Received February 19, 2020, accepted March 4, 2020, date of publication March 10, 2020, date of current version March 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2979281

Building Ontology-Driven Tutoring Models for Intelligent Tutoring Systems Using Data Mining

MAIGA CHANG[1,2], (Member, IEEE), GIUSEPPE D'ANIELLO[3], (Member, IEEE), MATTEO GAETA[3], (Senior Member, IEEE), FRANCESCO ORCIUOLI[4], (Member, IEEE), DEMETRIOS SAMPSON[5,6], (Senior Member, IEEE), AND CARMINE SIMONELLI[7]

[1]School of Computing and Information Systems, Athabasca University, Edmonton, AB T5J 3S8, Canada
[2]Department of M-Commerce and Multimedia Applications, Asia University, Taichung 41354, Taiwan
[3]Department of Information and Electrical Engineering and Applied Mathematics, University of Salerno, 84084 Fisciano, Italy
[4]Department of Management and Innovation Systems, University of Salerno, 84084 Fisciano, Italy
[5]Department of Digital Systems, University of Piraeus, 18534 Piraeus, Greece
[6]School of Education, Curtin University, Perth, WA 6102, Australia
[7]NTT DATA Italia spa, 80143 Naples, Italy

Corresponding author: Giuseppe D'Aniello (gidaniello@unisa.it)

This work was supported in part by the Italian Ministry of University and Research under Project INF@NZIA DIGI.tales 3.6 (specifically under the Work Package 2), and in part by the PON Research and Competitiveness, from 2007 to 2013—http://www.infanziadigitales.it/en.

ABSTRACT Pedagogical (Tutor or Tutoring) Models are an important element of Intelligent Tutoring Systems (ITS) and they can be described by sets of (tutoring) rules. The implementation of a Tutoring Model includes both the formal representation of the aforementioned rules and a mechanism able to interpret such representation and execute the rules. One of the most suitable approaches to formally represent pedagogical rules is to construct semantic web ontologies that are highly interoperable and can be integrated with other models in an ITS like the subject domain and the student model. However, the main drawback of semantic web-based approaches is that they require a considerable human effort to prepare and build relevant ontologies. This paper proposes a novel approach to maintain the benefits of the semantic web-based approach in representing pedagogical rules for an ITS, while overcoming its main drawback by employing a data mining technique to automatically extract rules from real-world tutoring sessions and represent them by means of Web Ontology Language (OWL).

INDEX TERMS Classification rule mining, intelligent tutoring systems, ontologies, pedagogical rules, semantic web, web ontology language (OWL).

I. INTRODUCTION

As defined in [1], an Intelligent Tutoring System (ITS) [2], [3] is a software system providing adaptive educational experiences. The main features of an ITS [4] include generation and delivery of (i) learning activities that are aligned to learners' current knowledge and skills status in order to foster meaningful learning ([5] and [6]), (ii) individualized feedback to stimulate next learning activities and avoid frustration ([7] and [8]), demotivation and disengagement due to unsuccessful performance, and (iii) guidance (typically in the form of hints for learning activities) that help learners (for instance [9], [10] and [11]) during the execution of their learning tasks [12], [13].

The associate editor coordinating the review of this manuscript and approving it for publication was Luigi De Russis.

From the system viewpoint, an ITS can be divided into five core conceptual components [14]: the *Expert Model* representing the domain knowledge; the *Domain Model* containing the knowledge about the actual teaching material; the *Student Model* storing learners' profile like, for instance, details about the learner's current problem-solving state and long-term knowledge progress, which are essential for adapting the experience; the *Pedagogical (Tutor or Tutoring) Model* providing the knowledge to tailor the selection and the provisioning of the teaching elements according to the student model; the *Communication (User Interface) Model* enabling the interactions between learner and system. Nkambou *et al.* [15] discuss the architectures of ITS.

Building the aforementioned models in an ITS requires to represent the knowledge included in such models and to define the mechanisms that are capable of reasoning and making inference over such a knowledge. In literature, a plethora of heterogeneous approaches to define these models are reported. Such approaches can be roughly divided into two macro-classes. The first class includes techniques that exploit formal languages to represent knowledge (e.g., ontology languages like Web Ontology Language-OWL); for instances, [16]–[18] and [19]. The second class is represented by techniques based on machine learning and data mining (e.g., artificial neural networks, reinforcement learning, rule mining); for instances, [20]–[24] and [25].

The main advantage of the techniques in the first class is that the knowledge represented in the models is accessible by both humans and automatic (software) agents. These models, and in particular ontological models, are easily reusable and interoperable, and support reasoning and inference mechanisms [26]. On the other hand, the human experts need to invest considerable effort and have specific competencies for building such models. The main advantage of the techniques of the second class is that it is possible to automatically build ITS models, to a large extent, based on processing existing data from real life scenarios using machine learning algorithms. The drawback of the second class is that the resulting models are typically represented as a black box, so they cannot be directly processed by humans [27]–[29].

In this context, an approach that combines the benefits of both techniques while overcoming their drawbacks is highly desirable. Thus, this paper presents a novel approach to build the pedagogical model of an ITS represented with the W3C Web Ontology Language (OWL) and a classification rule mining algorithm over a dataset containing the data observed during real-world tutoring sessions.

The proposed approach provides an approach to automatically build an ontology-driven ITS from real data, maintaining the benefits of both the first and the second classes of techniques. Furthermore, the ITS resulting by the application of the proposed approach is able to represent knowledge but also to execute reasoning on it against new learners' interactions. The allowed type of reasoning consists of selecting the most suitable tutoring action in response to the current interaction state of the learner within the given learning environment.

In this paper, the proposed approach is applied to the Tutoring Model of an ITS, for validation purposes. Further research could be conducted to study its applicability also to other models in Intelligent Tutoring Systems. The remaining part of the paper is structured as it follows: in the Section II an introduction to rule mining algorithms and computational ontologies (formal languages, methods, etc.) is provided. Section IV proposes the description of the hybrid approach to build tutoring models. Sections V, VI and VII provide details on the three phases of the proposed approach. Section VIII shows the application of the proposed approach in the context of a complete case study related to the definition of an adaptive tutor for 5-6 years-old children. Lastly, in the Section IX final remarks and possible future works are briefly discussed.

II. BACKGROUND KNOWLEDGE

A. OWL FOR KNOWLEDGE REPRESENTATION AND REASONING

Web Ontology Language (OWL) is the language provided by W3C[1] to represent knowledge. OWL is one of the main components of the so-called Semantic Web stack.[2] OWL is based on Description Logic [30] and provides a set of suitable capabilities like providing high levels of interoperability and integration and a formal inference mechanism. With respect to the aforementioned Semantic Web Stack, OWL stays upon Resource Description Framework Schema (RDF-S) and RDF. In particular, borrowing the description in [31], OWL is primarily concerned with defining terminology that can be used in RDF documents, i.e., classes and properties. Moreover, individuals in OWL are instances of classes. Most ontology languages have some mechanisms for specifying a taxonomy of the classes. In OWL, firstly, it is possible to specify taxonomies for both classes and properties and, secondly, it is possible to attach formal definition for classes by means of boolean operators or *restrictions*. In particular, the mechanism of *restrictions* represents one of the main tools exploited in this work. More in details, OWL allows to define the meaning of a class in terms of restrictions on the property values that may occur for individuals belonging to the class. Ontologies written in OWL can be embedded in information systems for transforming them into ontology-driven systems able to receive input, add this input to the current knowledge base, execute inference and provide reasoning results.

B. CLASSIFICATION RULE MINING

Association rule mining (ARM) [32] is a data mining task aiming at eliciting patterns from data. Such patterns are extracted as rules. ARM can be classified as an unsupervised machine learning technique. More formally, the problem of extracting association rules from data can be expressed as it follows.

Being $I = \{i_1, i_2, \ldots, i_n\}$ a set of n binary attributes (objects or items) and $T = \{t_1, t_2, \ldots, t_m\}$ a set of transactions (a dataset). Each transaction in T contains an identifier and includes a subset of I. A rule can be defined as an implication $X \rightarrow Y$, where $X, Y \subseteq I$ and $X \cap Y = \emptyset$. The itemsets X and Y are called antecedent and consequent of the rule.

It is possible to define two important measures for association rules: support and confidence. The support of an association rule is defined as:

$$Sup(X \rightarrow Y) = \frac{P(X \cap Y)}{|T|}, \tag{1}$$

i.e., $P(X \cap Y)$ is the number of transactions in which X and Y occur together and $|T|$ is the total number of transactions. The support provides information on the statistic relevance of itemsets X and Y.

[1]http://www.w3.org/TR/owl2-overview/
[2]http://www.w3.org/standards/semanticweb/

The second measure is called confidence and can be defined as it follows:

$$Conf(X \to Y) = \frac{Sup(X \to Y)}{Sup(X)}, \tag{2}$$

i.e., the support of the rule $X \to Y$ divided by the support of the itemset X. The confidence measures the strength of the rule $X \to Y$.

