IST-Africa 2016 Conference Proceedings
Paul Cunningham and Miriam Cunningham (Eds)
IIMC International Information Management Corporation, 2016
ISBN: 978-1-905824-55-7

Designing Adaptive Learning Support through Machine Learning Techniques

Robert O. OBOKO[1], ELizaphan M. MAINA[2], Peter W. WAIGANJO[3], Elijah I. OMWENGA[4], Ruth D. WARIO[5]
[1, 3, 4] University of Nairobi, P. O. Box 30197, 00100, Nairobi, Kenya
+254 020 4447870, Fax: +254 020 4447870, [1] roboko@uonbi.ac.ke,
[3] waiganjo@uonbi.ac.ke, [4] eomwenga@uonbi.ac.ke
[2]Kenyatta University, P.O. Box 43844, 00100, Nairobi, Kenya
+254 20 8710901-19, Fax: +254 020 8711575, maina.elizaphan@ku.ac.ke:
[5]University of Free State, Private Bag X13, Kestell 9866, Republic of South Africa
+27768377990 Fax: 058 7185247, wariord@ufs.ac.za

Abstract: The use of web 2.0 technologies in web based learning systems has made learning more learner-centered. In a learner centered environment, there is need to provide appropriate support to learners based on individual learner characteristics in order to maximize learning. This requires a Web-based learning system to have an adaptive interface to suit individual learner characteristics in order to accommodate diversity of learner needs and abilities and to maintain an appropriate context for interaction and for achieving personalized learning. The purpose of this paper is to discuss how machine learning techniques can provide adaptive learning support in a Web-based learning system. In this research, two machine learning algorithms namely: Heterogeneous Value Difference Metric (HVDM) and Naive Bayes Classifier (NBC) were used. HVDM was used to determine those learners who were similar to the current learner while NBC was used to estimate the likelihood that the learner would need to use additional materials for the current concept. To demonstrate the concept we used a course in object oriented programming (OOP).

Keywords: Web-based learning system; Machine learning; Heterogeneous Value Difference Metric; Naive Bayes Classifier; Adaptive interface

1. Introduction

Adaptive learning techniques can be used to support acquisition of domain knowledge. They change the system's interface to suit the learner's characteristics. The techniques work by reducing the number of information elements to be processed at one time to a manageable level according to the learner's expertise. They also provide favorable presentation of the learning material and give hints to make it easier for the learner to identify an appropriate navigation path to follow through the web-based course content. The techniques can reduce the cognitive load suffered if used appropriately. Examples of these techniques are adaptive navigation support, adaptive presentation support and adaptive curriculum sequencing [1].

For instance, with adaptive navigation support through link hiding, some learning material links can appear or disappear automatically to reduce the student's information space to relevant information elements only. This can be based on patterns of how the student had previously used the materials in the course [2].

Adaptation should take place right from the onset of learning, but the learner models need to be initialized for this to take place. Rather than assume that the learners have no domain knowledge or they are the same, learner model initialization techniques with

machine learning can be used to estimate the learner's entry level of domain knowledge based on the learner's profile and the entry level knowledge of similar learners who started learning in the system earlier than the current learner. Initialization of learner models has been considered important because it is not reasonable to assume that all learners have the same level of domain knowledge at the beginning or that they have no knowledge at all about the domain at the beginning [3]. If the system uses these assumptions about the level of performance of the learners, then there is a high likelihood of the system giving unreasonable advice or suggestions to the learners, which may end up causing cognitive load and, therefore, disrupting instead of supporting the learner's learning [3].

Machine learning algorithms can be used to enable a system to "learn" from what it "experiences". The experience is represented in the form of some data sets such as logs of user activities in the system. The system then uses this learned knowledge to perform better in solving similar problems, as it acquires more experience i.e. as it is exposed to more data sets. Some measure of performance is used to determine this improvement [4]. Adaptive user interface systems rely on models of learners, models of the domain and inference rules in order to make adjustments to their interfaces to suit individual learners. Machine Learning algorithms can be used for learner modeling since an algorithm can infer the learner's category from the provided data or propose a new cluster for some of the learners, which can be given a name. Such a category is added to the learner model for instance, the current level of knowledge or preference to use additional learning material in an online learning system. For this research, machine learning algorithms were used to learn some of the attributes of the learners. Specifically, they were used to learn the initial knowledge level of the learner and whether the learner would choose to use additional learning materials while studying a concept or not. The course used in this study i.e. Introduction to Object Oriented Programming was designed into the global information, goals and concepts, with the domain content of each concept being constituted by various types of learning objects which were combined to form knowledge modules when required by learners.

