A[gobble]

DISC, Université de Franche-Comté, CNRS, institut FEMTO-ST, 16 Route de Gray, Besançon, F-25000, Francedaniel.soto_forero@univ-fcomte.frhttps://orcid.org/0000-0003-0753-4673

DISC, Université de Franche-Comté, CNRS, institut FEMTO-ST, 16 Route de Gray, Besançon, F-25000, Francesloubaton@gmail.comhttps://orcid.org/0009-0006-0732-8326

DISC, Université de Franche-Comté, CNRS, institut FEMTO-ST, 16 Route de Gray, Besançon, F-25000, Francemarie-laure.betbeder@univ-fcomte.frhttps://orcid.org/0000-0002-8103-4098

DISC, Université de Franche-Comté, CNRS, institut FEMTO-ST, 16 Route de Gray, Besançon, F-25000, Francejulien.henriet@univ-fcomte.frhttps://orcid.org/0000-0002-7671-4574

Automatic Real-Time Adaptation of Training Session Difficulty Using Rules and Reinforcement Learning in the AI-VT ITS

Daniel Soto Forero[1*], Simha Ackermann[1], Marie-Laure Betbeder[1], Julien Henriet[1]

1 Introduction

Intelligent tutoring systems (ITS) are computer tools that provide services to help in the learning process. The purpose of these systems is to allow learners to acquire knowledge and develop skills in a specific field [1]. Some of these systems include modules with different artificial intelligence algorithms and techniques because it allows training to be more effective, increases the quality through individualized learning pathways, and facilitates the data analysis of all learners [2]. The work described in this paper focuses on AI-VT ITS.

The AI-VT (Artificial Intelligence-Virtual Trainer) ITS is a generic educational tool that aims to accompany learners by proposing sheets of exercises called sessions. Inside each session, the expected abilities are divided into skills—themselves broken down into sub-skills. The learner chooses a skill to work on, and the system generates a session composed of exercises associated with several sub-skills of the chosen skill. The system offers a list of exercises at the beginning of a session using the case-based reasoning paradigm with a database of questions.

Case-based reasoning (CBR) is an artificial intelligence generic paradigm based on a type of reasoning that tries to imitate human behavior to solve new problems by remembering past experiences (analogy reasoning). The basic principle is that similar problems have similar solutions. If a new problem arises, and a similar problem has already been solved, the solution to the new problem can be inferred from the solutions already found for similar problems [3]. Formally we have the set P which is the problem space and S the solution space, then a problem x and its solution y belong to these spaces $x \in P$ and $y \in S$. If y is a solution of x then we have the couple $(x, y) \in P \times S$. In CBR we have a base of n problems and their associated solutions (x^n, y^n). This base is called the case base. The purpose of CBR is, when given a new

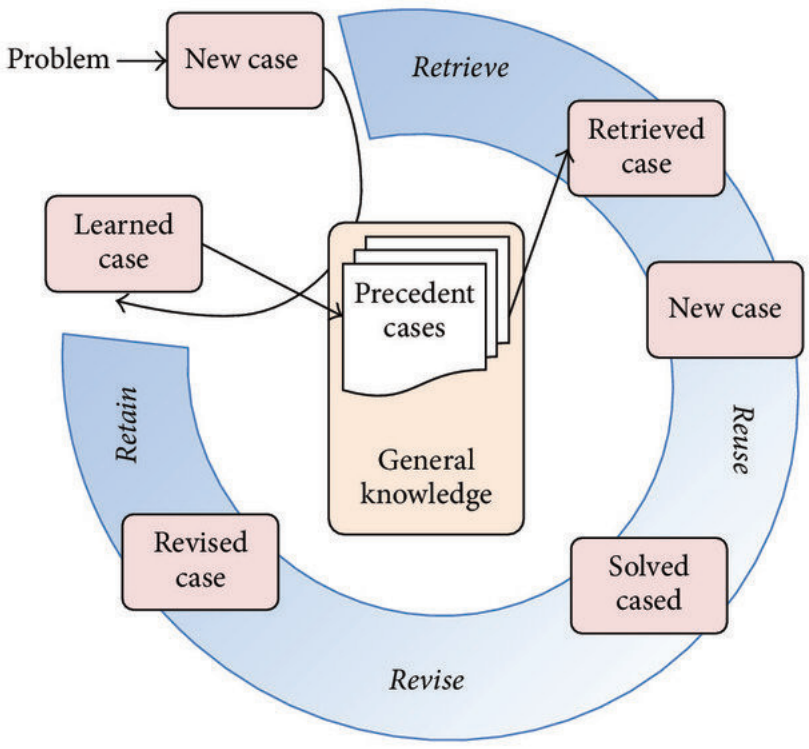

Fig. 1 Case-Based Reasoning Cycle (Source: [6])

problem x^z to find its solution y^z using the case base [4]. The CBR process is divided into four sequential steps that form a cycle. Figure 1 shows the cycle and the flow of information as well as the associations of each of the steps : retrieve, reuse, revise and retain [5].

Recommendation systems in learning environments consider the requirements, needs, profiles, talents, interests, and evolution of the learners to adapt and recommend resources or exercises to improve the acquisition and mastery of concepts and knowledge in general. The adaptation of these systems can be two types: the adaptation of the presentation that shows learners the resources according to their weaknesses and the adaptation of the navigation that changes the structure of the course according to the level and learning style of each learner [7]. The proposed model belongs to the adaptation of navigation category.

Recommendation techniques are useful in ITS because they can detect changes and evolve to an optimal state such as Thompson's sampling (TS) algorithm, a probabilistic type algorithm that belongs to the category of reinforcement learning algorithms, where the algorithm chooses in time t an action a from a set A, gets a reward for the action a and depending on the value of reward, adjusts their decision strategy for choose in time $t + 1$ another a action, with the aim of maximizing the reward. It is based on the Bayesian principle, where there is an a priori probability distribution. With the data obtained, an a posteriori probability distribution is generated with which it tries to maximize the estimate of the expected value. For the Bernoulli variant, where the reward has only two possible values (0 and 1 or success and failure), the base distribution used is the Beta distribution, which is defined in $[0, 1]$ and parameterized by the two values α and β [8].

This paper presents the new recommendation system that has been designed and integrated into AI-VT. This recommendation system is the result of developments based on requirements identified during previous experiments. Indeed, the AI-VT system has been tested in a school, during a session of eight training sessions with two groups of learners. The results have shown the necessity to change the type and difficulty of the exercises in real-time during the session. This ability must take into account the learner's results and the indicator variables produced. As a consequence, a dynamic adaptation module is required. In this case the dynamic adaptation must be able to work and generate adapted recommendations from the cold-start stage and continue with few data. It is also necessary to analyze each learner or group of learners without any previous training stages. In addition, it must also avoid the use of subjective estimators that can generate recommendations deviating from the learner's reality. For this reason, a specific recommendation module is proposed and integrated into the system. This module calculates a factor value based on the grades obtained in the past and their complexity in order to propose questions at a level of complexity adapted to each learner.

The proposed recommender system model in this paper considers the learner's past grades and their complexity to estimate their knowledge and mastery in different skills and sub-skills inside the AI-VT system and then to adapt the sessions to maximize the acquisition of knowledge and the mastery of different areas contained in the same defined skill. The model has two versions—a deterministic version based on rules that establishes the learner's capabilities with defined parameters and intervals and a stochastic version based on the reinforcement learning paradigm that proposes a modification of the Thompson Sampling algorithm to get the adaptation. The versions are complementary and the system can select which of the two versions to use. This choice depends on the results, session difficulty and learning profiles. The effectiveness of the adaptation is variable and, in some cases, the deterministic model works better than the stochastic one and inversely.

This paper is organized as follows: Section II shows the specification of the AI-VT system; Section III presents the related works about intelligent training systems, recommender systems, and adaptation methods; the proposed model is explained in Section IV; Section V shows the experimental description, results and discussion; and lastly, the conclusions are discussed in Section VI.