Typically, rule association mining algorithms have two main steps. The first one has the goal to explore the dataset and find frequent itemsets that have a support measure greater than a given threshold. In the second step, the algorithms explore the identified frequent itemsets and generate rules that have confidence measure greater than a given threshold from them.

In this context, Classification Rule Mining (CRM) [33] is a data mining task where the main goal is to execute classification processes by using association rules. In particular, CRM algorithms extract (from data) rules $X \to Y$ where Y is one of the possible classes for the classification task. More in details, once extracted classification rules from a dataset T it is possible to classify a new transaction t_{new} by using such rules. Among the different existing CRM algorithms, there is a technique called Classification Based on Predictive Association Rules (CPAR) [34]. The advantages of CPAR with respect to other techniques are:

- it generates a small set of high quality rules from the dataset;
- it generates a rule by considering also the previous generated rules in order to avoid redundancies; and,
- it adopts an effective heuristic to predict the class for a transaction (it uses only the k best rules satisfied by the transaction).

An important measure to evaluate the performance of algorithms for predictive association rule mining is accuracy. In particular, in this work we adopt the Laplace expected error estimate [34]:

$$w = \frac{n_c + 1}{n_{tot} + k}, \tag{3}$$

where k is the number of classes, n_{tot} is the total number of examples satisfying the rule body, among which n_c examples belong to c that is the class in the consequent of the current rule.

C. ITS BEHAVIOR SCHEME

The behavior of an ITS can be described as formed by an outer loop and inner loop (depicted in Fig. 1), as reported in the work of VanLehn [35], [36]. The *outer loop* provides learners with a sequence of *tasks* with different difficulties. The default behavior foresees that the next task, to be presented, is more difficult than the previously presented one (mastery learning). However, if the learner's performance is not good enough, the ITS can propose an alternative learning content or have the next task with less difficulty for the learners.

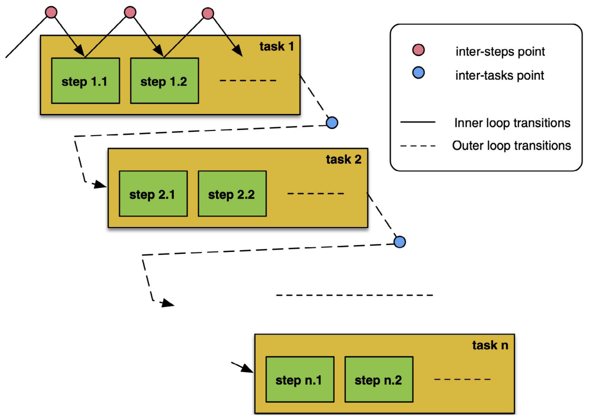


FIGURE 1. VanLehn's schema.

Moreover, there is an *inner loop* for each task where a sequence of steps has to be done by the learners in order to achieve the task objectives and provide a solution for the associated problem. The ITS can provide: i) adaptive feedback (positive, negative, etc.) in response to the learners' answers for the current step, and/or ii) hints to anticipate the next step of the same task. In Fig. 1, inter-steps points and inter-tasks points are moments in which feedback and hints are provided to students by the tutor.

Plausible actions that can be carried out by the ITS are called *tutoring actions*. The selection of the right tutoring action can be accomplished on the basis of pedagogical strategies, learners' profiles, context, domain, etc. The tutoring model in an ITS can be a set of rules (pedagogical rules) that are executed to select one or more suitable tutoring actions (at the end of a step/task) according to some variables like learner's performance, emotional/affective states, etc. Therefore, in our vision, we need a set of classification rules able to classify the learner's interaction data at the end of each step with respect to one or more tutoring actions that become the admissible classes.

III. RELATED WORKS

The use of ontologies to support the tutoring model in Intelligent Tutoring Systems has been proposed in numerous studies. George and Lal [26] proposed a survey of existing approaches based on ontologies to achieve personalization in the e-learning domain. Oguejiofor *et al.* [37] proposed an architecture of an ontology-based ITS for personal air vehicles with multimedia contents. Garcia *et al.* [38] proposed an ontology-based tutoring system in the control engineering education field. Stamatis *et al.* [39] proposed an ITS based on ontologies which is capable of adapting the learning contents according to the learner's profile. Vesin *et al.* [40] proposed a Java framework for building tutoring systems based on ontologies, namely the Protus 2.0 framework. The architecture of the framework is based on ontologies, with different ontological models for each component to improve the flexibility and interoperability among the modules, easing the development of tutoring systems. Rani *et al.* [41] proposed an ontology-driven adaptive tutoring system based on the Felder-Silverman learning style model. Dermeval *et al.* [42] proposed an ontological model to connect gamification and ITS concepts. The ontology has been manually developed following the Methontology methodology [43]. Sklavakis and Refanidis [44] proposed an interesting framework, namely Mathesis, for ontology-driven development of ITS for Math. They define an ontological representation of both declarative and procedural knowledge of a model-tracing tutor. Moreover, the framework provides knowledge engineering tools integrated into Protege [45] to support the development and management of the knowledge base of the ITS. In all these works, the authors have manually defined the ontological models, often using an ontology editor like Protege.

Unfortunatelly, the manual definition of ontologies is time-consuming and expensive. The process usually requires professionals like knowledge engineers and domain specialists. For these reasons, many approaches, both automatic and semi-automatic, have been defined to support the ontology construction process. The automatic process for building an ontology with a data-driven approach is also called ontology learning. The work of Drumond and Girardi [46] provides an extensive survey of ontology learning techniques, classified according to the type of input used in the learning process: structured data (as databases); semi-structured data (as dictionaries), and unstructured data (natural language text). In our work, we enrich the ontology by using the knowledge contained in a structured dataset regarding tutoring sessions. In the approaches based on structured data, the main issue to solve is the identification of the pieces of structural information that can provide relevant knowledge to obtain a well-defined and useful ontological schema. In our case, we adopt a classification algorithm based on association rule mining to find recurrent patterns in tutoring actions that can be exploited to automatically realize an ITS.

Association Rule Mining (ARM) has been used in scientific literature to build or enrich an existing ontology. Asim *et al.* [47] in a recent survey on ontology learning techniques analyzed some works regarding the use of ARM in the ontology building process. The authors argue that ARM is usually used to find non-taxonomic relations that can enrich an existing ontology. For instance, Idoudi *et al.* [48] use the Apriori algorithm to generate rules used to enrich the ontology in the medical sector. Drynomas *et al.* [49] proposed a similar approach in the computer science domain. Paiva *et al.* [50] and Ghezaiel *et al.* [51] have proposed approaches based on Frequent Pattern growth algorithm to enrich existing ontologies. What seems to emerge by this analysis is that, although approaches able to build an ontology from unstructured corpora could have a great utility in the ontology learning process, they are still far from achieving very good results [46], [47]. As also highlighted by Asim *et al.* [47] and Singh *et al.* [52] the quality of the learned ontologies is usually improved by means of human intervention. This is why we decided to use a template ontology (which is manually built) and then to use an automatic data mining technique to enrich such template ontology. In this way, by starting from a well-defined schema, we obtain an ontological schema of good quality, without the need for post-processing it to improve the quality of the obtained schema. In future work, we will also explore the use of unstructured corpora to automatically build the template ontology.

IV. APPROACH FOR BUILDING TUTORING MODELS

The main idea underlying the approach is to build an ontology that is capable of mapping learners' interaction data with respect to a set of tutoring actions. This mechanism enables an artificial tutor observing a learner in terms of his or her interactions with the learning environment and providing a suitable tutoring action in order to improve the learning process. If the learning environment discretizes the learning process in a sequence of activities, the learner's interaction data comes from the last activity but also includes a set of data aggregating information from the previous activities (i.e., learning history). Whilst typically an ontology is built by humans (i.e., knowledge engineers and domain experts), in this work, an automatic ontology building process is proposed. This process uses a data mining technique to extract classification rules from a dataset containing data regarding real-world tutoring sessions. An ontology-based ITS will use these rules, represented in the form of class restrictions, to automatically decide which is the best tutoring action to perform according to the learner's interactions. The rules will be used in the process of ontology construction. Specifically, the ontology is constructed according to the following process. Starting from the structure of the dataset, a template ontology is manually built to map the structure of the interactions of learners and tutors into classes and properties (further details are in Section V). Then, the process will automatically enrich the ontology by means of the classification rules extracted from the dataset, following the structure of the template ontology. The main phases of the process are described in section IV.B.

Before describing the process in details, the main idea is explained with the following illustrative example. As reported in the previous sections, in order to provide a macro-behavior to the ITS we have adopted the VanLehn's schema that guarantees flexibility to the tutoring model constructed through the proposed approach. Instantiating the VanLehn's schema allows to offer the interactive level (discussing questions with a peer or tutor, solving a problem with a peer or tutor, etc.) in the conceptual framework proposed in [53], in which the author emphasizes the superiority of interactive student activities with respect to constructive, active and passive ones. In particular, in order to mine tutoring rules we need to use data coming from observations of human tutors involved in real tutoring sessions. Of course, this aspect is an advantage because the approach itself is able to capture the experience and competence of human tutors and reflect them into the mined tutoring rules. The other side of the coin is that training data must be of high quality, also from a pedagogical viewpoint, in order to avoid mining poor-quality tutoring rules.