2. Objectives

The aim of this paper is to discuss how machine learning techniques can provide adaptive learning support in a Web-based learning system. The specific objectives are:

1. To describe the design requirements for an adaptive user interface in a web-based learning systems
2. To demonstrate how machine learning techniques can be used to provide adaptive learning support in a web-based learning systems

3. Methodology

To design an adaptive user interface, six adaptive user interface components were considered: learner model, domain model, data collection techniques, multiple representations to cater for various types of learners, inference mechanisms and adaptation rules.

3.1 The Learner Model Component

The learner model had domain dependent attributes, which mostly related to prior programming experience, non-programming computer experience, level of comfort and the learner's attitude to programming [5]–[7]. The other domain dependent information included concept level knowledge and tendency to use additional learning materials. These attributes were captured in a learner model by capturing appropriate values for the concepts the learners were studying. Hence it was an overlay over the domain model. These latter

two attributes were used during adaptation while the other domain dependent attributes were used to infer these two from information of the current learner and other learners already enrolled and learning in the web-based system. These attributes are shown in Figure 1.

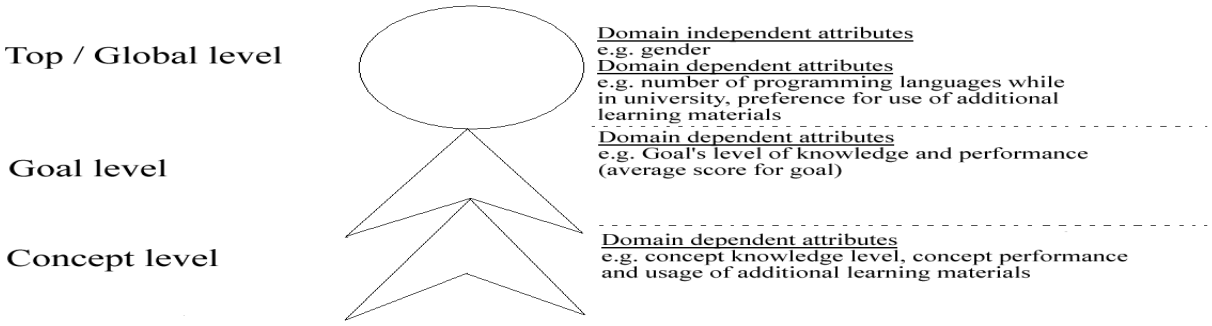

Figure 1 : Learner model showing domain independent and domain dependent attributes

The domain independent attribute included was gender [5]. Figure 1 shows the learner model used for adaptation, with its' domain independent and domain dependent attributes. The attributes are shown at the global level, goal level and concept level.

3.2 The Domain Model Components

The domain area was programming where students were required to learn how to program using the object oriented concepts such as class design and inheritance. The domain model i.e. the top-down structure of the course had 4 goals with a total of 14 concepts which the learner could choose to study. The concepts were divided into 2 parts, Class Design and Inheritance. Class design had 8 concepts whereas Inheritance had 6 concepts. Each concept had a number of knowledge modules for delivering course content.

3.3 Data Collection Techniques

Data was collected obtrusively and non-obtrusively, as shown in Table 1. The students who participated in this study were 92 first year students of BSc. Information Technology from two campuses of one of the Kenyan public universities in Nairobi. The students were studying the course called Introduction to Object Oriented Programming. They had already taken a pre-requisite course Programming Methodology but they had never formally been taught or used the problem solving approach.

Obtrusive collection of data is direct whereas non-obtrusive collection is implicit. In the table, data collected obtrusively is shown in the first column whereas data collected non-obtrusively is shown in the second column.