2 Description of the AI-VT System

The AI-VT system is a general ITS that proposes a set of exercises to learners. This set of exercises is generated by a multi-agent system using the case-based reasoning [9].

The global structure of AI-VT is shown in Figure 2. There is a database of questions. Each one of them is associated with a context, the text of a question, and a complexity level. The questions belong to a sub-skill level, and the sub-skills belong to a skill level. The teacher and the learner are the principal actors in the system. The teacher has the capacity to configure the whole system, number of skills, sub-skills in a skill, number of questions, complexity of each of them, number of complexity levels, and time per session. The learner can start the series of a specific sub-skill, access complementary support resources, and answer the test questions in the sessions proposed by the system.

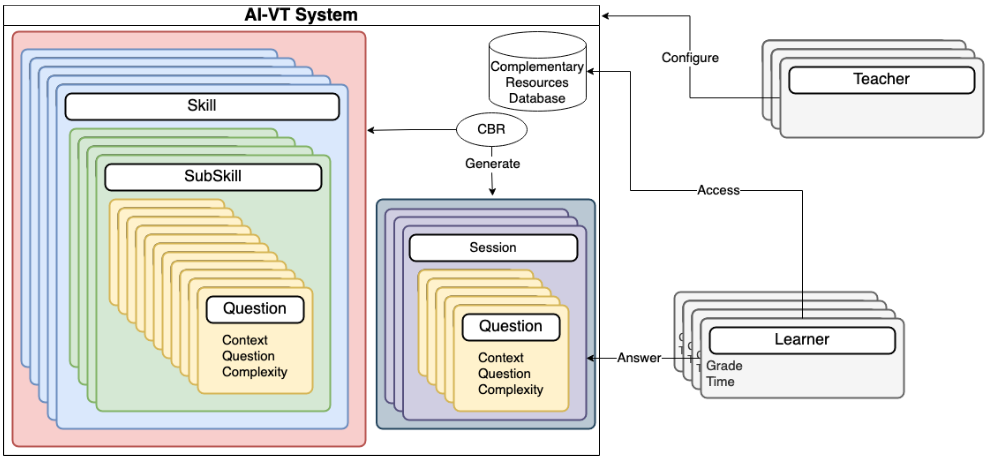

Fig. 2 AI-VT System Structure

Using the CBR philosophy, the global AI-TV system assumes that there are learners with similar learning performances, needs, and abilities. It is then possible to group them and thus improve the general learning process for all of them.

The AI-VT algorithm tries to propose a list of exercises according to the selected time of work, skill, and sub-skill while considering the balance between repetitiveness and variety. The number of exercises per session changes depending on the level and proficiency of each learner. The list of exercises is generated at the beginning of each session; therefore, it does not modify the course of the session depending on the answers provided by the learner, as the exercise lists are static during the session [10].

3 Related Works

3.1 Recommender Systems in ITS

In ITS, the recommender systems show positive effects for learners because they help find relevant resources and maintain motivation. Also, with the application of artificial intelligence techniques to personalization recommendations, the systems are more efficient. The effects were validated by Huang *et al.* [11], measuring the difference between preliminary tests and tests after the completion of a course as well as a control group that took the same course without the recommendation system. Also, Ingkavara *et al.* [12] highlights that technologies and recommender systems can accommodate different needs and aspirations as well as promote self-regulated learning. This type of learning helps learners acquire skills to improve their speed and performance because there are variable objectives, a structured environment, variable learning times, and support and reinforcement resources.

Recommender systems require information about the learner and their requirements so that a module can define a profile and perform a set of algorithms to produce a recommendation. As an example, the work of Lalitha and Sreeja [13] uses the KNN (K-Nearest Neighbors) algorithm to identify the common characteristics between the learners and to extract the adapted resources from the web using Random Forest.

There are different approaches to generating and proposing personalized resources and paths. The algorithms and representations used are diverse. For example, in Zhao *et al.* [14], learner data is collected and classified into groups with similar characteristics to determine the performance of each learner using the Data Envelopment Analysis (DEA) method, which allows for identifying the specific needs of each group and thus proposes a personalized learning path.

The recommendation can propose resources, themes, or exercises. Zhou and Wang [15] develops an exercise recommendation system for learning English. The system contains a main module that represents learners as vectors according to the DINA model where the acquired knowledge points are stored. The vector is n-dimensional $K = \{k_1, k_2, ..., k_n\}$ and each dimension corresponds to a knowledge point. For example, $k_1 = 1$ then the learner masters knowledge point k_1, and if $k_2 = 0$ then the learner has to study knowledge point k_2 because they have not acquired mastery in that point. This representation allows creation of a learner profile that will later be used to recommend which exercises to perform or which resources to consult in order to acquire mastery in the dimensions requiring more knowledge.

Some recommendation and personalization works consider complementary variables to the grades as in Ezaldeen *et al.* [16], which links the analysis of the learner's behavior and semantic analysis. The first step consists in collecting the necessary data to create a learner profile. The complete profile of the learner is associated with a group of predefined learning categories according to their preferences and historical data. Then, having a value for each category, the system searches concepts associated with the categories and generates a guide to obtain resources the system can recommend on the web.

Deep Learning has been used as a recommender system in some intelligent tutoring systems, as in the work of Gomede *et al.* [17], where a model that predicts learners' preferences using deep autoencoders is proposed. The effectiveness of the model has been tested based on learners' preferences, demonstrating that this type of model can indeed improve the adaptation of a learning system with respect to other types of algorithms and combinations evaluated. In this case, this work is based more on learner's experience than on progress in acquiring and mastering knowledge.

Another technique used in order to adapt the intelligent tutoring system consists of predicting the performance of the learners and modifying the system according to the data obtained from the algorithm. In such a case, the prediction can be done with an algorithm or a combination of artificial intelligence models. One of the drawbacks of this type of approach is that, generally, the data used to train the models does not integrate important aspects into the learning process [18]. Many of the adaptation methods and algorithms presented in this section correspond to models designed for specific intelligent tutoring systems that, require variables and data that are not always available, making them not directly applicable in a generic way to other systems. The models using Deep Learning are sometimes very complicated, require a lot of hyperparameters and the architecture is complex and hard to understand.

3.2 Case-Based Reasoning (CBR)

CBR is used in several fields because of its generality. Medicine, sports, and industrial production are the representative fields. The work of Smyth and Cunningham [19] uses CBR for the prediction of running times for athletes. The runners were followed and analyzed to predict the finishing times of a race. Here, an algorithm for those who do the marathon is implemented using the KNN algorithm to find similar cases and make the prediction with the weighted average of the best finishing times of similar cases found in the first stage of the CBR.

The model in Smyth and Willemsen [20] tries to predict the best personal times for skaters based on the analogy that skaters with similar characteristics and racing histories will have similar times, too. Nevertheless, sometimes calculating an average of the similar times found is not enough because there are variables that can considerably change each skater's performance, such as the type of race, a specific track, the race distance, and so forth. Consequently, the authors make the predictions over groups of characteristics such as race distance and date.

CBR is also used as a recommendation system. Bahramian *et al.* [6] implements a tourism recommender system with CBR and artificial neural networks to help users select a tour and visit points of interest in a given city. The system database is updated with the user feedback and the recommendation is generated according to user preferences.

The use of CBR in ITS and personalization shows positive results as indicated by the work of Supic [21] whose model follows the traditional cycle of CBR by combining traditional and digital learning models. The main contributions are the representation of cases and the recommendation of personalized learning paths according to other learners' information. To demonstrate the effectiveness of the model, an initial case base was created to recommend courses to 120 learners. The results were obtained with the help of exams before and after following the course recommended by the model.