A. ILLUSTRATIVE EXAMPLE

The following illustrative example shows how the OWL-based ontology reasoning selects the right tutoring action for learners. Such (simple) tutoring model needs a class hierarchy.

First, the class `Interaction` as subclass of `owl:Thing` is defined. The individual `Interaction` represents the last interaction of a learner with the learning environment as well as his or her learning history. For the sake of simplicity two properties are defined to express a single interaction individual: `lastAttemptResult` as an Object Property and `attemptsNumber` as a Data Property. Second, the class `Result` (as a subclass of `owl:Thing`) and its two subclasses `PositiveResult` and `NegativeResult` are defined. The first subclass contains the successful individual steps whilst the second subclass contains the failed individual steps.

Third, two additional classes as subclasses of `owl:Thing`: `PositiveFeedback` and `Hint` are defined. The two classes are provided as class restrictions. These two classes represent two examples of the possible types of action a tutor can provide the learners with. Each type of action can be instantiated with specific content. For instance, a Feedback can be Positive or Negative, or it could be motivational or content-based. Therefore, the proposed approach allows defining different types of actions by defining further subclasses. In particular, in this illustrative example, `PositiveFeedback` is defined as equivalent to the rule in Listing 1.

```
attemptsNumber value 1 and
    lastAttemptResult some PositiveResult
attemptsNumber value 2 and
    lastAttemptResult some PositiveResult
```

Listing 1. Rule `PositiveFeedback`.

Such rule means that the computerized tutor must provide a positive feedback if the new interaction data asserts that the learner, on the current task, made only one attempt and the result is positive, or she made two attempts and the last result is positive. Moreover, `Hint` is defined as equivalent to the rule reported in Listing 2.

```
attemptsNumber value 3 and
    lastAttemptResult some NegativeResult
```

Listing 2. Rule `Hint`.

The rule means that the computerized tutor must provide a hint to scaffold the learner if the new interaction data asserts that the learner, on the current task, made three attempts and the last result is still negative. Now, let us show a simple case considering the three new interactions inserted into the ontology as individuals belonging to the class `Interaction`. The characteristics of the individuals representing interactions are shown in Table 1.

TABLE 1. Input data for the illustrative example.

individual	type	attemptsNumber property value	lastAttemptResult property value
interaction01	Interaction	3	not_passed
interaction02	Interaction	1	passed
interaction03	Interaction	3	passed
interaction04	Interaction	2	passed

With this configuration, the ontology reasoner (Hermit 1.3.8.413) correctly classifies the individual `interaction01` to the class `Hint`, and `interaction02` and `interaction04` to the class `PositiveFeedback`. Lastly, for individual `interaction03`, the ontology reasoner cannot classify it to either `Hint` or `PositiveFeedback`. The illustrative example shows how to use an ontology to build a tutoring model. The proposed approach adopts the same mechanism but is capable of automatically building the classification rules (class restrictions) by mining past interaction-tutoring data.

B. PROPOSED APPROACH: OVERALL WORKFLOW

Two sub-workflows are involved in the proposed approach. The first one consists in i) building the tutoring model by exploiting data coming from a set of real-world tutoring sessions, ii) applying an association rule mining algorithm on it in order to extract classification rules and iii) enriching an existing ontology with formal definitions of concepts representing tutoring actions. The second one is used to deal with a new interaction (from the learner) and respond with a suitable tutoring action.

Fig. 2 shows the first sub-workflow that has three main phases: dataset preparation, CPAR application and ontology enrichment. The second sub-workflow is simpler. It uses the enriched ontology, inserts new individuals representing new interaction data in it and starts a reasoner to classify such individuals correctly in the classes representing tutoring actions as well as the illustrative example in Section IV-A.

V. DATASET PREPARATION

The first phase in the sub-workflow of building the tutoring model gathers data for each task the learner is involved through observing the real-world tutoring sessions. In particular, four types of data are collected for each task instance: i) task information (e.g., difficulty level of the task, number of the step in the task), ii) learner's response (e.g., correctness, time-to-answer), iii) learner's history (e.g., number of attempts to the same step, level of the received hint), and iv) tutoring action (e.g., feedback, hint, action).

All these data, for a specific task instance, are put together in the same row of the output dataset. Moreover this phase accomplishes also a mapping between the entities in a template ontology and the elements (column names and values) of the constructed dataset. Such mapping is useful to give additional information to the rule mining algorithm applied in the next phase. Take a look to the illustrative example in Section IV-A. With this case we have a reduced sample

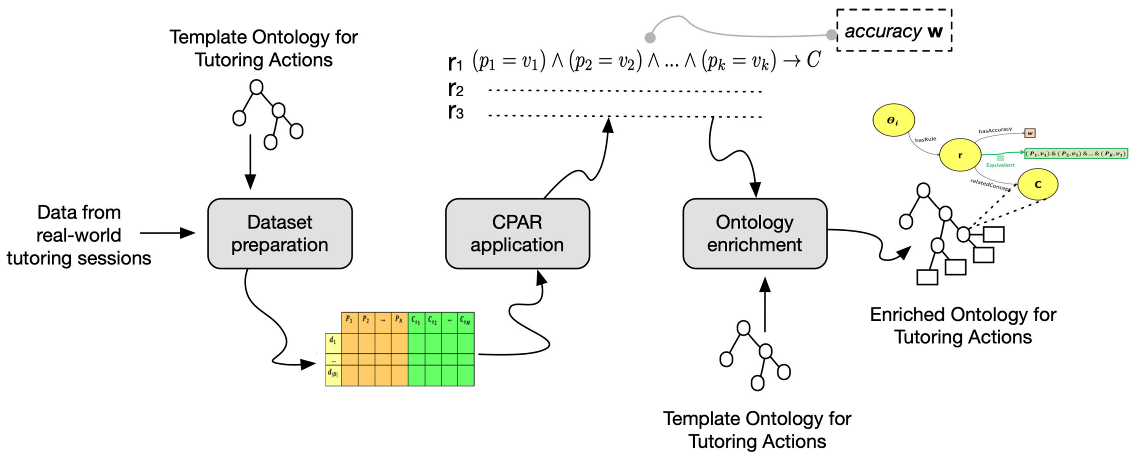

Template Ontology for Tutoring Actions
Data from real-world tutoring sessions
Dataset preparation
CPAR application
Ontology enrichment
accuracy w
r1 $(p_1 = v_1) \wedge (p_2 = v_2) \wedge ... \wedge (p_k = v_k) \rightarrow C$
r2
r3
Enriched Ontology for Tutoring Actions
Template Ontology for Tutoring Actions

FIGURE 2. First sub-workflow of the approach: building the tutoring model.

TABLE 2. Mapping between the dataset and the template ontology.

Dataset features	Template Ontology
result	`lastAttemptResult`
attempts	`attemptsNumber`
feedback: positive feedback	`PositiveFeedback`
feedback: negative feedback	`NegativeFeedback`
hint: no hint	`NoHint`
hint: hint level 1	`HintLevel1`

dataset with four columns (*result*, *attempts*, *feedback*, *hint*), two admissible values (*positive feedback* and *negative feedback*) for the column *feedback* and two admissible values (*hint level 1* and *no hint*) for column *hint*. Table 2 contains the mapping between the dataset features and the classes and properties of the template ontology. Specifically, the first two rows of Table 2 map features (condition features) of the dataset into properties of the template ontology. Whilst the last two elements map features (decision features) of the dataset into classes of the template ontology.

A consistent set of features (both condition and decision) is that used for the case study (see Section VIII). Now, for the sake of generality, assume to have a dataset D (previously, in Section II-B, we have indicated such set as T) with features $A = \{A_1, A_2, \ldots, A_n\}$, as depicted in Fig. 3.

Let $V(A_i) \in V_i = \{v_i^1, v_i^2, \ldots v_i^{|V_i|}\}$ be the set of possible values for the attribute A_i. Let $A = Co \cup De$, where Co is the set of condition features and $De = \{L_1, L_2, \ldots, L_m\}$ is the set of decision features, and $V(De) = \bigcup_{L \in De} V(L)$.

Let O be a template ontology modelled in OWL and similar to that of the illustrative example in Section IV-A. O includes several entities but those important for the mapping are the set of OWL classes $O_{classes} = \{C_1, C_2, \ldots, C_u\}$ representing tutoring actions and the set of OWL `DataTypeProperties / ObjectTypeProperties` $O_{properties} = \{p_1, p_2, \ldots, p_v\}$ representing condition features used to determine how to classify new interaction data with respect to the aforementioned OWL classes.

