Increasing the Adaptivity of an Intelligent Tutoring System with Educational Data Mining: A System Overview

http://dx.doi.org/10.3991/ijet.v11i03.5103

I. Jugo, B. Kovačić and V. Slavuj
University of Rijeka, Rijeka, Croatia

***Abstract*—Intelligent Tutoring Systems (ITSs) are inherently adaptive e-learning systems usually created for teaching well-defined domains (e.g., mathematics). Their objective is to guide the student towards a predefined goal such as completing a lesson, task, or mastering a skill. Defining goals and guiding students is more complex in ill-defined domains where the expert defines the model of the knowledge domain or the students have freedom to follow their own path through it. In this paper we present an overview of our system's architecture that integrates the ITS with data mining tools and performs a number of educational data mining processes to increase the adaptivity and, consequently, the efficiency of the ITS.**

***Index Terms*—e-learning, intelligent tutoring systems, educational data mining, adaptive e-learning.**

I. INTRODUCTION

Educational Data Mining (EDM) is a growing field of research concerned with applying the existing artificial intelligence and machine learning methods, as well as developing new ones, for the purpose of analyzing and learning from data originating in educational environments. One of the main objectives of this field is to develop tools that are easy to use for teachers and other non-experts in data mining. Additionally, these tools need to be integrated into various e-learning systems to provide insights to teachers and improve the students' learning experience. This paper presents an overview of the architecture of a web-based ITS (WITS), developed at our institution, which implements those objectives. Our goal for this particular system was to increase its adaptivity and, consequently, overall efficiency. We developed an architecture model that consists of: a) an integration layer, b) expert modules for EDM analyses, and c) an updated tutoring model. By improving the adaptivity of the tutoring model we aim to improve the overall efficiency of the system.

II. RELATED WORK

The paper at hand addresses three main issues: (1) the development of an integration layer between DM tools and our WITS, which serves as the basis for automatization of EDM processes; (2) the implementation of a two-step clustering evaluation process; and (3) the development of algorithms for automatic evaluation of frequent paths through the knowledge domain discovered by SPM algorithms. A similar approach that required teachers to analyze and select frequent rules has been applied for the purpose of recommending web pages in [1]. In [2], authors present an integration module for a Moodle block that enables the users to perform three DM analyses and export the raw output to a file. In our system, the results are automatically evaluated and used by the tutoring module to dynamically adapt the learning structure to the current students' knowledge level. Examples of more recent approaches that employ sequential pattern mining algorithms to improve the results of desktop ITSs can be found in [3] and [4].

Student clustering is another important research topic in EDM. An overview of the clustering analysis critical steps is presented in [5]. In cluster analysis, a fundamental problem is to determine the best number of clusters. A variety of methods have been proposed to estimate the number of clusters. Ref. [6] gives an overview of those methods. Their performance has been analyzed in [7] and [8]. In our system we implemented the silhouette statistic method. Finally, the obtained cluster structure can be evaluated through descriptive statistics or a number of more complex methods [9], while the interpretation depends on the research area and the nature of data. Our system relies on descriptive statistics to create an algorithm that automatically grades the clusters in relation to cluster members activity levels as well as learning efficiency.

III. RESEARCH ENVIRONMENT

Our WITS, first described in [10], provides a platform for learning in ill-defined domains [11]. Such domains consist of a number of knowledge units (KUs) that do not have a strictly defined order in which they have to be taught/learned. Instead, the system relies on a domain expert to define the structure of the domain.

The tutoring process begins when the student selects a KU, and the system creates a "learning structure". The learning structure is a database construct that contains the current KU and all the KUs one level below it. Fig.1 represents some possible tutoring model situations for a sample domain model depicted on the left hand side. These also represent paths the user creates (and the system records) while advancing through the knowledge domain. On the right hand side of Fig. 1 are four out of numerous possible paths the user can create while using our system.

The first situation is the simplest – the student selects and is presented with learning materials for unit A (LA), and is then asked a question about the same unit (QA). Notwithstanding the correctness of the answer, the system asks the student one initial question for the units one level below the current KU in the hierarchy (IQB and IQC). If the student answers both questions correctly, the learning structure is completed and the student is redirected to the starting page. However, if the student had reached the previously defined threshold on units B and C earlier, then the first situation would end after the LA□QA sequence. This is because the tutoring model does not add already completed KUs to the learning structure.