3.3 Thompson Sampling (TS)

Recommendation systems are generally used in commerce and on the market to recommend products to customers, but these techniques may be used in other fields to recommend training exercises, planning paths, study resources, etc. These recommendations are based on historical data and user feedback. In Eide *et al.* [22], a Recurrent Neural Network as a recommender system is proposed for a marketplace. The network had been tested in combination with several variants of the Thompson Sampling algorithm because it allows for the maximization of the exploration opportunities and ensures a learning process. With a hit-rate metric, TS improves click rates by 7% compared to other strategies.

The recommendation of products for e-commerce with an extension of the Bernoulli Thompson Sampling algorithm is proposed in Brodén *et al.* [23], where combinations of TS with sleeping bandit and dynamic partitioning are tested with information belonging to a single user in the same session. The results of metrics precision and recall show high values of accuracy for three different types of products.

Another application of TS appears in Akerblom *et al.* [24], where a variant of TS is implemented to find the minimax paths in a network with stochastic weights and partial knowledge. The work with three different test scenarios demonstrates the algorithm's effectiveness despite the complexity and lack of information.

Mao *et al.* [25] developed a recommender system with a reinforcement algorithm that considers learner dynamic preferences in the context of practical quiz question generation. In this case, learners must send a feedback score to the system, which then uses a transformation matrix (distance between questions) and a weight of reward for each question. Some equations calculate the reward for each learner and the direction for newly generated questions. This integrated recommendation system was evaluated with respect to a random system and a greedy system, using an ordinary session in two scenarios (10 and 30 questions) as a basis. The metric used to compare the algorithms is the cumulative reward. In both scenarios the proposed system obtained a higher cumulative reward, demonstrating that the recommendations generated by the proposed system obtain better opinions from the learners.

As a consequence, this study of the literature shows that the AI-VT recommender model requires:

- identification of learners' difficulties in the learning process with few data
- specific adaptation of content or presentation of the system according to the difficulties encountered for each learner and their current grades
- adaptation without initial data (cold-start)
- dynamic responses without training step
- a minimum number of subjective descriptive parameters for each learner

The module proposed in its two versions contributes significantly to the fulfillment of the objectives by making the contents presented by the application appropriate for each learner and by generating diverse adaptations for each particular type of learning. Identification is achieved quickly, in a few steps and is continuously analyzed and adapted according to the new answers provided by the learners.

4 Proposed Model

The proposed model attempts to change the list of exercises generated by CBR, according to the learner's partial results. Consequently, the model requires the grades, answer delays, and question complexities in order to perform the recommendation for one learner. The proposed model aims at identifying the weaknesses of each learner, adapting the content presented to the learners according to the identified weaknesses, and extracting and exploiting data from the answers at each level of complexity even when learners exhibit unexpected behaviors. The model variables and parameters are detailed in Table 1. The model is proposed in two versions, one deterministic and the other stochastic.

The proposed deterministic model calculates the mastery rate with the equations 1, 2 and 3. Equation 1 computes the mean of historic grades for a particular complexity level. This equation contains a penalty term for the response time that may or may not be applied depending on the value of parameter λ. Equation 2 computes the mastery rate for the first complexity level and Equation 3 calculates the mastery rate for each complexity level. The mastery rate helps to guide the learner within the complexity levels using historical grades and to recommend the complexity level closest to the current one where weaknesses have been detected. Mastery rate m_c is a value on a scale of 0 to 100 that reflects the level of proficiency attained by the learner in a sub-skill. It serves to condense the learner's scores for each sub-skill. It is calculated based on the (latest) scores and weighted by the complexity of the exercises. The mastery rate must take into account the complexity of the exercises as follows: A 100% mastery rate cannot be achieved solely with low-complexity exercises; conversely, it can be achieved with maximum-complexity exercises alone. We then propose a tiered system. With c_n levels of complexity, exercises of complexity c

should enable the attainment of a maximum mastery rate of $c * 100/c_n$. Finally, the mastery rate is calculated recursively.

Table 1 Variables and parameters of the proposed model

ID	Description	Domain
c_n	Total number of complexity levels	$\mathbb{N} \mid c_n > 0$
g_m	Max value into the scale of grades	$\mathbb{N} \mid g_m > 0$
g_t	Grade threshold	$(0, g_m) \in \mathbb{R}$
s	Number of defined paths	$\mathbb{N} \mid s > 0$
s_c	Defined fixed current path	$[1, s] \in \mathbb{N}$
Δs	Step for beta distribution parameters in path s	$(0, 1) \in \mathbb{R}$
t_m	Max value in time answer	$\mathbb{R} \mid t_m > 0$
nv	Number of intervals for each path	$\mathbb{N} \mid nv > 0$
g_c	Grade for a learner for a question with complexity c	$[0, g_m] \in \mathbb{R}$
ng_c	Grade of learner with time penalization	$[0, g_m] \in \mathbb{R}$
t_c	Time to answer a question with complexity c	$[0, t_m] \in \mathbb{R}$
m_c	Mastery rate of learner for a complexity c	$[0, 100] \in \mathbb{R}$
$v_{s,nv}$	Limits for each nv interval	$[0, 100]$
n_q	Number of questions to consider of history	$\mathbb{N}$
$f(g)$	Function to calculate the mean of grades g	$\mathbb{R}_+$
ncl	New calculated complexity level	$\mathbb{N}$
α_c	Value for α in complexity c	$\mathbb{R} \mid \alpha_c > 0$
β_c	Value for β in c complexity	$\mathbb{R} \mid \beta_c > 0$
$\Delta\beta$	Step of beta parameter initialization	$\mathbb{N} \mid \Delta\beta > 0$
λ	Weight of time penalization	$(0, 1) \in \mathbb{R}$
G_c	Set of d grades in complexity level c	$\mathbb{R}^d$, $d \in \mathbb{N} \mid d > 0$
x_c	Normalized average grades	$[0, 1] \in \mathbb{R}$
n_c	Number of total questions in a session	$\mathbb{N} \mid n_c > 0$
ny_c	Number of questions in complexity level c	$\mathbb{N} \mid 0 < ny_c \leq n_c$
y_c	Proportion of questions in complexity level c	$[0, 1] \in \mathbb{R}$
r	Total value for adaptability defined metric	$[0, c_n] \in \mathbb{R}$
sc	Total value for cosine similarity metric	$[-1, 1] \in \mathbb{R}$

$$f(g_c) = < g_c > - \left(< g_c > * \lambda * \frac{< t_c >}{t_m} \right) \tag{1}$$

$$m_1 = \frac{10}{c_n} * f(g_1)_{n_q} \tag{2}$$

$$m_c = max \left(m_{c-1} + \frac{10}{c_n} * f(g_c)_{n_q}, \frac{c * 10}{c_n} * f(g_c)_{n_q} \right) \tag{3}$$

With the calculation of the mastery rate for the last level of complexity, it is possible to recommend an adapted complexity level using a predefined strategy. Equation 4 uses an indicator function I_c to determine the adapted complexity level using a table $v_{s,nv}$ of predetermined intervals.

$$ncl = I_c(inf(v_{s,nv}) <= m_c <= sup(v_{s,nv})), \; \forall nv \tag{4}$$

The complete steps for the deterministic algorithm are shown in Algorithm 1.