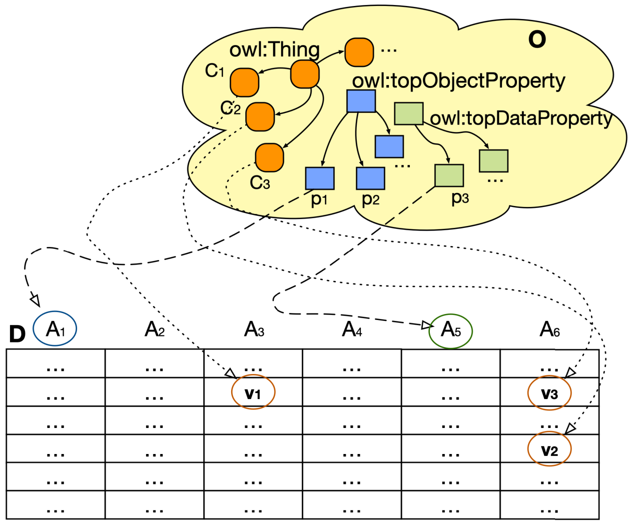


FIGURE 3. A mapping between a generic dataset and a template ontology.

The mappings $\varphi_1(D) \subseteq (O_{properties} \times Co)$ and $\varphi_2(D) \subseteq (O_{classes} \times V(De))$ could be performed by humans or artificial (software) agents working on the textual descriptions of properties and classes. φ_1 and φ_2 are two functions whose domains are the set of all possible datasets. In particular, it is needed to associate condition attributes of the dataset D with properties in the ontology and distinct values of decision attributes in the dataset with classes in the ontology. If you look at Fig. 3, plausible mapping instances are:

$$\varphi_1(D) = \{(p_1, A_1), (p_3, A_5)\} \tag{4}$$

$$\varphi_2(D) = \{(C_1, A_3 : v_1), (C_2, A_6 : v_2), (C_3, A_6 : v_3)\} \tag{5}$$

By considering φ_2, we can say that we have two decision features, i.e., $A_3 \in De \subset A$ and $A_6 \in De \subset A$, whose distinct values $v_1 \in V_3$, $v_2 \in V_6$ and $v_3 \in V_6$ can be consequents in classification rules. Lastly, by considering φ_1 it is possible to affirm that we have two condition features, i.e., p_1 and p_3 which can be parts of the antecedents in the aforementioned rules.

VI. CLASSIFICATION BASED ON PREDICTIVE ASSOCIATION RULES (CPAR) APPLICATION

The second phase is committed to run the CPAR rule mining algorithm and to construct the mappings instances φ_1 and φ_2 on the dataset D.

The output of this phase is a set of classification rules (a simple example of such rules is provided in Section IV-A) mined from data coming from real-world tutoring sessions. Assume that the original dataset is D, the features (attributes) in D are $A = Co \cup De$, and the cardinality of De is $|De| = m$.

The CPAR algorithm foresees only one decision feature in the input dataset, thus the original dataset D must be decomposed in m datasets, where m is the number of decision features in D, i.e., we have m ontology classes mapped onto the attributes of D. Therefore, we will have m datasets $D_1, \ldots, D_m$, where the attribute (or feature) set in D_i (for $i = 1, \ldots, m$) is $Co \cup L_i$. Once obtained the m datasets, CPAR can be applied on $\varphi_1(D_i)$ and $\varphi_2(D_i)$ (the application of the mappings on the i-th dataset in order to handle ontology elements in the place of dataset features and values) for all $i = 1, \ldots, m$ in order to obtain m sets of rules, namely $R_1, \ldots, R_m$. Being $\bar{R} = \{R_1, R_2, \ldots, R_m\}$ the union of the m sets of rules.

Moreover, for each $R \in \bar{R}$, the generic rule in this set has the following form:

$$r : (p_1 = v_1) \wedge (p_2 = v_2) \wedge \ldots \wedge (p_k = v_k) \rightarrow C \quad (6)$$

where p_i is a property in the ontology O and $(p_i, A_j) \in \varphi_1(D)$, where $A_j \in A$. Moreover $v_i \in V(A_j)$ and $V(A_j)$, as written before, is the set of all admissible values used in the dataset D for the feature A_j. The operator $\wedge$ is the logical operator *and*. Additionally, the consequent part of the rule is represented by C that is a class in O such as $(C, L : v) \in \varphi_2(D)$, where $v \in V(L)$ and $L \in De$. Lastly, $w = \omega(r)$ is the accuracy (calculated by using the Laplace measure described in Eq.3) of the rule and R the set of rules to which r belongs. In general, the rule r classifies a new learner's interaction with class C if the interaction data satisfies the conditions in such a rule. Interaction data comes with the form of a vector of values associated to the features Co. In the case we have more than one rule with the same antecedent, in order to eliminate the redundancy, it is possible to put in the final output only the rule with the greater accuracy.

VII. ONTOLOGY ENRICHMENT

The third phase is used to insert the rules, coming from the previous phase, into the template ontology in order to enable the inference mechanism to classify new interaction data with respect to suitable tutoring actions. As already stated before, the template ontology contains classes representing tutoring actions and properties (both data and object type) representing learners' interaction attributes and other characteristics that can be mapped on the provided dataset (see Section V). Classes and properties that have to be considered are those involved in the mapping realized during the preparation of the dataset.

The preliminary step of this phase is realized by using two filtering operations. The first one is realized by dropping from all $R \in \bar{R}$ the rules having accuracy less than a given threshold.

The second one is executed to eliminate inconsistencies from the sets of rules. In particular, we need to handle the cases in which rules, with overlapping antecedents, have different consequents. The adopted method must be applied to all $R \in \bar{R}$ and is described below. Assume that a rule $r \in R$ is represented as the tuple (X, Y, w), where X is the antecedent of the rule, Y is the consequent of the rule and w is the accuracy. The algorithm to filter (second filtering operation) is shown in Listing 3.

```
$R' \leftarrow \emptyset$
$T \leftarrow \emptyset$
for $r$ **in** $R$:
    if **not** $r$ **in** $T$:
        $Q'$.append($r$)
        $I \leftarrow$ findInconsistencies($r$)
        $Q'$.extend($I$)
        $\bar{r}$ = selectBestRule($Q'$)
        $R'$.append($\bar{r}$)
        $T$.append($Q' - \{\bar{r}\}$)
```

Listing 3. Algorithm for eliminating the inconsistencies from the rules.

The algorithm returns the filtered set R' and must be applied to each $R \in \bar{R}$. The function `findInconsistencies` applied on r is able to find all the rules $r' = (X', Y', w')$ in R such as:

- $X \subseteq X'$ or $X' \subseteq X$, and
- $Y \neq Y'$.

The function `selectBestRule` applied on a set of inconsistent rules Q' returns the most suitable rule in such a set. A plausible policy to define the most suitable rule in a set of inconsistent rules is the following one: select the rule with the greatest accuracy and, in the case of more than one rule with the maximum accuracy, select the one with less terms (with respect to the conjunctive form).

Once constructed $\bar{R}' = \{R'_1, R'_2, \ldots, R'_m\}$ it is possible to enrich the template ontology. The changes to be applied to the template ontology can be executed by using a framework like *Apache Jena*.[3] The procedure described in Listing 4 must be applied for all $R \in \bar{R}'$. Assume that the form of the generic rule r in R is $\{(p^r_1 = v^r_1) \wedge (p^r_2 = v^r_2) \wedge \ldots \wedge \left(p^r_{k_r} = v^r_{k_r}\right) \rightarrow C\}$. For each $r \in R$ we have to modify the ontology by enriching the definition of class C (in the ontology) using Listing 4. Additional rules $r' \in R$ with the same consequent C can be added to the above definition of the class C. Different rules definitions must be comma-separated.

[3]https://jena.apache.org/documentation/ontology/

TABLE 3. Fragment of the dataset used for the case study.

taskLevel	step	outcome	time	attempt	isLastStep	hintReceived	feedback	hint	action
TaskLevel#1	StepNumber#1	OutcomeCORRECT	[0-9]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#2	OutcomeCORRECT	[0-9]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#3	OutcomeCORRECT	[0-9]	1stAttempt	true	NoHintReceived	PositiveFeedback	NoHint	Reward
TaskLevel#2	StepNumber#1	OutcomeCORRECT	[10-14]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#2	StepNumber#2	OutcomeCORRECT	[10-14]	1stAttempt	true	NoHintReceived	PositiveFeedback	NoHint	Reward
TaskLevel#3	StepNumber#1	OutcomeCORRECT	[10-14]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#3	StepNumber#2	OutcomeCORRECT	[0-9]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#3	StepNumber#3	OutcomeCORRECT	[0-9]	1stAttempt	true	NoHintReceived	PositiveFeedback	NoHint	Reward
TaskLevel#1	StepNumber#1	OutcomeERROR	[10-14]	1stAttempt	false	NoHintReceived	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#1	OutcomeERROR	[0-9]	2ndAttempt	false	NoHintReceived	SimpleNegativeFeedback	HintLevel#1	NoAdditionalAction
TaskLevel#1	StepNumber#1	OutcomeCORRECT	[10-14]	3rdAttempt	false	HintLevel#1Received	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#2	OutcomeERROR	[10-14]	1stAttempt	false	NoHintReceived	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#2	OutcomeERROR	[0-9]	2ndAttempt	false	NoHintReceived	SimpleNegativeFeedback	HintLevel#1	NoAdditionalAction
TaskLevel#1	StepNumber#2	OutcomeCORRECT	[0-9]	3rdAttempt	false	HintLevel#1Received	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#3	OutcomeERROR	[15-25]	1stAttempt	true	NoHintReceived	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#3	OutcomeERROR	[15-25]	2ndAttempt	true	NoHintReceived	SimpleNegativeFeedback	HintLevel#1	NoAdditionalAction

```
Class: <ontoName#$C$>
    EquivalentTo:
      (ontoName#$p_1^r$> value $v_1^r$)
      and (ontoName#$p_2^r$> value $v_2^r$)
      and ...
      and (ontoName#$p_{k_r}^r$> value $v_{k_r}^r$) [,]
          [...]
```

Listing 4. Enriching the definition of class C using the rules $r \in R$.