The fourth situation is the most complex one and is created if the student answers all initial questions (B,C,D,E) incorrectly. From these examples it is clear that the system adapts the tutoring process to the students' knowledge level only as it omits completed units. Moreover, it will create the exact same learning structure for all students. The new system architecture aims to make this part more adaptive based on the analysis of students interactions (learning activity and efficiency).

IV. INTEGRATION WITH DM TOOLS

In order to make the EDM processes automatic we needed an integration layer that enables continuous communication with DM tools. The system architecture is presented in Fig. 2. Currently, the integration layer can communicate with Weka [12] and SPMF [13]. In this way, re-implementing any specific algorithm into our application was avoided. Additionally, it ensured that the data from our system can be analyzed by a DM expert on another machine, running the same DM tools, with absolute confidence that the results will be the same (where it is possible, depending on the algorithm). An important advantage of this architecture is that the administrator can use any clustering or SPM algorithm provided by either tool. The default tool-algorithm pairs are: Weka-KMeans for clustering and SPMF-PrefixSPAN for SPM. The integration layer is separated into modules for (1) *data preparation* which accesses raw learning data, and creates engineered features, (2) *data input/output* which writes prepared data to a file in the correct format and reads the resulting files generated by DM tools, (3) *communication* which prepares and performs API calls to the required DM tool; and (4) *data processing* which processes the results either by forwarding them to the ITS interface (data visualization) or updating the ITS database.

Further details on the integration architecture and the clustering analysis are available in [14] and [15].

V. AUTOMATIC CLUSTERING EVALUATION

The integration with DM tools described in the previous section enabled us to expand the student model with information about the students' activity. Such an enriched model is one of the prerequisites for increasing the adaptivity of our system.

The student model in our system now consists of two parts. The first is a knowledge matrix which keeps track of the students' progress. The second is the activity model, which consists of four engineered features created from records about the students' interactions with the system. Three of those features represent the level of students' engagement with the system; namely, (1) the number of times the student accessed the learning materials, (2) time spent learning, and (3) the number of times the student tested his/her knowledge, while the fourth feature represents the students' efficiency in answering questions about the presented content. These features are created using formulas that take into consideration the students' current status in the domain (percentage of the content covered)

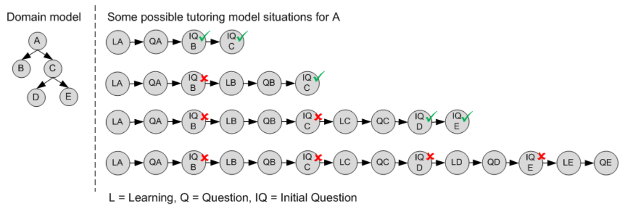
Domain model
A
B
C
D
E
Some possible tutoring model situations for A
LA
QA
IQ B
IQ C
LB
QB
LC
QC
IQ D
IQ E
LD
QD
LE
QE
L = Learning, Q = Question, IQ = Initial Question

Figure 1. Possible tutoring situations for a sample domain

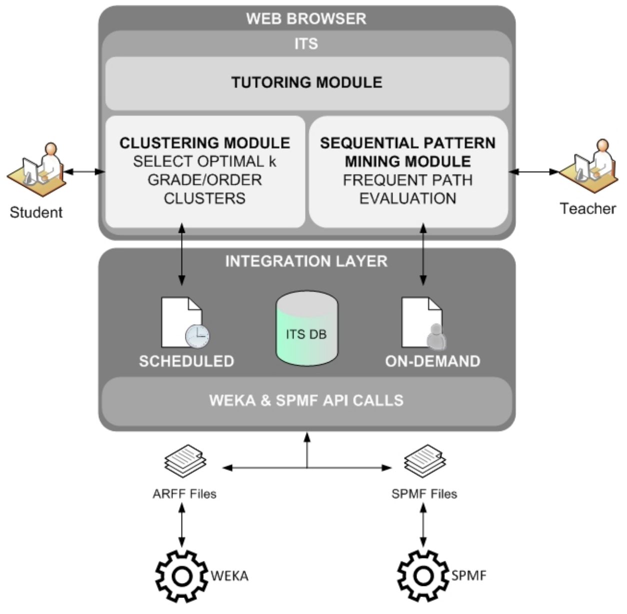

Figure 2. The system architecture

and are standardized. Clustering analysis if performed on this dataset in hourly intervals and the students' activity model is updated with the ID of the cluster the student was assigned to. This is a well-known approach to using DM in e-learning systems but it is used mainly in Java-based desktop ITSs and requires human analysis of the clustering results: selecting the model with the optimal number of clusters.