Table 1: Showing how different elements of data were collected

Obtrusively collected data	Non-obtrusively collected data
• Personal information (e.g. gender)	• The marks in pre-tests and concept tests
• Survey information on the learners' experience	• The sequence of studying the concepts
• The sub-tasks defined by the learner during problem solving	• The usage (used or did not use) of additional study materials for a concept
• A percentage figure expressing the extent of task completion – task progress	• How long it took the learner to study a concept

• The date of commencement of a sub-task	• The concepts whose study was aborted (opened but no answers were submitted for the test)
• The time elapsed since commencement of a sub-task	
• The written tests at the end of each of the two sections of the course	
• The answers to the self-evaluation pre-test and concept evaluation	

3.4 Multiple Representations

First, the level of difficulty of the concepts was considered during the design of multi-representations of the course. Three levels of knowledge were considered i.e. novice, intermediate and advanced. The presentation of the course content was organized into 3 abstractions to suit learners at the 3 different levels of knowledge. Secondly, multiple ways of presenting concept names were also designed, using different colors, according to the level of difficulty of a concept and the overall level of knowledge of the learner. This was done according to the traffic light metaphor, based on the learner's readiness to study the concepts. Some concept names were shown in red (meaning the learner was not ready to learn them), others in orange (meaning the learner was ready to learn them) and others in green (meaning the learner was ready to learn them and was expected to do well). Thirdly, for additional materials, there were 2 representations designed: to show or not to show the additional materials at the end of the lesson page.

3.5 Inference

Machine learning algorithms were used to predict the initial level of knowledge of the learner and the likelihood of the learner using additional learning materials for each concept. **Heterogeneous Value Difference Metric** (HVDM) was used to determine those learners who were similar to the current learner. The knowledge levels of these learners were used to initialize the current learner's knowledge level, on commencing learning using adaptive support [2]. **Naive Bayes Classifier** (NBC) was used to estimate the likelihood that the learner would need to use additional materials for the current concept. This was based on the on additional material links' click history of the current learner and the other learners who had studied the course up to the current concept [8].

3.6 Adaptation of the User Interface

Adaptation rules were used to determine the appropriate adjustments to be made to the system's interface. To provide an adaptive user interface the design ensured:

(a) The presentation of the appropriate abstraction of course content, in terms of appropriate level of difficulty. This was determined by the level of knowledge of the learner. The three levels of knowledge namely novice, intermediate and advanced were matched to three types of course content presentation

(b) The annotation of the names of the concepts to show the learner's readiness to learn them was in terms of three colors which were matched to the three levels of knowledge

(c) The decision to present or not to present additional material links was made based on whether the likelihood of usage of additional materials in the learner model was inferred as Yes or No.

4. Technology Description

To complement instructional design, adaptive features can also be added to a system, especially online learning systems which are highly interactive and are used by diverse users. This is partly to reduce the expertise reversal effect which arises due to subjecting expert learners to instructional approaches suited for novice learners [9]. Adaptive features in general aid the system in providing learning support that is appropriate to an individual learner's attributes, thus reducing undesirable cognitive load effects. For web-based learning, adaptive hypermedia techniques such as adaptive presentation and adaptive navigation design are handy [1]). As part of reducing undesirable cognitive load, learners also need to have control over their learning so that they can choose to ignore the system's suggestions if they are not appropriate. This way, undesirable cognitive load effects such as redundancy and split attention effects can be avoided.

In order to encourage constructivist learning, especially in training programs involving authentic, complex, ill-structured tasks, learners need to be motivated to utilize any cognitive resources which become available due to reduction in extraneous and intrinsic cognitive load. Encouraging learners to apply metacognitive monitoring of their learning processes by increasing metacognitive awareness is important. The learners need to be aided to engage in planning, management of task related information for the problem solving process, monitoring, debugging and evaluation, reflection, self-explanation and external representation of their internal knowledge [10]. This can be done through prompting learners while they are engaged in complex problem solving, providing them with space where they can express themselves and allowing them to have control over their learning and problem solving processes, among others.