The previous OWL code (written in the Manchester Syntax) fragment handles the case of OWL DataType properties. In order to handle the case of OWL ObjectType properties, we need to put into the template ontology some additional elements to support the mapping of an attribute of the dataset with the ObjectType property, but the approach is almost the same as shown in Listing 4. Clearly, it is possible to combine DataType and ObjectType properties in the same rule. After this operation is finished, the template ontology is instantiated and the ontology-driven tutoring model is ready to infer tutoring actions for new incoming interaction data.

VIII. CASE STUDY: INF@NZIA DIGI.TALES 3.6 PROJECT

In order to validate the proposed approach a case study is proposed. It is important to underline that the aim of the case study is to show the results of the proposed approach in the case of a dataset collected in a real-world context. Instead, the validity of the resulting enriched ontology is demonstrated by applying an automatic reasoner for executing rules without errors. The case study adopts a dataset gathered in the context of an Italian University and Research Ministry co-funded project, namely *INF@NZIA DIGI.Tales 3.6* [54]–[56]. In particular, the dataset has been collected by the University of Salerno [1] in the context of further experimentation, whose results are reported in the work [57] that was partially supported by the same aforementioned project. In particular, that work provides more details about the experimentation of an interactive game-based Edu App, namely *Bigfoot the pedestrian*, based on Augmented Reality. Instead, for the case study reported in this Section, a different Edu App [17] adopting the behavior of an ITS (Section II-C) has been used as a learning environment for 5 to 6s years old children. The interactions of children with the Edu App and the corresponding tutoring actions were traced by the support staff. The role of the tutor was played by primary school teachers (from the Educational Institutions involved by the University of Salerno). Subsequently, such traces have been pre-processed to construct the dataset used to train the tutoring model of the ITS for the aim of this case study. A fragment of the aforementioned dataset is reported in Table 3. With respect to the composition of the dataset, it records a whole ITS internal loop including: level of difficulty of the executed task (*taskLevel*), step number in the task (*step*), correctness of the learner's answer to the task (*outcome*), time spent to provide the answer (*time*), count of the attempts (*attempt*), if the current step is the last step for the task (*isLastStep*), level of the received hint (*hint*), feedback provided by the tutor taking into account the correctness of the learner's answer (*feedback*), hint provided by the tutor to help the learner for the next attempt (*hint*) and additional action provided by the tutor to motivate and/or engage the learner (*action*).

The template ontology prepared for the case study is partially depicted in Fig. 4 obtained by using the well-known ontology editor *Protégé* [45]. In the aforementioned figure it is possible to observe the classes representing decision features and properties (all DataType properties in this case)

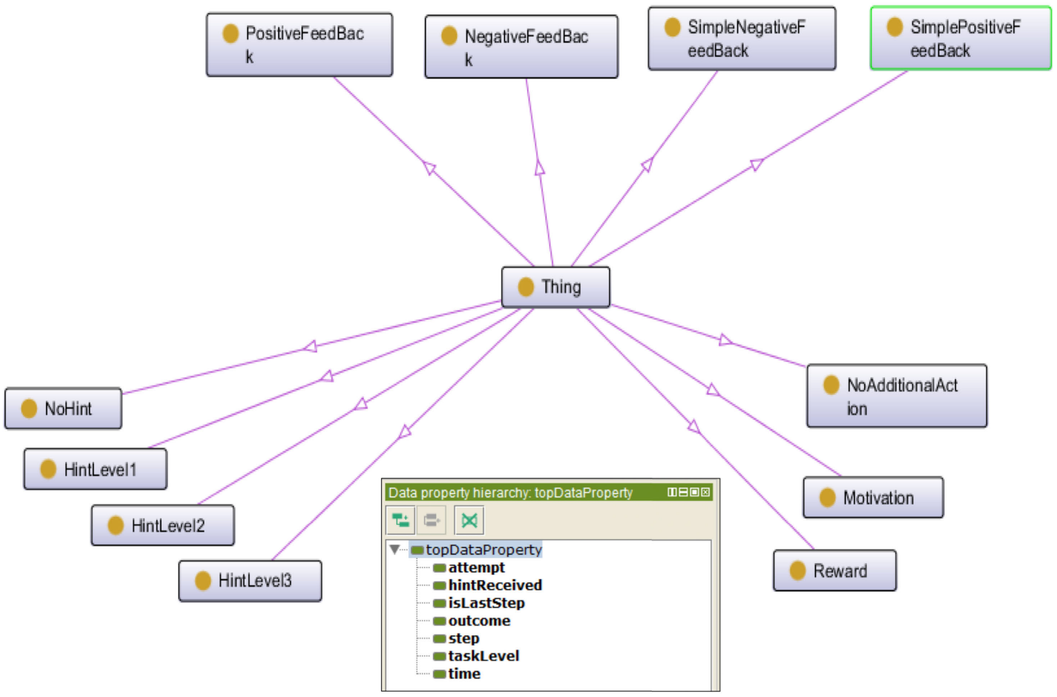
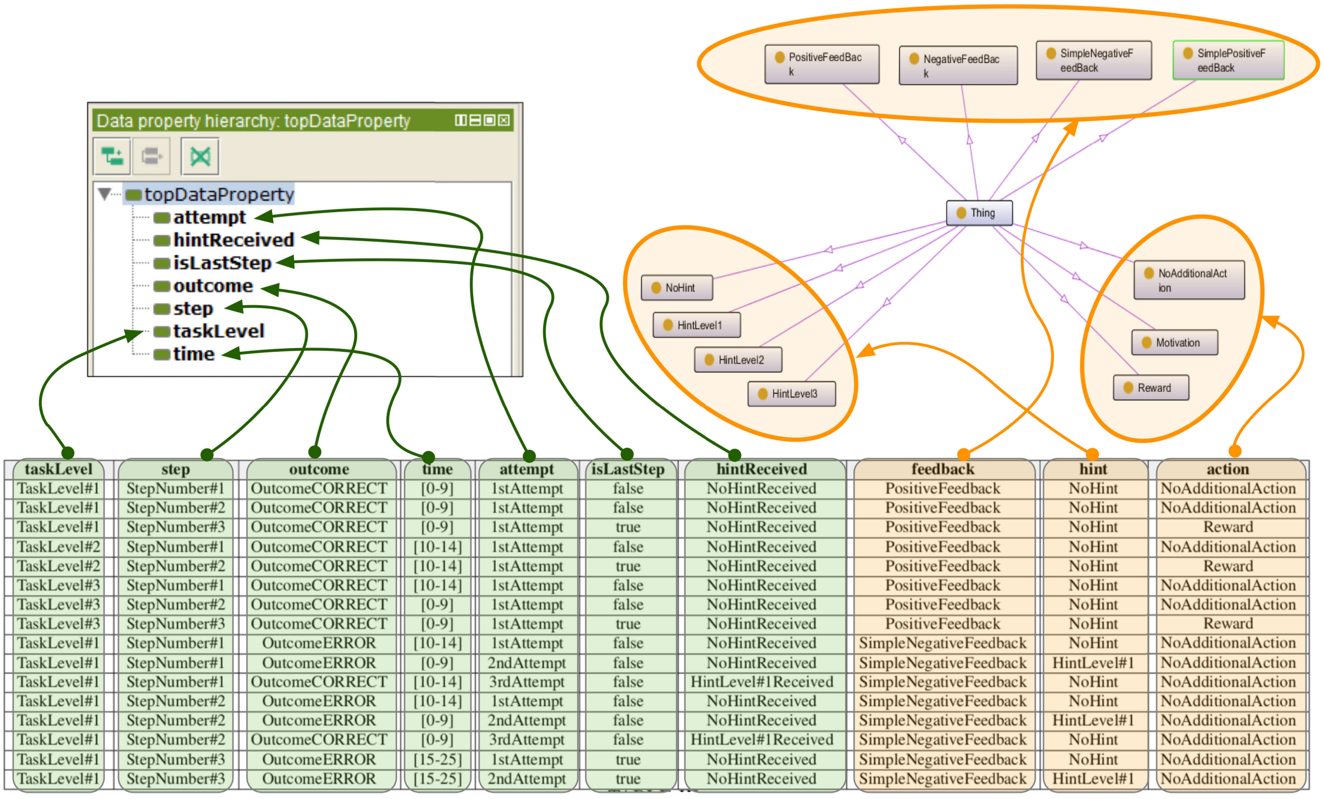