Our contribution to this approach is twofold. Firstly, we have developed a module that automatically evaluates the acquired models and, using the silhouette statistic [7], selects the best number of clusters. Secondly, we developed an evaluation module based on the pedagogical assumption that different clusters represent more and less active/effective groups of students, and the system should tend to guiding less efficient students towards the actions and paths created by more efficient groups. Instead of just detecting clusters, our aim was to grade them automatically. In this way, for example, the students from a cluster with a lower grade can be "tutored towards" the activity levels of the cluster with a higher grade. Fig. 3 shows how our approach automatizes the process of selecting the model with the best number of clusters using the integrated silhouette statistics module.

It also shows the next step - the automatic grading of clusters, so that in the following steps (see Section VII) we can lead the students from (the lowest graded) cluster D towards paths and activity levels of students from cluster C, students from cluster C towards those belonging to cluster B, etc., thus increasing the adaptivity and overall efficiency of the system. More details on the integration architecture and the clustering analysis can be found in [15].

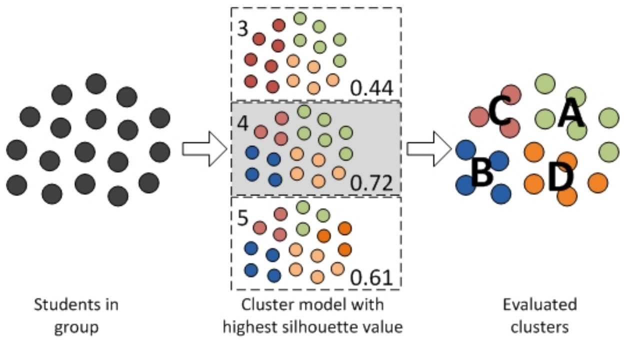

Figure 3. Clustering evaluation process

The following Section describes the remaining elements of our systems architecture.

VI. FINDING AND EVALUATING FREQUENT PATHS

The approach to finding and evaluating frequent paths has been used by some of the systems previously mentioned in Section II. It requires the intervention of domain experts to manually identify and select useful or productive paths to be displayed or recommended to users. This slows down the process significantly. Our contribution to this approach is the algorithm for automatic evaluation of frequent paths found by the SPM algorithm.

Paths that the students take through a knowledge domain can be frequent for a number of reasons. From the standpoint of the teacher, those reasons can be either positive or negative. The following make the argument for the positive reasons: discovering common paths through the domain structure, discovering optimal paths through the domain structure, following the domain structure put forth by the expert or discovering new frequent paths that reveal new paths through the structure. Conversely, the negative reasons may be the following: error in structure, questions with no correct answer, hard questions or part of domain structure, errors in system/algorithm/database, etc. Both types of frequent paths may prove useful once they have been evaluated.

Paths that reflect errors in structure or questions (e.g. "circular" paths such as "*learn → incorrect answer → learn*") should be communicated to the teacher or the administrator, so they can correct them as soon as possible. Paths that reveal new structures and generate the highest gains in the knowledge matrix should be used to lead the students in order to increase both the system effectiveness (knowledge acquisition time and test results) and user satisfaction. This results with a list of frequent paths (FPs) which is returned to our (FP) evaluation algorithm. The algorithm analyses each FP to provide a final score or ranking that accompanies every FP before it is stored in the ITS database. When analyzing the FP, we take into consideration a number of factors: a) all updates to the knowledge matrix that resulted from the FP, b) the length of the path (PL), and c) the structure of the path (number of learnings, repetitions, questions and initial questions displayed to the student).

Table I illustrates the point using the domain structure and paths displayed in Fig. 1. By way of example, let us assume that each question was answered correctly, for which the knowledge matrix value for the corresponding KU is increased by 2 (this value is determined by the teacher at the time the question is created), while some initial questions where not.