5. Designing the Adaptive User Interface

5.1 Initialization of Learner Model Concept Scores Using HVDM

The estimation of the initial level of performance of a learner was based on the current levels of performance of the other learners already learning OOP using the system was done by applying HVDM machine learning algorithm. The assumption was that if the learners were similar in terms of profile and background attributes, then the level of performance of those already in the system could be used to estimate the initial level of performance of the current learner, who was joining the course [11], for purposes of initializing the learner model to make adaptive learning possible right from the beginning of learning using the web-based system. The initialized scores were used to estimate the learner's level of readiness for the different concepts. In turn, the level of readiness was used to suggest concepts which the learner was expected to do well in and a suitable level of difficulty of concept's content for the learner. The learner model is later updated with the learner's actual performance. These two features of adaptation at concept level and adaptation at the level of content for a concept to facilitating learners in an online learning system are called adaptive navigation support with link annotation and adaptive text presentation support respectively, and are shown in Figure 2.

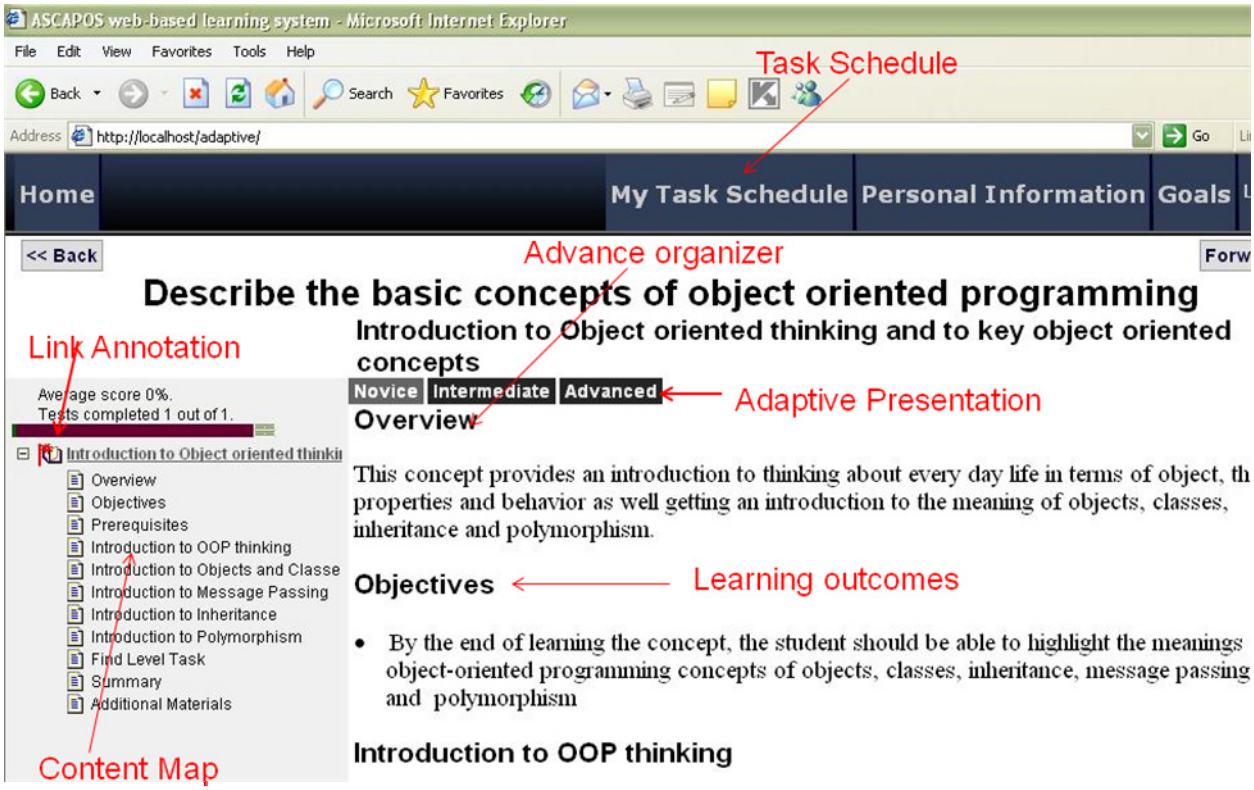

Figure 2: Adaptive link annotation and adaptive text presentation using initialized scores