Algorithm 1 Deterministic Recommendation Model

```
for each $n_q$ questions $q$ do
    $< g >\leftarrow \frac{1}{n_q}\sum_{i\in A} g_i$                ▷ $A$ the set of the last $n_q$ grades
    $r \leftarrow f(g_c)_{n_q}$                                      ▷ eq 1
    Calculate $m_c$ with $r$                                         ▷ for last complexity level $c$, eq 3
    for each $nv_s$ do
        Get $ncl$                                                    ▷ Using the $v$ table, eq 4
    end for
end for
```

The proposed stochastic model uses the Beta family of distribution of probability to define dynamically the new complexity level (Equation 5) inspired by the Thompson Sampling reinforcement learning algorithm. This model version allows the recommendation of non-contiguous complexity levels, but the priority is to recommend levels where faults have been detected. The model requires a distribution of probability for each complexity level, the initial parametrization of all distributions of probability can force the model to recommend contiguous complexity levels.

$$B(x,\alpha,\beta) = \begin{cases} \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}du} & for\ x \in [0,1] \\ 0 & otherwise \end{cases} \tag{5}$$

In this case, the grade threshold variable g_t is necessary to determine the variability of the distribution of probability for each complexity level. All the distributions of probability are dynamic, Equations 6, 7 and 8 show the correlated update rules. These rules modify the values by inverse reward. Each complexity level has an associated Beta probability distribution with predefined initial values for the parameters α and β. Equation 6 computes the last grade in a specific complexity level of a learner and applies a penalty for response time, according to the λ parameter. Equations 7 and 8 update the α and β values using the threshold variable g_t as an indicator.

$$ng_c = g_c - \left(g_c * \lambda * \frac{t_c}{t_m}\right) \tag{6}$$

$$ng_c \geq g_t \rightarrow \begin{cases} \beta_c = \beta_c + \Delta_s \\ \beta_{c-1} = \beta_{c-1} + \frac{\Delta_s}{2} \\ \alpha_{c+1} = \alpha_{c+1} + \frac{\Delta_s}{2} \end{cases} \tag{7}$$

$$ng_c < g_t \rightarrow \begin{cases} \alpha_c = \alpha_c + \Delta_s \\ \alpha_{c-1} = \alpha_{c-1} + \frac{\Delta_s}{2} \\ \beta_{c+1} = \beta_{c+1} + \frac{\Delta_s}{2} \end{cases} \tag{8}$$

Once the parameters of the probability distributions have been defined for each level of complexity, as shown in Equation 9, the adaptation is chosen by considering the maximum value of the set of random values obtained from the all the Beta probability distributions.

$$ncl = max_x(\mathbb{E}[Beta(\alpha_x, \beta_x)]), 0 <= x <= c_n \tag{9}$$

The steps for the stochastic algorithm are shown in the algorithm 2.

Algorithm 2 Stochastic Recommendation Model

```
Initialize the a priori distributions of probability
for each questions $q$ do
    With $i$ as the actual complexity level $c$
    Calculate $ng_i$                                  ▷ eq 6
    Update parameters $\alpha_i$ and $\beta_i$        ▷ eq 7 and eq 8
    Get $ncl$                                         ▷ eq 9
end for
```

5 Results and Discussion

In this section a comparison is presented of the AI-VT system with and without the proposed recommendation module (deterministic and stochastic). The behavior of the model was tested with a generated dataset, because the synthetic data can be an efficient way to compare models, allows the exploration of the impact of sample size in the performance of the model, protects sensible and personal information, and can reproduce representative information [28]. This dataset contains the grades and response times of 1000 learners for five different levels of complexity. The data description is shown in Table 2. The approximation of the learner grades is generated with the logit-normal distribution of probability because it is experimentally the best representation model [26].

The defined general test parameters for the model are detailed in Table 3. The specific parameters for the deterministic variant are in Table 4. Using these parameters, the model is formally written as Equations 10, 11, 12, 13 and 14. Table 6 shows the definition v of intervals. The stochastic variant parameter values are in Table 5.

Table 2 Description of variables for dataset used

ID	Description	Domain
q_c	Complexity Level of a question q	$[0, c_n] \in \mathbb{N}$
$q_{g,c}$	Obtained grade g for question q with complexity c	$[0, g_m] \in \mathbb{R}$
$q_{t,c}$	Time employed t for a question q with complexity c	$[0, t_m] \in \mathbb{R}$

Table 3 Values for tested scenarios

ID	c_n	g_m	t_m	s	s_c	λ
Value	5	10	120	3	2	0.25

Table 4 Initial values for the deterministic recommendation model

ID	nv_1	nv_2	nv_3	n_q	$v_{s,nv}$
Value	8	7	7	3	Table 6

Table 5 Initial values for the stochastic recommendation model (x represents all learners, y represents all complexity levels greater than 1)

ID	g_t	$\alpha_{x,1}$	$\alpha_{x,y}$	$\beta_{x,1}$	$\Delta\beta_{x,y}$	Δ_1	Δ_2	Δ_3
Value	6	2	1	1	1	0.3	0.5	0.7

Table 6 Table v with the values for three paths ($s = 3$), sevent to eight intervals ($nv_1 = 8, nv_2 = 7$ *and* $nv_3 = 7$) and five CL (Complexity Levels)

Path 1		Path 2		Path 3	
Interval	CL	Interval	CL	Interval	CL
$[0, 20]$	0	$[0, 15]$	0	$[0, 10]$	0
$[21, 30]$	1	$[16, 25]$	1	$[11, 20]$	1
$[31, 45]$	1	$[26, 35]$	2	$[21, 30]$	2
$[46, 50]$	2	$[36, 42]$	2	$[31, 36]$	2
$[51, 65]$	2	$[43, 50]$	3	$[37, 43]$	3
$[66, 75]$	3	$[51, 75]$	3	$[44, 65]$	3
$[76, 90]$	3	$[76, 100]$	4	$[66, 100]$	4
$[91, 100]$	4				

$$m_1 = \frac{10}{5} * f(g_1)_3 \tag{10}$$

$$m_2 = max\left(m_1 + \frac{10}{5} * f(g_2)_3, \frac{20}{5} * f(g_2)_3\right) \tag{11}$$

$$m_3 = max\left(m_2 + \frac{10}{5} * f(g_3)_3, \frac{30}{5} * f(g_3)_3\right) \tag{12}$$

$$m_4 = max\left(m_3 + \frac{10}{5} * f(g_4)_3, \frac{40}{5} * f(g_4)_3\right) \tag{13}$$

$$m_5 = max\left(m_4 + \frac{10}{5} * f(g_5)_3, \frac{50}{5} * f(g_5)_3\right) \tag{14}$$

The generated dataset is a simulation of learners grades for answers to fifteen questions at each of the five levels of complexity. The dataset simulates, via the logit-normal probability distribution, a weakness in each level of complexity for 70% of learners in the first ten questions. The difficulty of the complexity is also simulated by reducing the average score and increasing the variance. Figure 3 shows the dataset distribution of 1000 learners' grades by complexity level. The results shown for the stochastic variant corresponds to the average of 20 executions.

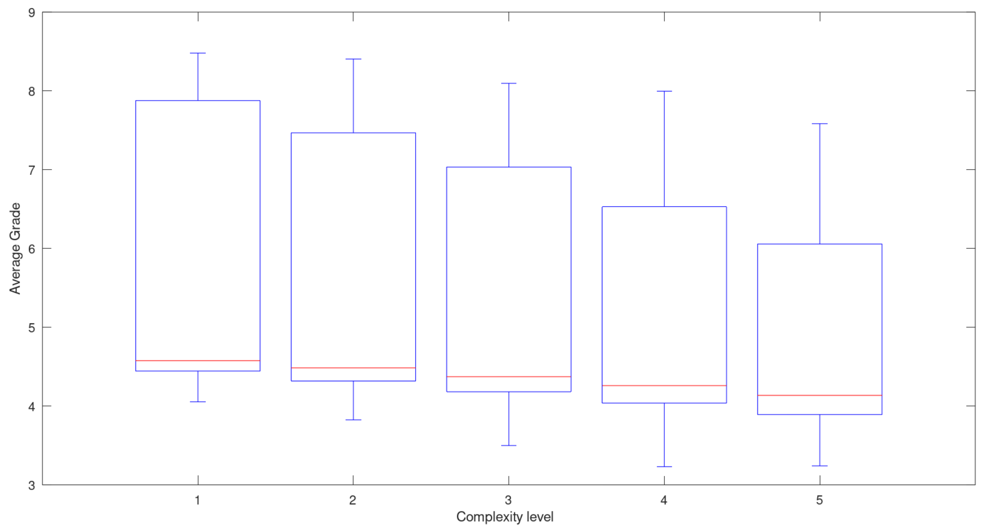

Fig. 3 Generated dataset for testing

The results of the first comparison without historical data (cold start) between the two versions of the proposed model and the original system (CBR) are shown in Figure 4, where different transition numbers and scales appear. The original system does not have transitions. All the learners are evaluated inside complexity level 0. Grades obtained during the session are not considered. The system with recommendation models tries to adapt the complexity level according to the obtained grades. The deterministic model generates four big transitions with a large number of learners in questions 5, 6, 8 and 12 all of them between contiguous complexity levels. The trends are downward for levels 0, 1, and 2 after the eighth question and upward for levels 1 and 3. The stochastic model starts by proposing all the possible levels of complexity but focuses on level 0. The transitions are constants, but for a small number of learners, the trends after the tenth question are downward for levels 0 and 4 and upward for levels 1, 2 and 3.