FIGURE 4. Part of the template ontology defined for the case study.



taskLevel	step	outcome	time	attempt	isLastStep	hintReceived	feedback	hint	action
TaskLevel#1	StepNumber#1	OutcomeCORRECT	[0-9]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#2	OutcomeCORRECT	[0-9]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#3	OutcomeCORRECT	[0-9]	1stAttempt	true	NoHintReceived	PositiveFeedback	NoHint	Reward
TaskLevel#2	StepNumber#1	OutcomeCORRECT	[10-14]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#2	StepNumber#2	OutcomeCORRECT	[10-14]	1stAttempt	true	NoHintReceived	PositiveFeedback	NoHint	Reward
TaskLevel#3	StepNumber#1	OutcomeCORRECT	[10-14]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#3	StepNumber#2	OutcomeCORRECT	[0-9]	1stAttempt	false	NoHintReceived	PositiveFeedback	NoHint	NoAdditionalAction
TaskLevel#3	StepNumber#3	OutcomeCORRECT	[0-9]	1stAttempt	true	NoHintReceived	PositiveFeedback	NoHint	Reward
TaskLevel#1	StepNumber#1	OutcomeERROR	[10-14]	1stAttempt	false	NoHintReceived	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#1	OutcomeERROR	[0-9]	2ndAttempt	false	NoHintReceived	SimpleNegativeFeedback	HintLevel#1	NoAdditionalAction
TaskLevel#1	StepNumber#1	OutcomeCORRECT	[10-14]	3rdAttempt	false	HintLevel#1Received	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#2	OutcomeERROR	[10-14]	1stAttempt	false	NoHintReceived	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#2	OutcomeERROR	[0-9]	2ndAttempt	false	NoHintReceived	SimpleNegativeFeedback	HintLevel#1	NoAdditionalAction
TaskLevel#1	StepNumber#2	OutcomeCORRECT	[0-9]	3rdAttempt	false	HintLevel#1Received	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#3	OutcomeERROR	[15-25]	1stAttempt	true	NoHintReceived	SimpleNegativeFeedback	NoHint	NoAdditionalAction
TaskLevel#1	StepNumber#3	OutcomeERROR	[15-25]	2ndAttempt	true	NoHintReceived	SimpleNegativeFeedback	HintLevel#1	NoAdditionalAction

FIGURE 5. Mapping of the dataset to the template ontology for the case study.

representing condition features. Fig. 5 depicts the mapping between the dataset of Table 3 and the template ontology of Fig. 4. After the execution of the first two phases of the proposed approach, 27 rules have been generated for decision attributes `Feedback`, `Hint` and `Action`. After the filtering operations (assuming we have a cut at threshold 0.78) the rules to be inserted in the template ontology (see Fig. 4) are reported in the Tables 4, 5 and 6.

TABLE 4. Case study: part of the generated rules for class feedback.

rule for Feedback	accuracy
outcome: OutcomeERROR & hintReceived: NoHintReceived ->NegativeFeedBack	0.85
outcome: OutcomeERROR & attempt: 1stAttempt -> NegativeFeedBack	0.84
outcome: OutcomeERROR & attempt: 2ndAttempt -> NegativeFeedBack	0.80
outcome: OutcomeCORRECT & time: [15-25] -> PositiveFeedBack	0.78

TABLE 5. Case study: part of the generated rules for class hint.

rule for Hint	accuracy
outcome: OutcomeCORRECT & time: [10-14] -> NoHint	0.82
outcome: OutcomeCORRECT & attempt: 1stAttempt ->NoHint	0.82
outcome: OutcomeCORRECT & hintReceived: NoHintReceived ->NoHint	0.82
outcome: OutcomeERROR & attempt: 3rdAttempt -> HintLevel1	0.78
attempt: 4th+Attempt ->HintLevel2	0.78

TABLE 6. Case study: part of the generated rules for class action.

rule for Action	accuracy
outcome: OutcomeERROR & attempt: 1stAttempt -> NoAdditionalAction	0.89
attempt: 1stAttempt & isLastStep: false -> NoAdditionalAction	0.89
attempt: 2ndAttempt ->NoAdditionalAction	0.88
outcome: OutcomeERROR & attempt: 3rdAttempt -> Motivation	0.78

The aforementioned rules have been inserted in the template ontology and a fully operable ontology is now available to provide tutoring actions. In order to avoid cases in which new interactions have not matches with rules' antecedents in the rule base written in the ontology, it is possible and plausible to adopt a simple strategy that enables the tutoring model to answer with default tutoring actions based on the value of the *outcome* attribute. Let us show an example of execution of the created ontology-driven tutoring model. If the new incoming interaction data are (taskLevel: TaskLevel#3), (step: StepNumber#3), (outcome: OutcomeERROR), (time: 9), (attempt: 3rdAttempt), (isLastStep: True), (hintReceived: NoHintReceived) the tutoring model provides the following tutoring actions: (Feedback, NegativeFeedback), (Hint, HintLevel1), (Action, Motivation).

The final enriched ontology has been validated in two ways. First, we used the OWL Validator Tool[4] of the University of Manchester to check that the ontology is syntactically correct. Second, we checked that the ontology is consistent and formally correct using a logical (rule-based) approach [58], which means that an OWL Reasoner (Hermit[5]) has been used to check that there are no conflicts in the ontology and that it is formally correct. Both the validations were successfully demonstrating that the approach produces well-defined ontology that can be used to support reasoning and inference in ITS.

[4]http://visualdataweb.de/validator/
[5]http://www.hermit-reasoner.com/

IX. FINAL REMARKS

The paper discusses an original approach for the automatic building of ontology-driven tutoring models. The proposed approach offers the benefit of representing the rules of the tutoring model by exploiting ontology languages (OWL), i.e., obtaining a scrutable model changeable by both humans and artificial agents, and, at the same time, avoiding the drawbacks of ontology engineering. In fact, it is possible to define ontology rules by applying a data mining technique known as CPAR that is able to extract rules from data coming from real-world tutoring sessions. The first results are promising and in future works the authors will implement more complex strategies to handle inconsistent rules and default actions.

REFERENCES

[1] G. Fenza and F. Orciuoli, "Building pedagogical models by formal concept analysis," in *Proc. 13th Int. Conf. Intell. Tutoring Syst. (ITS)*, vol. 9684, 2016, pp. 144–153.
[2] A. C. Graesser, X. Hu, and R. Sottilare, "Intelligent tutoring systems," in *International Handbook of the Learning Sciences*. Evanston, IL, USA: Routledge, 2018, pp. 246–255.
[3] M. Ponticorvo, A. Rega, and O. Miglino, "Toward tutoring systems inspired by applied behavioral analysis," in *Intelligent Tutoring Systems* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2018, pp. 160–169.
[4] M. C. Polson and J. J. Richardson, *Foundations of Intelligent Tutoring Systems*. New York, NY, USA: Psychology Press, 2013.
[5] G. Adorni, S. Battigelli, D. Brondo, N. Capuano, M. Coccoli, S. Miranda, F. Orciuoli, L. Stanganelli, A. Sugliano, and G. Vivanet, "CADDIE and IWT: Two different ontology-based approaches to anytime, anywhere and anybody learning," *J. e-Learn. Knowl. Soc.*, vol. 6, no. 2, pp. 53–66, 2010.
[6] N. Capuano, M. Gaeta, S. Miranda, F. Orciuoli, and P. Ritrovato, "LIA: An intelligent advisor for e-learning," in *Proc. 1st World Summit Knowl. Soc. (WSKS)*, Athens, Greece, vol. 5288, 2008, pp. 187–196.
[7] R. Rajendran, S. Iyer, and S. Murthy, "Personalized affective feedback to address students' frustration in ITS," *IEEE Trans. Learn. Technol.*, vol. 12, no. 1, pp. 87–97, Jan. 2019.
[8] R. Rajendran, S. Iyer, S. Murthy, C. Wilson, and J. Sheard, "A theory-driven approach to predict frustration in an ITS," *IEEE Trans. Learn. Technol.*, vol. 6, no. 4, pp. 378–388, Oct. 2013.
[9] K. VanLehn, J. Wetzel, S. Grover, and B. V. D. Sande, "Learning how to construct models of dynamic systems: An initial evaluation of the dragoon intelligent tutoring system," *IEEE Trans. Learn. Technol.*, vol. 10, no. 2, pp. 154–167, Apr. 2017.
[10] F. Colace, M. D. Santo, M. Lombardi, F. Pascale, A. Pietrosanto, and S. Lemma, "Chatbot for e-learning: A case of study," *Int. J. Mech. Eng. Robot. Res.*, vol. 7, no. 5, pp. 528–533, 2018.
[11] D. Arnau, M. Arevalillo-Herráez, and J. A. Gonzalez-Calero, "Emulating human supervision in an intelligent tutoring system for arithmetical problem solving," *IEEE Trans. Learn. Technol.*, vol. 7, no. 2, pp. 155–164, Apr. 2014.
[12] G. D'Aniello, M. Gaeta, V. Loia, F. Orciuoli, and D. Sampson, "Situation awareness enabling decision support in seamless learning," in *Proc. Int. Conf. Intell. Netw. Collaborative Syst.*, Sep. 2015, pp. 440–445.
[13] G. D'Aniello, A. Gaeta, M. Gaeta, and S. Tomasiello, "Self-regulated learning with approximate reasoning and situation awareness," *J. Ambient Intell. Hum. Comput.*, vol. 9, no. 1, pp. 151–164, Feb. 2018.
[14] E. El-Sheikh and J. Sticklen, "A framework for developing intelligent tutoring systems incorporating reusability," in *Methodology and Tools in Knowledge-Based Systems*. Berlin, Germany: Springer, 1998, pp. 558–567.
[15] R. Nkambou, R. Mizoguchi, and J. Bourdeau, *Advances in Intelligent Tutoring Systems*, vol. 308. Springer, 2010.
[16] R. Mizoguchi, J. Vanwelkenhuysen, and M. Ikeda, "Task ontology for reuse of problem solving knowledge," in *Towards Very Large Knowledge Bases: Knowledge Building & Knowledge Sharing*. Amsterdam, The Netherlands: IOS Press, 1995, pp. 46–59.