5.2 Adaptive Presentation of Additional Learning Materials Using NBC

The NBC was used to predict the class or category into which a learner would fall. For adaptive presentation of additional learning materials, the two classes were: use and no use of additional learning materials. NBC has to acquire knowledge or learn from data sets to enable it to do this classification. Learning for the NBC machine learning algorithm involves establishing prior probabilities of the different classes and estimating the conditional probabilities of the various feature values, which are attributes of the learner in the learning process that determine whether the learner will want to use or not use additional learning materials, given each of the classes. The NBC classifier assigns the most likely class to a given training example, as described by its feature values. Another example of such classification is the problem of classifying learners based on knowledge level, in which a learner might be labeled as being at the beginning, intermediate or advanced level [11]. The inferred values i.e. classes are then used to update the learner model to make the new class the current class of the learner, is then used to guide the adaptation of the system's user interface. Adaptation in this was to hide or unhide additional learning materials for each concept for a learner.

The links to additional materials were well labeled with descriptive text to indicate what the nodes underlying the links were about. Therefore, clicks on the links were assumed to indicate intention to read the materials [12]. The click history of the current student and the other students was used to calculate the probability of the student wanting to access the additional materials [8].

The data used by this algorithm was collected non-obtrusively. As the learners clicked on additional learning material links, a log was made. If the learner submitted a concept evaluation test without having clicked on the additional learning materials, then access = N was added to the log for the student for this concept, indicating no use. Otherwise access = Y was added to the log to indicate use of additional learning materials.

6. Testing the Adaptive User Interface

To test the adaptive user interface, 92 first year students of BSc. Information Technology from two campuses of one of the Kenyan public universities in Nairobi were used. The students were studying the course called Introduction to Object Oriented Programming. The course had two tests: pre-test taken before the learners embarked on learning and a post-test, taken at the end of learning the content of each concept. The scores of the learner from the pre-test were used to estimate the learner's initial knowledge level. The pre-test questions were designed in such a way that there were two questions at the novice level of difficulty, two questions at the intermediate level of difficulty and two more questions at the advanced level of difficulty. If both questions had been answered incorrectly, then the learner was assigned the level of knowledge preceding the current knowledge level of the questions. If one question was answered incorrectly after two attempts, the learner was assigned the current level of knowledge of the questions. If it is the novice level and both answers were incorrect, then the learner was still assigned the novice level. If the level was the advanced and the learner answered both questions correctly, then he was still assigned the advanced level.

The information that was passed to the NBC was based on the clicks the learner made while accessing additional learning materials. If the learner made no clicks, i.e. did not access the material by the time of answering the concept evaluation questions i.e. the post-test, then information to indicate no access was sent to the classifier (NO entry). If the learner made clicks, the corresponding information to indicate access of the additional materials (YES entry) was sent to the classifier. This information on clicks was stored in the database (either Y or N) against corresponding concepts and the current learner.

Figure 3 shows course content of the concept "Introduction to Class Features" under the goal called "Demonstrate the use of classes and objects in object oriented programming". It shows that for a learner at intermediate level of knowledge, the concept has been annotated with orange colour. This colour indicates that the learner is ready to study the concept. The tab for intermediate level of difficulty of content is also proposed. Figure 3 also shows that links for additional materials have been displayed, indicating that the learner and other similar learners tended to use the materials in previously studied concepts.