TABLE I.
FREQUENT PATH EVALUATION ATTRIBUTES

A	B	C	D	E	∑	PL	Ls	Qs	IQs
+2	+0.2	+0.2			2.4	4	1	1	2
+2	-0.1 +2	+0.2			4.1	6	2	2	2
+2	-0.1 +2	-0.1 +2	+0.2	+0.2	6.2	10	3	3	4
+2	-0.1 +2	-0.1 +2	-0.1 +2	-0.1 +2	9.6	14	5	5	4

Initial questions have a small impact on the knowledge matrix as their purpose is to act as indicators to the tutoring model on how to guide the student. Correct answers to IQs usually add just 0.2 while incorrect answers subtract 0.1 from the current value in the knowledge matrix. By analyzing the paths in Fig. 1 we can easily identify the shortest and the longest ones. From the Table I. ∑ column we can quickly identify the path that produced the highest gain in the KM. Even though the last path produced the highest gain, when we consider its length and structure, it is clear that it is not optimal.

FP evaluation takes each FP and assigns to it a score and the cluster(s) that created it. Path scoring algorithm evaluates the knowledge gain of all the students that followed a particular path, the duration and length of the path, the time of appearance (start, middle or end of the total time the students had access to the knowledge domain) and a number of other factors. Ranking the FPs by efficiency and connecting them to the cluster(s) in which they occur is the final step and the last one that is done in scheduled runs. All the aforementioned steps are prerequisites which enable the last phase: the modification of the tutoring model of the WITS to increase the systems adaptivity and efficiency.

VII. DYNAMIC LEARNING STRUCTURE CREATION

The previous two Sections presented the modules that use the results of well-known DM algorithms to evaluate both the students' activity and the paths they take through our domain. The modules update and store their results every hour to make use of new data from the constantly growing dataset of student interactions with the system. With these two modules in place we can finally modify our tutoring module to dynamically adapt the learning structures for each student. The presumption is that instead of leading all students through the same learning structures, we can optimize their learning experience, i.e., shorten the time needed to complete the domain and improve the students' test results or the overall course grade.

This part of the system works in real time as opposed to the previous parts of our architecture. When the student selects a KU, the tutoring model creates a learning structure based on the domain model. To illustrate this process, we can again use the simple domain model displayed in Fig. 1. The initial version of our tutoring model for the selected unit A, always creates the same learning structure (B, C). The improved tutoring algorithm can now dynamically adapt the learning structure, i.e., the structure can be C, B or it can be B, E, C, etc. The tutoring algorithm first checks if the student already has a cluster ID assigned to him/her. If not, the student is presented by the default learning structure. Conversely, if the student has a cluster

ID, the algorithm will check the grade of the cluster and query the PFP database for paths that start with or contain the selected unit for his cluster and the cluster one grade higher. If there are such paths, the algorithm then checks the student's knowledge matrix and filters the paths that contain KUs already completed by the student.

Finally, an appropriate PFP is selected, the learning structure is created and the student is guided through it. Each such learning structure modification is logged and the students' results are recorded in order to monitor and update the effectiveness of the created learning structure.

In this way, we have created a WITS that is fully integrated with DM tools and performs a number of automated EDM processes. It is important to emphasize that the system starts to adapt to the behavior of the group almost instantly because the adaptivity feature is initiated after the first day of granting students access to the knowledge domain, during which the group usually creates around 40% of the final dataset (records of their interactions with the system). As expected, the results of the adaptive tutoring algorithm get better as the dataset grows and more FPs are created by students.

VIII. CONCLUSION

In this paper we have given an overview of our system architecture that uses EDM to increase the adaptivity of our WITS, developed for teaching in ill-defined domains. The domain model in such domains is created by the expert based on his/her assumptions regarding the best way to learn the domain. The underlying assumption is that each student is unique as to precognition, working habits, etc., so it is safe to anticipate that the students will create their own paths through the domain model.

The presented system architecture uses the records of learning interactions created by students to discover those paths, but employs new algorithms to discriminate between productive and nonproductive paths. The tutoring algorithm is improved in the pedagogical aspect through automatic cluster grading which enables it to guide the students towards more productive paths through the domain.