After the generation of the first session, the system can continue with the next list of exercises. In this case, the three models have been initialized with the same data and equal values for all learners. Figure 5 allows observation of the first transition of the original system. This transition shows that the system acts only with the past obtained grades and that the transitions are very slow. Even if the grades are different during the session, all learners must follow the same path; however, the recommendation models change. The deterministic model presents three transitions in questions

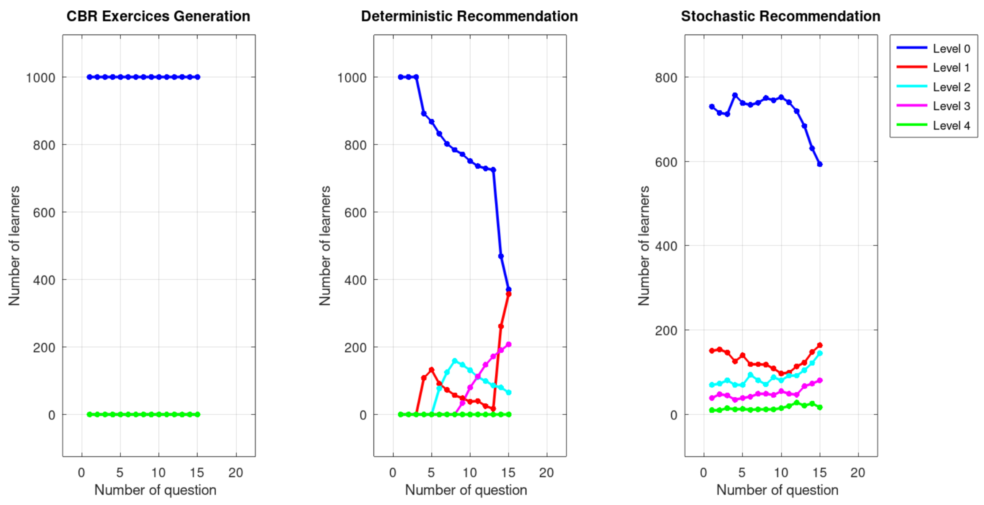

Fig. 4 Recommended complexity levels with two version models, first session case without historical grades data (cold start)

3, 5, and 12. The trends are static for level 3, variable for Level 2, and strongly downward for level 0. The stochastic model continues with smooth transitions but always tries to prefer the weaker level. In this case, the model has identified complexity level 1. Here, levels 0 and 1 are downward, level 2 is static, and levels 3 and 4 are upward.

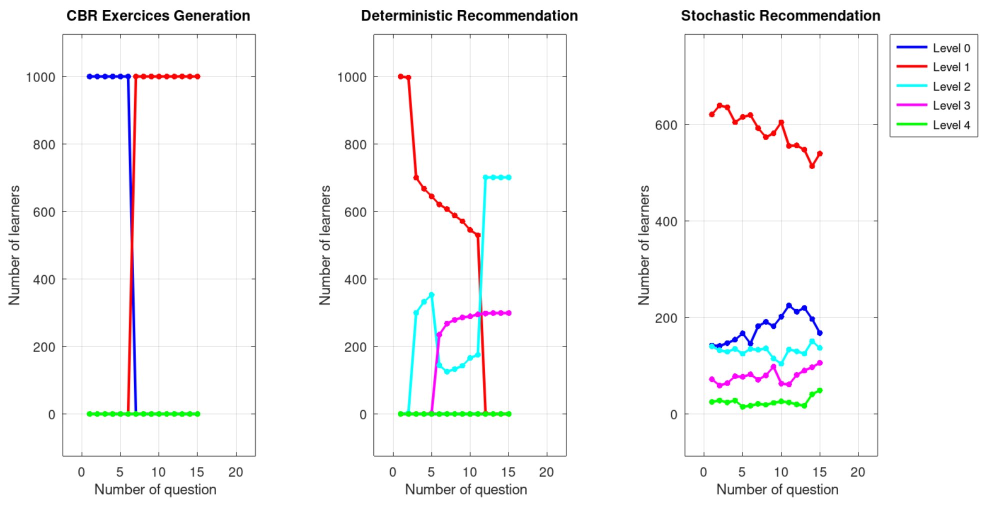

Fig. 5 Recommended complexity levels with two version models; third session case with historical grades data for complexity level 0

Finally, when the initialization data considers the two complexity levels 0 and 1 as evaluated, then naturally, the system must start with level 1 or 2. Because the original system changes very slowly between levels, this system starts with complexity level 1 as shown in Figure 6. Like the other two comparisons, the changes in this system are not progressive but direct for all. The deterministic recommendation model in this case adopts the same strategy and proposes a direct change for all learners around the fifth question. The stochastic variant continues with small, constant changes and a preference for level 2. The trends are very stable except for levels 1 (upward) and 2 (downward).

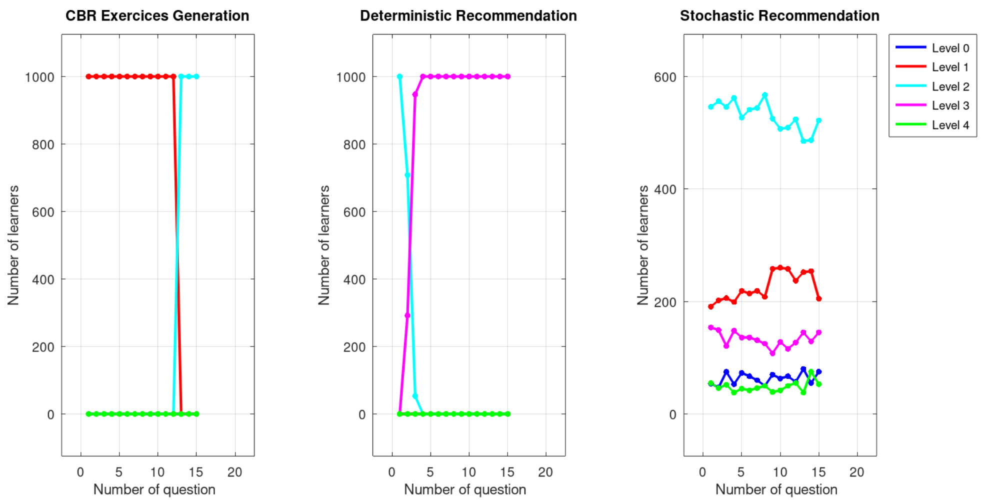

Fig. 6 Recommended complexity levels with two version models; sixth session case with historical grades data for complexity levels 0 and 1

A representative example of recommended complexity level progression for two specific learners with the deterministic model is shown in Figure 7 and with the stochastic model in Figure 8, where the value of average grades in the session x_c, and the proportion of questions for each complexity level y_c are calculated in each case. In both cases, the learner evolution between the different complexity levels is notable. The main difference is the type of change. The deterministic model proposes sequential changes by shifting between consecutive levels in a bottom-up fashion. On the other hand, the stochastic model proposes jumps between levels until a convergence is reached while maintaining the upward trend.