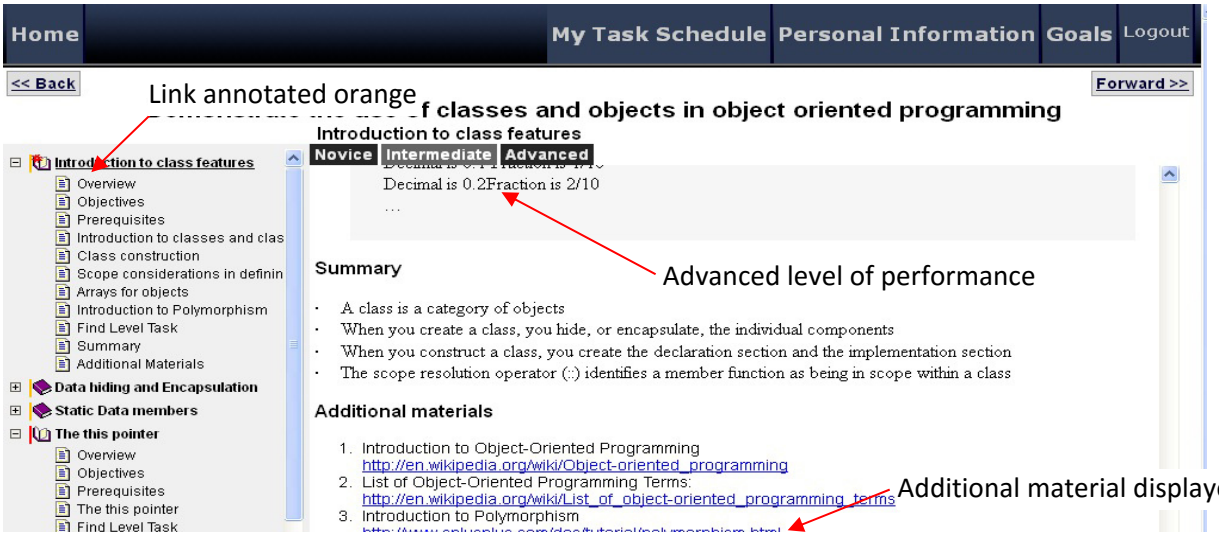

Figure 3: Link annotation, adaptive presentation and additional material presentation for intermediate knowledge level learner

Figure 4 shows course content of the concept "Introduction to Class Features" under the goal called "Demonstrate the use of classes and objects in object oriented programming". It

shows that for a learner at advanced level of knowledge, the concept has been annotated with green color. The colour indicates that the learner is ready and is expected do well in this concept. The tab for advanced level of performance of content is also proposed. Figure 4 also shows that links for additional materials have not been displayed, indicating that the learner and other similar learners tended not to use the materials in previously studied concepts.

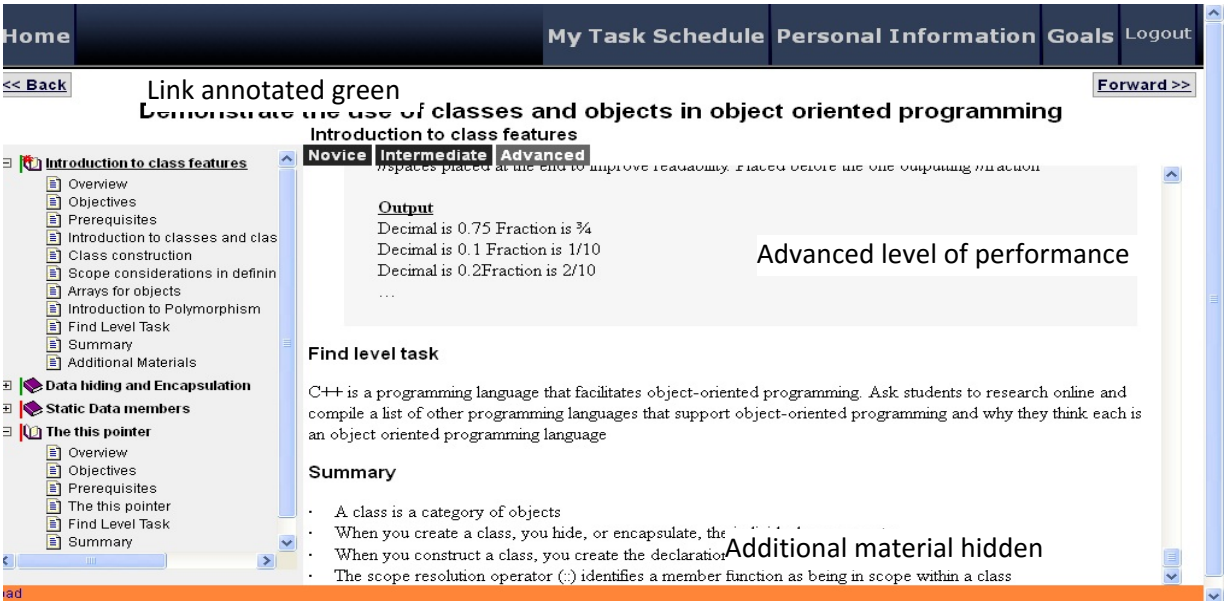

Figure 4: Link annotation, adaptive presentation and additional material presentation for advanced knowledge level learner