[17] V. Centola and F. Orciuoli, "ITSEGO: An ontology for game-based intelligent tutoring systems," in *Proc. 8th Int. Conf. Comput. Supported Educ. (CSEDU)*, vol. 1, 2016, pp. 238–245.

[18] M. M. Hilles and S. S. A. Naser, "Knowledge-based intelligent tutoring system for teaching mongo database," *Eur. Acad. Res.*, vol. 4, no. 10, pp. 1–12, 2017.

[19] G. Fenza, V. Loia, and F. Orciuoli, "Providing smart objects with intelligent tutoring capabilities by semantic technologies," in *Proc. Int. Conf. Intell. Netw. Collaborative Syst. (INCoS)*, Sep. 2016, pp. 103–109.

[20] S. Cetintas, L. Si, Y. Ping Xin, and C. Hord, "Automatic detection of off-task behaviors in intelligent tutoring systems with machine learning techniques," *IEEE Trans. Learn. Technol.*, vol. 3, no. 3, pp. 228–236, Jul. 2010.

[21] J. Chen and M. Reformat, "Learning categories from linked open data," *Commun. Comput. Inf. Sci.*, vol. 444, no. 3, pp. 396–405, 2014.

[22] H. A. M. Hassan, G. Sansonetti, F. Gasparetti, and A. Micarelli, "Semantic-based tag recommendation in scientific bookmarking systems," in *Proc. 12th ACM Conf. Rec. Syst. (RecSys)*, 2018, pp. 465–469.

[23] R. O. A. Paiva, I. I. B. S. Pinto, A. P. da Silva, S. Isotani, and P. Jaques, "A systematic approach for providing personalized pedagogical recommendations based on educational data mining," in *Intelligent Tutoring Systems*. Cham, Switzerland: Springer, 2014, pp. 362–367.

[24] F. Gasparetti, A. Micarelli, and G. Sansonetti, "Exploiting Web browsing activities for user needs identification," in *Proc. Int. Conf. Comput. Sci. Comput. Intell.*, vol. 2, Mar. 2014, pp. 86–89.

[25] G. Fenza, F. Orciuoli, and D. G. Sampson, "Building adaptive tutoring model using artificial neural networks and reinforcement learning," in *Proc. IEEE 17th Int. Conf. Adv. Learn. Technol. (ICALT)*, Jul. 2017, pp. 460–462.

[26] G. George and A. M. Lal, "Review of ontology-based recommender systems in e-learning," *Comput. Educ.*, vol. 142, Dec. 2019, Art. no. 103642.

[27] A. Seeliger, M. Pfaff, and H. Krcmar, "Semantic Web technologies for explainable machine learning models: A literature review," in *Proc. PROFILES SEMEX, 1st Workshop Semantic Explainability (SemEx), Co-Located 18th Int. Semantic Web Conf. (ISWC)*, 2019, pp. 30–45.

[28] J. Kim and J. Seo, "Human understandable explanation extraction for black-box classification models based on matrix factorization," pp. 1–8, Sep. 2017, *arXiv:1709.06201*. [Online]. Available: https://arxiv.org/abs/1709.06201

[29] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM Comput. Surv.*, vol. 51, no. 5, pp. 1–42, Aug. 2018.

[30] M. Krötzsch, F. Simancik, and I. Horrocks, "Description logics," *IEEE Intell. Syst.*, vol. 29, no. 1, pp. 12–19, Jan./Feb. 2014.

[31] J. Heflin, "An introduction to the OWL Web ontology language," Lehigh Univ. Nat. Sci. Found., Jan. 2007, pp. 1–24. [Online]. Available: http://www.cse.lehigh.edu/~heflin/IntroToOWL.pdf

[32] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *ACM SIGMOD Rec.*, vol. 22, no. 2, pp. 207–216, Jun. 1993.

[33] B. Liu, W. Hsu, and Y. Ma, "Integrating classification and association rule mining," in *Proc. 4th Int. Conf. Knowl. Discovery Data Mining*, 1998, pp. 1–7.

[34] X. Yin and J. Han, "CPAR: Classification based on predictive association rules," in *Proc. SIAM Int. Conf. Data Mining*, May 2003, pp. 331–335.

[35] K. Vanlehn, "The behavior of tutoring systems," *IJ Artif. Intell. Educ.*, vol. 16, no. 3, pp. 227–265, 2006.

[36] B. du Boulay, "Commentary on Kurt VanLehn's the behaviour of tutoring systems," *IJ Artif. Intell. Educ.*, vol. 16, no. 3, pp. 267–270, 2006.

[37] E. Oguejiofor, R. P. Kicinger, E. Popovici, T. Arciszewski, and K. A. De Jong, "Intelligent tutoring systems: An ontology-based approach," *Int. J. IT Archit., Eng. Construct.*, vol. 2, no. 2, pp. 1–14, 2004.

[38] I. Garcia, C. Benavides, H. Alaiz, J. Alfonso, J. Alija, and A. Alonso, "A semantic Web-based learning environment for control engineering education," *IFAC Proc. Volumes*, vol. 42, no. 24, pp. 262–267, 2010.

[39] P. Stamatis, I. Panagiotpoulos, C. Goumopoulos, and A. Kameas, "Combining agents and ontologies for building an intelligent tutoring system," in *Proc. CSEDU Conf.*, 2015, pp. 1–10.

[40] B. Vesin, M. Ivanović, A. Klašnja-Milićević, and Z. Budimac, "Protus 2.0: Ontology-based semantic recommendation in programming tutoring system," *Expert Syst. Appl.*, vol. 39, no. 15, pp. 12229–12246, Nov. 2012.

[41] M. Rani, R. Nayak, and O. P. Vyas, "An ontology-based adaptive personalized e-learning system, assisted by software agents on cloud storage," *Knowl.-Based Syst.*, vol. 90, pp. 33–48, Dec. 2015.

[42] D. Dermeval, J. Albuquerque, I. I. Bittencourt, S. Isotani, A. P. Silva, and J. Vassileva, "GaTO: An ontological model to apply gamification in intelligent tutoring systems," *Frontiers Artif. Intell.*, vol. 2, p. 13, Jul. 2019.

[43] M. F. Lopez, A. Gomez-Perez, and N. Juristo, "Methontology: From ontological art towards ontological engineering," in *Proc. Ontol. Eng. AAAI Spring Symp. Ser.* Menlo Park, CA, USA: American Asociation for Artificial Intelligence, Mar. 1997, pp. 1–8.

[44] D. Sklavakis and I. Refanidis, "The MATHESIS meta-knowledge engineering framework: Ontology-driven development of intelligent tutoring systems," *Appl. Ontol.*, vol. 9, nos. 3–4, pp. 237–265, 2014.

[45] M. A. Musen, "The protégé project: A look back and a look forward," *AI Matters*, vol. 1, no. 4, pp. 4–12, Jun. 2015.

[46] L. Drumond and R. Girardi, "A survey of ontology learning procedures," *WONTO*, vol. 427, pp. 1–13, Oct. 2008.

[47] M. N. Asim, M. Wasim, M. U. G. Khan, W. Mahmood, and H. M. Abbasi, "A survey of ontology learning techniques and applications," *Database*, vol. 2018, pp. 1–24, Jan. 2018.

[48] R. Idoudi, K. S. Ettabaâ, B. Solaiman, K. Hamrouni, and N. Mnif, "Association rules based ontology enrichment," *Int. J. Web Appl.*, vol. 8, no. 1, pp. 16–25, 2016.

[49] E. Drymonas, K. Zervanou, and E. G. Petrakis, "Unsupervised ontology acquisition from plain texts: The ontogain system," in *Proc. Int. Conf. Appl. Natural Lang. Inf. Syst.* Berlin, Germany: Springer, 2010, pp. 277–287.

[50] L. Paiva, R. Costa, P. Figueiras, and C. Lima, "Discovering semantic relations from unstructured data for ontology enrichment: Asssociation rules based approach," in *Proc. 9th Iberian Conf. Inf. Syst. Technol. (CISTI)*, Jun. 2014, pp. 1–6.

