results of our proof-of-concept implementation of the proposed approach. In the basic setting for subgroup discovery we applied the BSD algorithm [10] with the quality function $q_{0.5}$ and a minimum quality limit of 0.8.

After performing subgroup discovery and constraining the subgroup description length to one concept for each attribute of our dataset we pick up exemplary the target value “Patient_unruhig = true” which means that the patient is nervous and in a state of agitation during the surgery. One of the subgroups from the derived amount of subgroups which fulfill the quality limit in this case is “Sedativum = true”. That is no surprise because one of possible logical consequences for the operating surgeon could be to sedate the patient. The resulting subgroup from the corresponding dataset confirms this practice. The identified pattern can now be used to generate a test case for the cataract ontology because we know there is a correlation with a certain quality limit between the above mentioned items. Incrementing the description length to two concepts yields further patterns, for example, “Diabetes = true” is described among other descriptions by the combination of concepts “Infusion_Lidocain = false” and “Gerinnungshemmer = true”. In this way further test cases can be created to extend our test ontology by mapping the identified patterns to a complex class.

Our first results and experiences in the case study demonstrate, that the proposed approach is working well. It is a very transparent process for test ontology generation since the subgroup patterns can be directly inspected, cf. [6], and transparently mapped to complex classes in the ontology. Currently, we are still refining the results with respect to the testing ontologies $T+$ and $T-$ and more complex test-coverage strategies.

5 Conclusions

In this paper, we have presented a novel approach for the automatic generation of test ontologies. Using subgroup discovery, we identify patterns that are mapped to axioms in our testing ontology. In particular, these are formalized using complex classes which can then be directly implemented in unit testing manner. We demonstrated the basic concepts in a case study using a real-world dataset. Currently, our proof-of-concept implementation focuses on a confidence-based strategy for test ontology generation. We aim to investigate the approach further for generating more complex testing ontologies ($T+$ and $T-$) and to apply a test-coverage-based approach in order to have fine-grained control when a testing ontology passes or fails. Furthermore, the presented basic approach can then also be extended using semi-automatic strategies, e. g., cf. [4].

Acknowledgements

The research project is a cooperation of the Karlsruhe Institute of Technology, Germany (KIT) and the denkbares GmbH. It is funded as a ZIM-KOOP by the German Federal Ministry of Economics and Technology (BMWI). The authors also thank the project executing organization AiF in Berlin, which is responsible for the allocation of the budget to the research center and the commercial company.

References

1. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In: Bocca, J.B., Jarke, M., Zaniolo, C. (eds.) Proc. VLDB. pp. 487–499. Morgan Kaufmann (1994)
2. Allemang, D., Hendler, J.A.: Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL. Morgan Kaufmann, Waltham Mass., 2. ed edn. (2011)
3. Atzmueller, M.: Subgroup Discovery – Advanced Review. WIREs: Data Mining and Knowledge Discovery 5(1), 35–49 (2015)
4. Atzmueller, M., Baumeister, J., Hemsing, A., Richter, E.J., Puppe, F.: Subgroup Mining for Interactive Knowledge Refinement. In: Proc. Conf. on Artificial Intelligence in Medicine (AIME). pp. 453–462. LNAI 3581, Springer, Heidelberg (2005)
5. Atzmueller, M., Lemmerich, F.: VIKAMINE - Open-Source Subgroup Discovery, Pattern Mining, and Analytics. In: Proc. ECML/PKDD. LNCS, vol. 7524, pp. 842–845. Springer, Heidelberg (2012)
6. Atzmueller, M., Puppe, F.: A Case-Based Approach for Characterization and Analysis of Subgroup Patterns. Journal of Applied Intelligence 28(3), 210–221 (2008)
7. Baumeister, J.: Advanced Empirical Testing. Knowl Based Syst 24(1), 83–94 (2011)
8. Guarino, N., Welty, C.: An Overview of OntoClean. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 151–171. Springer Berlin Heidelberg (2004)
9. Lehmann, J., Völker, J.: Perspectives on Ontology Learning, Studies on the Semantic Web, vol. Volume 018. IOS Press / AKA (2014)
10. Lemmerich, F., Rohlfs, M., Atzmueller, M.: Fast Discovery of Relevant Subgroup Patterns. In: Proc. FLAIRS. pp. 428–433. AAAI Press, Palo Alto, CA, USA (2010)
11. Liu, B., Hsu, W., Ma, Y.: Integrating Classification and Association Rule Mining. In: Proc. KDD. pp. 80–86. AAAI Press (August 1998)
12. Scherer, K.P., Rieder, C., Henninger, C., Germann, M., Baumeister, J., Reutelshöfer, J.: Modeling and Visualization of Cataract Ontologies. In: Proc. ADVCOMP (2014)
13. Völker, J., Vrandečić, D., Sure, Y.: Automatic Evaluation of Ontologies (AEON). In: Gil, Y., Motta, E., Benjamins, V., Musen, M. (eds.) The Semantic Web – ISWC 2005, Lecture Notes in Computer Science, vol. 3729, pp. 716–731. Springer Berlin Heidelberg (2005)
14. Vrandecic, D., Gangemi, A.: Unit Tests for Ontologies. In: Proc. 1st Intl. Workshop on Ontology Content and Evaluation in Enterprise. LNCS, Springer, Montpellier, France (2006)
15. Vrandečić, D.: Ontology Evaluation. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 293–313. Springer Berlin Heidelberg (2009)
16. Wrobel, S.: An Algorithm for Multi-Relational Discovery of Subgroups. In: Proc. PKDD-97. pp. 78–87. Springer, Heidelberg, Germany (1997)
17. Zablith, F., Antoniou, G., d'Aquin, M., Flouris, G., Kondylakis, H., Motta, E., Plexousakis, D., Sabou, M.: Ontology Evolution: A Process-Centric Survey. KER 30, 45–75 (1 2015)