7. Benefits

An approach on how to develop an adaptive user interface for the web-based learning system has been described in this paper. This is an important contribution to the engineering of e-learning systems. It not only emphasizes inclusion of supportive features for both cognitive and meta-cognitive learning but also provides a template for factoring into system design critical factors such as user understanding and profiling, automatic profile updating, provision of personalized user interfaces and domain or task modeling as opposed to mere delivery of content. This will impact the field of development of web-based learning systems whose users vary in many ways and therefore, making it necessary for the systems to respond to unique learner needs. In this era of cognitive computing the study has demonstrated the use of two machine learning algorithms (HVDM and NBC) in designing an adaptive user interface.

8. Conclusion

This paper has discussed a model to guide the design of adaptive user interfaces for learning. The overlay model is updated with values inferred from previous learner activities, by using 2 machine learning algorithms. These algorithms were Heterogeneous Value Difference Metric (HVDM) and Naïve Bayes Classifier (NBC). The design shows that it is possible to use machine learning algorithms to implement adaptive educational hypermedia technologies such as adaptive navigation support with link annotation, adaptive navigation support with link hiding and adaptive presentation support. This design can also be integrated with other designs where machine learning algorithms have been applied to other aspects of e-learning such as collaborative learning to provide computer supported collaborative learning [13]. To establish the effectiveness of this design in improving web-

based learning we propose an experimental design to be carried out in a realistic web-based learning environment.

References

[1] P. Brusilovsky and others, "Adaptive and intelligent technologies for web-based eduction," *KI*, vol. 13, no. 4, pp. 19–25, 1999.

[2] O. E. L. A. Oboko R Wagacha P, "Comparison of Different Machine Learning Algorithms for the Initialization of Student Knowledge Level in a Learner Model-Based Adaptive E-Learning System," *Int. J. Comput. ICT Res.*, vol. 3, no. 1.

[3] P. De Bra, "Pros and cons of adaptive hypermedia in web-based education," *Cyberpsychology Behav.*, vol. 3, no. 1, pp. 71–77, 2000.

[4] T. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[5] S. Bergin and R. Reilly, "Programming: factors that influence success," in *ACM SIGCSE Bulletin*, 2005, vol. 37, no. 1, pp. 411–415.

[6] P. R. Ventura, "Identifying Predictors of Success for an Objects-First CS1," *Comput. Sci. Educ.*, vol. 15, no. 3, pp. 223–243, 2005.

[7] S. Wiedenbeck, "Factors affecting the success of non-majors in learning to program," in *Proceedings of the first international workshop on Computing education research*, 2005, pp. 13–24.

[8] R. Oboko and P. W. Wagacha, "Using Adaptive Link Hiding to Provide Learners with Additional Learning Materials in a Web-Based System," *J. Res. Cent. Educ. Technol.*, vol. 8, no. 1, pp. 11–25, 2012.

[9] S. KALYUGA, "Managing Cognitive Load in Adaptive ICT-Based Learning."

[10] M. Valcke, "Cognitive load: updating the theory?," *Learn. Instr.*, vol. 12, no. 1, pp. 147–154, 2002.

[11] V. Tsiriga and M. Virvou, "A framework for the initialization of student models in web-based intelligent tutoring systems," *User Model. User-adapt. Interact.*, vol. 14, no. 4, pp. 289–316, 2004.

[12] K. C. Sia, S. Zhu, Y. Chi, K. Hino, and B. L. Tseng, "Capturing user interests by both exploitation and exploration," in *User Modeling 2007*, Springer, 2007, pp. 334–339.

[13] E. Muuro, P. W. Wagacha, and R. Oboko, "Models for Improving and Optimizing Online and Blended Learning in Higher Education," Jared Keengwe and J. J. Agamba, Eds. IGI Global. Pennsylvania, USA, 2014, pp. 204–219.