[51] L. B. Ghezaiel, C. Latiri, and M. B. Ahmed, "Ontology enrichment based on generic basis of association rules for conceptual document indexing," in *Proc. Int. Conf. Knowl. Eng. Ontol. Develop. (KEOD)*, vol. 1, 2012, pp. 53–65.

[52] S. Singh and M. S. Aswal, "Ontology learning procedures based on Web mining techniques," in *Proc. Int. Conf. Adv. Eng. Sci. Manage. Technol. (ICAESMT)*. Dehradun, India: Uttaranchal Univ., Mar. 2019, pp. 1–7.

[53] M. T. H. Chi, "Active-constructive-interactive: A conceptual framework for differentiating learning activities," *Topics Cognit. Sci.*, vol. 1, no. 1, pp. 73–105, Jan. 2009.

[54] O. Miglino, M. Ponticorvo, and L. S. Sica, "Theoretical perspectives of hands-on educational practices—From a review of psychological theories to block magic and INF@NZIA DIGI.Tales 3.6 projects," in *E-Learning: Instructional Design, Organizational Strategy and Management*. 2015.

[55] M. Ponticorvo, R. Di Fuccio, F. Ferrara, A. Rega, and O. Miglino, "Multisensory educational materials: Five senses to learn," in *Methodologies and Intelligent Systems for Technology Enhanced Learning*, vol. 804. Cham, Switzerland: Springer, 2019, pp. 45–52.

[56] O. Miglino, A. D. Ferdinando, R. D. Fuccio, A. Rega, and C. Ricci, "Bridging digital and physical educational games using RFID/NFC technologies," *J. e-Learn. Knowl. Soc.*, vol. 10, no. 3, pp. 89–106, Sep. 2014.

[57] M. De Angelis, A. Gaeta, F. Orcioli, and M. Parente, "Improving learning with augmented reality: A didactic re-mediation model from Inf@ nzia DigiTales 3.6," *J. e-Learn. Knowl. Soc.*, vol. 15, no. 3, pp. 287–300, 2019.

[58] S. Tartir, I. B. Arpinar, and A. P. Sheth, "Ontological evaluation and validation," in *Theory and Applications of Ontology: Computer Applications*, R. Poli, M. Healy, and A. Kameas, Eds. Dordrecht, The Netherlands: Springer, 2010, pp. 115–130.

MAIGA CHANG (Member, IEEE) is a Full Professor with the School of Computing Information and Systems, Athabasca University, Canada. His studies mainly focus on game-based learning, training and assessment, learning behavior pattern analysis and extraction, learning analytics and academic analytics, data mining and artificial intelligence, intelligent agent technology, and mobile learning and ubiquitous learning. He is currently the Chair of the IEEE Technical Committee of Learning Technology (IEEE TCLT), an Executive Committee Member of the Asia–Pacific Society for Computers in Education (APSCE), the Global

Chinese Society for Computing in Education (GCSCE), and the Chinese Society for Inquiry Learning (CSIL). He is a Secretary and Treasurer of the International Association of Smart Learning Environments (IASLE). He is also Chair of the Digital Game and Intelligent Toy Enhanced Learning Special Interest Group (SIG) under the IEEE TCLT. He also serves on academic international conference events include being the Program Chair of the 15th International Conference on Intelligent Tutoring Systems (ITS 2019), the IEEE TCLT Flagship Conference, the International Conference on Advanced Learning Technologies (IEEE ICALT, from 2017 to 2019), the International Conference on Smart Learning Environments (ICSLE 2015, 2018, and 2019), and the International Conference on Technology for Education (IEEE T4E 2019). He has given more than 97 talks and lectures at different conferences, universities, and events. He has participated in more than 290 international conferences and workshops as a Program Committee Member, and he has also coauthored more than 237 edited books, special issues, book chapters, journals, and international conference papers. He is the Editor-in-Chief of the *International Journal of Distance Education Technologies*, the Section Editor of *Education Sciences*, and an Associate Editor of *Transactions on Edutainment* (Springer), and Advisory Board Member of the *Journal of Computers and Applied Science Education*.

GIUSEPPE D'ANIELLO (Member, IEEE) graduated and received the Ph.D. degree in information engineering from the University of Salerno, Italy, in 2013 and 2018, respectively. He is currently a Postdoctoral Researcher with the Department of Information and Electrical Engineering and Applied Mathematics, University of Salerno. He has been a Research Scientist and the Project Manager of the CORISA (Research Consortium on Agent System). His current research interests include situation awareness, semantic web, computational intelligence, and granular computing. He has coauthored several scientific articles on the aforementioned topics, which have been published in international journals and conference proceedings. He is member of the IEEE SMC Technical Committee on Cognitive Situation Management. He is also a member of the IEEE Computational Intelligence Society, the IEEE Systems, Man, and Cybernetics Society, and the IEEE Computer Society.

MATTEO GAETA (Senior Member, IEEE) received the degree in information science from the University of Salerno. He is a Full Professor of information processing systems and the Scientific Coordinator of the KnowMIS Lab–Knowledge Management and Information Systems Lab, University of Salerno. He has authored over 200 scientific articles published in journals, proceedings, and books, and he has planned and designed more than ten information systems. His main research interests are complex information systems, decision support systems, knowledge management systems, and computational intelligence. He is also a member of the editorial board of international journals and the program committees of many conferences. He was promoted as the Director of Research and Development Centers, Technology Transfer and Spin-offs. He is the Scientific Coordinator and Manager of several International Research Projects. He is the Coordinator of the Working Group ex. art.14 D.M. 593/2000 for the financing of SMEs of the Italian Ministry of Instruction, University and Research (MIUR). He is the Editor-in-Chief of the *International Journal of Information Technology, Communications, and Convergence* (Inderscience), and an Associate Editor of the *Journal of Ambient Intelligence and Humanized Computing* (Springer).

FRANCESCO ORCIUOLI (Member, IEEE) received the master's degree *(cum laude)* in computer science from the Faculty of Mathematical, Physical, and Natural Sciences, University of Salerno, in 1999. Since 2015, he has been an Associate Professor with the University of Salerno. He is currently with the Department of Scienze Aziendali–Management and Innovation Systems. He has participated in numerous research projects also including ARISTOTLE and ALICE (EU FP7), TITAN, MODERN, eJRM and EVO-SMART (PON Research and Competitiveness, from 2007 to 2013), ELeGI (EU FP6), and m-Learning (EU FP5). He is author of almost 130 scientific papers on social semantic web, computational intelligence and distributed software architecture applied to situation awareness, technology enhanced learning, and knowledge management. His current research activities are focused on the definition of techniques, models and methodologies based on (temporal) formal concept analysis, rough sets, fuzzy cognitive maps, semantic web, and multiagent systems enabling decision support systems, and intelligent tutoring systems. He is member of the IEEE Computational Intelligence Society Membership and the IEEE Systems, Man, and Cybernetics Society Membership.

DEMETRIOS SAMPSON (Senior Member, IEEE) is currently a Professor of digital systems for learning and education with the Department of Digital Systems, University of Piraeus, Greece, since 2003, and a Professor of learning technologies with the School of Education, Curtin University, Australia, since 2015. He is the coauthor of 350 articles in scientific books, journals, and conferences, and the editor of 15 books, 37 special issues in academic journals, and 40 international conference proceedings, with more than 6100 citations and an H-index of 37 as listed in Google Scholar. He has received the Best Paper Award at International Conferences on Learning Technologies for ten times. He has been a Keynote/Invited Speaker in 90 International/National Conferences and/or Postgraduate Programs. He has been the project director, principal investigator, and/or a research consultant in 70 research and innovation projects with external funding at the range of 16 million €. He has supervised 155 honours and master's degree students for successful completion. He has been developing and delivering the Analytics for the Classroom Teacher MOOC offered by edX, which has attracted more than 16 000 participants from 160 countries around the world, since October 2016. He leads an international university–industry consortium (Learn2Analyse) that promotes professional development in educational data literacy for online education and training professionals and higher education students, co-funded by the European Commission (Erasmus+ Knowledge Alliance Program). He was the Editor-in-Chief of the *Journal of Educational Technology and Society*, from 2003 to 2018, and he also served or serves as member of the steering committee and/or advisory and/or editorial board of 25 international/national journals, as well as in various leadership roles in 80 international conferences and at the Program Committee of 600 international/national conferences. He was a recipient of the IEEE Computer Society Distinguished Service Award, in July 2012, and named a Golden Core Member of the IEEE Computer Society in recognition of his contribution to the field of learning technologies. He was also a recipient of the Golden Nikola Tesla Chain Award of the International Society for Engineering Pedagogy (IGIP) for the International outstanding achievements in the field of Engineering Pedagogy, in September 2018.

CARMINE SIMONELLI received the master's degree in computer engineering from the University of Salerno, Italy, in 2016. During the university studies, he worked on many research projects. His research interests are concerned with web crawling and artificial intelligence. He is currently a Software Engineer with NTT DATA Italia spa, Naples.