To compare the original system and two recommendation model variants numerically, a set of equations (equation 15 and equation 16) has been defined that describes the ideal recommender system if the learner's objective is standard learning. The metric calculates a value for each complexity level according to grade averages and the number of recommended questions in that complexity level. The purpose of this metric is to give a high score to the recommender systems that propose more exercises at the complexity level where the learner has registered a lower average grade with the idea of reinforcing the knowledge at that complexity level and if they

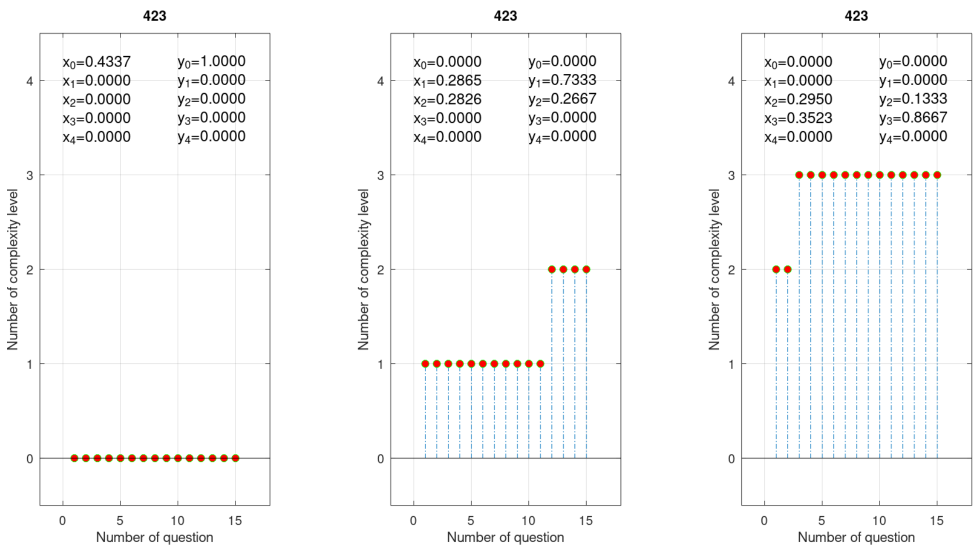

Fig. 7 Example of a deterministic recommendation for a representative learner in the three configured scenarios

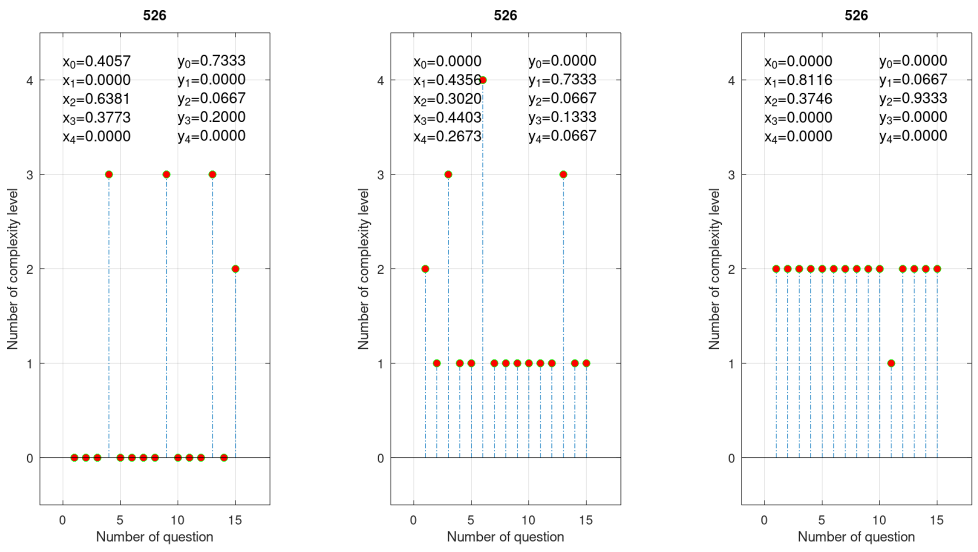

Fig. 8 Example of a stochastic recommendation for a representative learner in the three configured scenarios

propose fewer exercises at complexity levels where the average grade is high because it is assumed that the learner has already acquired sufficient knowledge at those complexity levels. Low scores are assigned to systems that recommend few exercises at complexity levels with low average grades and, conversely, if they propose many exercises at complexity levels with high average grades.

$$rp_c = e^{-2(x_c+y_c-1)^2} \tag{15}$$

$$r = \sum_{c=0}^{c_n-1} rp_c \tag{16}$$

In equation 15, x_c is the normalized average of grades in complexity level c (equation 17), and y_c is the normalized number of answered questions in complexity level c (equation 18).

$$x_c = \frac{< g_c >_{G_c}}{g_m} \tag{17}$$

$$y_c = \frac{ny_c}{n_c} \tag{18}$$

Figure 9 shows the global equation for the metric rp inside the domain of two variables x_c and y_c. The maximum value for r in a specific complexity level is 1. The global maximum value for the tested scenarios is 5; therefore a good recommender system should have a high r value.

The results of metric calculations for the original system and the two models in the three defined scenarios are shown in Table 7.

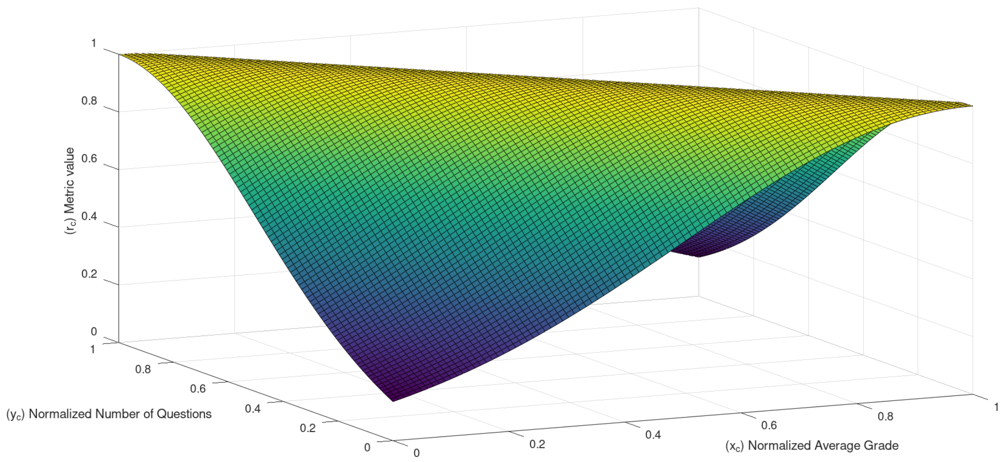

Fig. 9 Function for metric evaluations in each complexity level (standard learning)

A metric for soft learning is defined in equation 19 and equation 20. A high score is assigned using this metric to the systems that propose more exercises in a complexity level where the average grade is around 0.4, and also if there are fewer exercises when average grades are higher, and the score value is more flexible with lower average

Table 7 Results Metric Table (CBR - System without recommendation model, DM - Deterministic Model, SM - Stochastic Model). A higher value is better

	c_0	c_1	c_2	c_3	c_4	Total (r)	Total (%)
Test 1							
CBR	0.5388	-	-	-	-	0.5388	10.776
DM	0.8821	0.7282	**0.9072**	**0.8759**	-	3.3934	67.868
SM	**0.9463**	**0.8790**	0.7782	0.7108	0.6482	**3.9625**	79.25
Test 2							
CBR	0.9445	**0.9991**	-	-	-	1.9436	38.872
DM	-	0.9443	**0.8208**	**0.9623**	-	2.7274	54.548
SM	**0.9688**	0.9861	0.8067	0.7161	0.6214	**4.0991**	81.982
Test3							
CBR	-	0.8559	0.7377	-	-	1.5936	31.872
DM	-	-	0.5538	**0.7980**	-	1.3518	27.036
SM	0.9089	**0.9072**	**0.9339**	0.7382	0.6544	**4.1426**	82.852

grades. The low scores are assigned to systems that recommend a higher number of questions in a complexity level with high average grades, and if there are too many or not enough recommended exercises when grades are lower.

$$rs_c = e^{-\frac{2}{100}(32x_c^2-28x_c+10y_c-4)^2} \tag{19}$$

$$r = \sum_{c=0}^{c_n-1} rs_c \tag{20}$$

Figure 10 shows the global equation for the metric rs inside the domain of two variables x_c and y_c. The maximum value for r in a specific complexity level is 1. The global maximum value for the tested scenarios is 5; therefore a good recommender system should have a high r value.