We believe that this approach has great potential as it a) removes the time delay that usually exists in EDM process, and b) overcomes the need for DM, programming or database experts. Thus, the system works autonomously using the data created by the same students the system is trying to adapt to. In this way, we are not creating a model based on one group of students and applying it on a completely new one the next semester or academic year.

Our future work will consist of expanding our activity model with more engineered features, experimenting with more clustering and SPM algorithms, as well as improving the results of our clustering and frequent path analysis modules.

REFERENCES

[1] C. Romero, S. Ventura, J.A. Delgado and P. De Bra, "Personalized links recommendation based on data mining in adaptive educational hypermedia systems". *Creating New Learning Experiences on a Global Scale*, Springer Berlin Heidelberg , 2007, pp. 292-306.

[2] C. Romero, C. Castro and S. Ventura, "A Moodle Block for Selecting, Visualizing and Mining Students' Usage Data." 7th Int. Conf. on Educational Data Mining, London, 2013.

[3] P. Fournier-Viger, R. Nkambou, A. Mayers, E. Mephu-Nguifo and U. Faghihi, "Multi-paradigm generation of tutoring feedback in robotic arm manipulation training". *Intelligent Tutoring Systems*, Springer Berlin Heidelberg, 2012, pp. 233-242. http://dx.doi.org/10.1007/978-3-642-30950-2_29

[4] J. S. Kinnebrew, K.M. Loretz, and G. Biswas, "A contextualized, differential sequence mining method to derive students' learning behavior patterns" in *Journal of Educational Data Mining*, vol. 5(1), pp. 190-219, 2013.

[5] G. W. Milligan, "A validation study of a variable weighting algorithm for cluster analysis" in *Journal of Classification*, vol. 6, pp. 53-71, 1989. http://dx.doi.org/10.1007/BF01908588

[6] A. D. Gordon, "Classification". Chapman & Hall/CRC, Boca Raton, FL, 2 edition, 1999.

[7] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis" in *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53-65, 1987. http://dx.doi.org/10.1016/0377-0427(87)90125-7

[8] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of data clusters via the gap statistic" in *Journal of the Royal Statistical Society B*, vol. 63, pp. 411-423, 2001.

[9] L. Kaufman and P.J. Rousseeuw, "Finding Groups in Data. An Introduction to Cluster Analysis". Wiley-Interscience, New York, 1990.

[10] B. Kovacic and Z. Skocir, "Development of distance learning system based on dialogue," in EUROCON. The IEEE Region 8 , vol.1, pp.224-228, 2003. http://dx.doi.org/10.1109/eurcon.2003.1248015

[11] C. F. Lynch, K.D. Ashley, V. Aleven, and N. Pinkward, "Defining Ill-Defined Domains; A literature survey" in Proc. Intelligent Tutoring Systems Ill-Defined Domains Workshop, Taiwan, 2006, pp. 1-10.

[12] M. Hall, et al., "The WEKA Data Mining Software: An Update" in SIGKDD Explorations, 2009, vol. 11, issue 1. http://dx.doi.org/10.1145/1656274.1656278

[13] P. Fournier-Viger, et al., "SPMF: a Java Open-Source Pattern Mining Library" in *J. Machine Learning Research*, 2014, vol. 15, pp. 3389-3393.

[14] I. Jugo, B. Kovačić and V. Slavuj, "A proposal for a web based educational data mining and visualization system" in Proc. Information Technologies and Information Society, 2013.

[15] I. Jugo, B. Kovačić and E. Tijan, "Cluster analysis of student activity in a web-based intelligent tutoring system" in Pomorstvo: journal of maritime studies, vol. 29, pp. 80-88, 2015.

AUTHORS

I. Jugo is a PhD student and a research assistant at the Department of Informatics, University of Rijeka, Croatia (e-mail: ijugo@inf.uniri.hr).

B. Kovačić, is an assistant professor with the Department of Informatics, University of Rijeka, Croatia (e-mail: bkovacic@inf.uniri.hr).

V. Slavuj is a PhD student and a research assistant at the Department of Informatics, University of Rijeka, Croatia (e-mail: vslavuj@inf.uniri.hr).

This work was supported in part by project "Enhancing the efficiency of an e-learning system based on data mining", code: 13.13.1.2.02., funded by the University of Rijeka, Croatia.

Submitted, 29 September 2015. Published as resubmitted by the authors on 07 November 2015.