The results of the metric calculation for the original system and two models in the three defined scenarios are shown in Table 8.

To compare the original system and the recommendation model, the metric for the diversity of propositions is used with cosine similarity (the cosine similarity between a vector A and a vector B, equation 21) between all the learners propositions. The results of the average cosine similarity are shown in Table 9.

$$sc = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{21}$$

With CBR exercise generation, the system proposes the same exercises to all learners, and the evolution of complexity levels is very slow almost one change every three or four sessions. This is because the system does not take into account the grades obtained during the session, but only at the end. The implemented recommender systems are more dynamic because big changes occur in groups or individually, and

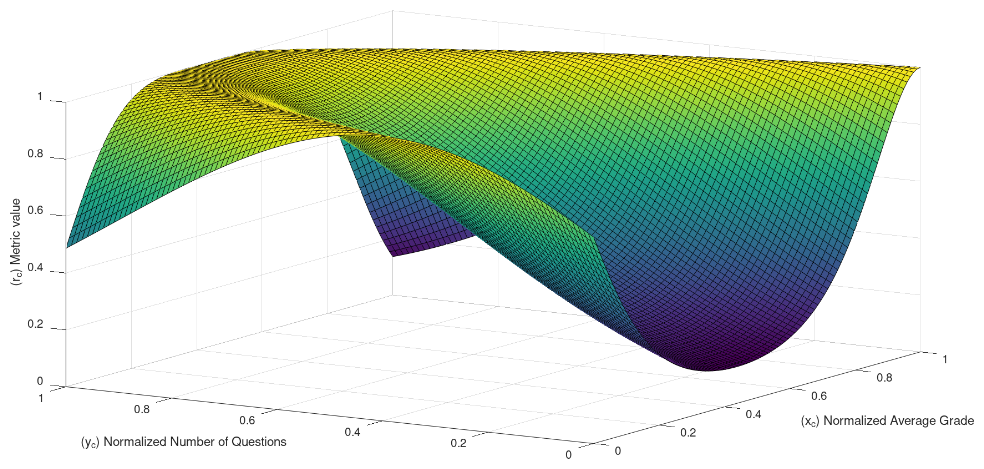

Fig. 10 Function for the metric evaluation in each complexity level (soft learning)

Table 8 Results Metric Table (CBR - System without recommendation model, DM - Deterministic Model, SM - Stochastic Model). A higher value is better

	c_0	c_1	c_2	c_3	c_4	Total (r)	Total (%)
Test 1							
CBR	**0.9979**	-	-	-	-	0.9979	
DM	0.8994	0.1908	**0.3773**	**0.2990**	-	1.7665	
SM	0.8447	**0.3012**	0.2536	0.2030	**0.1709**	**1.7734**	
Test 2							
CBR	**0.4724**	**0.7125**	-	-	-	1.1849	
DM	-	0.6310	**0.3901**	**0.4253**	-	1.4464	
SM	0.2697	0.7089	0.2634	0.2026	**0.1683**	**1.6129**	
Test3							
CBR	-	**0.9179**	0.2692	-	-	1.1871	
DM	-	-	0.2236	**0.9674**	-	1.191	
SM	0.1873	0.3038	**0.6345**	0.2394	**0.1726**	**1.5376**	

Table 9 Average diversity of propositions for all the learners. A lower value represents greater diversity. (CBR - System without recommendation model, DM - Deterministic Model, SM - Stochastic Model)

Model	Scenario 1	Scenario 2	Scenario 3
CBR	1	1	1
DM	0.9540	0.9887	0.9989
SM	0.8124	0.8856	0.9244

the evolution is faster. As can be seen, these adaptations in levels of complexity change within a few sessions. But considering the learner grades, the deterministic model suggests level changes to a large number of learners suddenly, because they are grouped inside an interval of mastery rate, while the stochastic model is more focused on individual personalization, and the complexity level changes are produced for a small number of learners. The two proposed models have the ability to detect a learner's weakness and direct the session for their particular needs. The generated database has allowed the simulation of various situations with the grades of 1000 learners, thus allowing evaluation of the behavior of recommender systems with different configurations.

The numerical results using the defined metric show that the distributions of questions in a session by the two versions of the recommendation model are different but with a similar overall trend for all learners. The deterministic model is more focused in the two contiguous complexity levels where weaknesses have been detected. That is the reason why it has scored higher in some specific complexity levels. The stochastic model attempts to distribute the questions in all the defined complexity levels. Globally, the stochastic model has obtained a higher score with the defined metric. Compared to the original system, the recommender model (deterministic and stochastic versions) obtains an overall increase in adaptability between 15% and 68% for all complexity levels.

According to the cosine similarity metric, the recommendation model in its two versions increases the diversity of propositions with respect to the original system in the three scenarios evaluated, which indicates that in addition to achieving adaptability, the weaknesses of each learner have been identified individually and personalized proposals have been generated with respect to these changes while maintaining the objective of making learners advance between levels of complexity. The results obtained with this metric also indicate that even if the learners present similar results, the two proposed versions have the capacity to generate different but convergent adaptations in global behavior. These observations tend to prove that the identification of each model is different but can be complementary.

In addition, these results show that the adaptation model allows changing the trend by recommending learners and groups of learners move forward or backward in adjacent levels of complexity. Indeed, the model continuously calculates the adaptation according to the data obtained in real-time. This type of behavior is difficult to obtain with other types of artificial intelligence tools, because such tools generate results that follow a fixed trend determined by training data and can hardly take into account new situations.

The proposed model manages to meet the objectives set. Indeed, according to the results obtained in the tests carried out, an improvement in the identification of learners and the subsequent adaptation proposed with respect to the learning system without real-time adaptation models is effectively obtained. The advantages of the proposed model are that it:

- requires few subjective parameters compared to other adaptation models found in the literature
- exploits data obtained directly from the assessment
- does not require learners to provide additional information

- is a generic method that can be applied in various contexts, subjects and even other intelligent tutoring systems

In this study, two metrics are proposed in order to measure the effectiveness of a referral system. These metrics are computed according to the number of training session suggested questions and the grade obtained by the learners. With these metrics, an ITS such as AI-VT can measure the degree to which a potential adaptation identifies learners' weaknesses at a given level of complexity and in a particular (sub-)skill.

6 Conclusions

Modules of recommendation are an essential piece for ITS systems because they help guide the individual learning process, allow identification of weaknesses, and redirect the complete process to improve knowledge and skills. The proposed model versions can detect a learner's weakness in real time and try to redirect the session to the best possible complexity level to help the learner acquire and master the knowledge before moving to superior complexity levels, because, generally, knowledge of inferior complexity levels is necessary to complete superior levels. Even if the dataset generated is a simulation of response times and grades for learners, the tests using it allow observation of the flexibility and robustness of the proposed recommendation model because the data for learners presents high diversity and forces the system to adapt to different types of configurations; therefore, we can conclude that the proposed recommendation model has the capacity to work in different situations and to propose alternative paths in each case to improve the overall learning process—even if the learning objective is different for each learner as demonstrated by the results obtained in the evaluation of the two proposed metrics. The proposed model also allows the diversity and customization of the system, since according to the results of comparison with the cosine similarity between all the recommendations generated for each learner, there is an increase with respect to the original system.

Future work includes tests with real data in a real standard environment to obtain information on certain, specific characteristic groups of learners and to compare the results obtained with theoretical ones, thus consolidating the behavioral evidence and the effectiveness of the proposed model. For AI-VT, this could include a module for the automatic correction of learner answers, provision of complementary aids, analysis of different interactions with the system to improve the learning process, and the identification of skill weaknesses and prediction of performances.

References

[1] Nkambou, R., Bourdeau, J. & Mizoguchi, R. Advances in Intelligent Tutoring Systems. (Springer Berlin, Heidelberg,2010)

[2] Tapalova, O. & Zhiyenbayeva, N. Artificial Intelligence in Education: AIEd for Personalised Learning Pathways. *Electronic Journal Of E-Learning.* pp. 15 (2022), https://eric.ed.gov/?q=Artificial+Intelligence+in+Education

[3] Roldan Reyes, E., Negny, S., Cortes Robles, G. & Le Lann, J. Improvement of online adaptation knowledge acquisition and reuse in case-based reasoning: Application to process engineering design. *Engineering Applications Of Artificial Intelligence.* **41** pp. 1-16 (2015), https://www.sciencedirect.com/science/article/pii/S0952197615000263

[4] Lepage, Y., Lieber, J., Mornard, I., Nauer, E., Romary, J. & Sies, R. The French Correction: When Retrieval Is Harder to Specify than Adaptation. *Case-Based Reasoning Research And Development.* pp. 309-324 (2020)

[5] Richter, M. & Weber, R. Case-Based Reasoning (A Textbook). (Springer-Verlag GmbH,2013)

[6] Bahramian, Z., Ali Abbaspour, R. & Claramunt, C. A Cold Start Context-Aware Recommender System for Tour Planning Using Artificial Neural Network and Case Based Reasoning. (2017), https://doi.org/10.1155/2017/9364903

[7] Muangprathub, J., Boonjing, V. & Chamnongthai, K. Learning recommendation with formal concept analysis for intelligent tutoring system. *Heliyon.* **6**, e05227 (2020), https://www.sciencedirect.com/science/article/pii/S2405844020320703

[8] Lin, B. Evolutionary Multi-Armed Bandits with Genetic Thompson Sampling. *2022 IEEE Congress On Evolutionary Computation (CEC).* pp. 1-8 (2022)

[9] Henriet, J., Christophe, L. & Laurent, P. Artificial Intelligence-Virtual Trainer: An educative system based on artificial intelligence and designed to produce varied and consistent training lessons. *Proceedings Of The Institution Of Mechanical Engineers, Part P: Journal Of Sports Engineering And Technology.* **231**, 110-124 (2017), https://doi.org/10.1177/1754337116651013

[10] Henriet, J. & Greffier, F. AI-VT: An Example of CBR that Generates a Variety of Solutions to the Same Problem. *Case-Based Reasoning Research And Development.* pp. 124-139 (2018)

[11] Huang, A., Lu, O. & Yang, S. Effects of artificial Intelligence–Enabled personalized recommendations on learners' learning engagement, motivation, and outcomes in a flipped classroom. *Computers And Education.* **194** pp. 104684 (2023), https://www.sciencedirect.com/science/article/pii/S036013152200255X

[12] Ingkavara, T., Panjaburee, P., Srisawasdi, N. & Sajjapanroj, S. The use of a personalized learning approach to implementing self-regulated online learning. *Computers And Education: Artificial Intelligence.* **3** pp. 100086 (2022), https://www.sciencedirect.com/science/article/pii/S2666920X22000418

[13] Lalitha, T. & Sreeja, P. Personalised Self-Directed Learning Recommendation System. *Procedia Computer Science.* **171** pp. 583-592 (2020), https://www.sciencedirect.com/science/article/pii/S1877050920310309, Third International Conference on Computing and Network Communications (CoCoNet'19)

[14] Zhao, L., Wang, D., Liang, F. & Chen, J. A recommendation system for effective learning strategies: An integrated approach using context-dependent DEA. *Expert Systems With Applications.* **211** pp. 118535 (2023), https://www.sciencedirect.com/science/article/pii/S0957417422016104

[15] Zhou, L. & Wang, C. Research on Recommendation of Personalized Exercises in English Learning Based on Data Mining. *Scientific Programming.* **2021** pp. 5042286 (2021,12), https://doi.org/10.1155/2021/5042286

[16] Ezaldeen, H., Misra, R., Bisoy, S., Alatrash, R. & Priyadarshini, R. A hybrid E-learning recommendation integrating adaptive profiling and sentiment analysis. *Journal Of Web Semantics.* **72** pp. 100700 (2022), https://www.sciencedirect.com/science/article/pii/S1570826821000664

[17] Gomede, E., De Barros, R. & Souza Mendes, L. Deep auto encoders to adaptive E-learning recommender system. *Computers And Education: Artificial Intelligence.* **2** pp. 100009 (2021), https://www.sciencedirect.com/science/article/pii/S2666920X21000035

[18] Chiu, T., Xia, Q., Zhou, X., Chai, C. & Cheng, M. Systematic literature review on opportunities, challenges, and future research recommendations of artificial intelligence in education. *Computers And Education: Artificial Intelligence.* **4** pp. 100118 (2023), https://www.sciencedirect.com/science/article/pii/S2666920X2200073X

[19] Smyth, B. & Cunningham, P. An Analysis of Case Representations for Marathon Race Prediction and Planning. *Case-Based Reasoning Research And Development.* pp. 369-384 (2018)

[20] Smyth, B. & Willemsen, M. Predicting the Personal-Best Times of Speed Skaters Using Case-Based Reasoning. *Case-Based Reasoning Research And Development.* pp. 112-126 (2020)

[21] Supic, H. Case-Based Reasoning Model for Personalized Learning Path Recommendation in Example-Based Learning Activities. *2018 IEEE 27th International Conference On Enabling Technologies: Infrastructure For Collaborative Enterprises (WETICE).* pp. 175-178 (2018)

[22] Eide, S., Leslie, D. & Frigessi, A. Dynamic slate recommendation with gated recurrent units and Thompson sampling. (2022), https://doi.org/10.1007/s10618-022-00849-w

[23] Brodén, B., Hammar, M., Nilsson, B. & Paraschakis, D. Ensemble Recommendations via Thompson Sampling: An Experimental Study within e-Commerce. *23rd International Conference On Intelligent User Interfaces.* pp. 19-29 (2018), https://doi.org/10.1145/3172944.3172967

[24] Akerblom, N., Hoseini, F. & Haghir Chehreghani, M. Online learning of network bottlenecks via minimax paths. (2023), https://doi.org/10.1007/s10994-022-06270-0

[25] Mao, K., Dong, Q., Wang, Y. & Honga, D. An Exploratory Approach to Intelligent Quiz Question Recommendation. *Procedia Computer Science.* **207** pp. 4065-4074

(2022), https://www.sciencedirect.com/science/article/pii/S1877050922013631, Knowledge-Based and Intelligent Information Engineering Systems: Proceedings of the 26th International Conference KES2022

[26] Arthurs, N., Stenhaug, B., Karayev, S. & Piech, C. Grades Are Not Normal: Improving Exam Score Models Using the Logit-Normal Distribution. *International Conference On Educational Data Mining (EDM)*. pp. 6 (2019), https://eric.ed.gov/?id=ED599204

[27] Zhang., K. & Aslan, A. AI technologies for education: Recent research and future directions. *Computers And Education: Artificial Intelligence*. **2** pp. 100025 (2021), https://www.sciencedirect.com/science/article/pii/S2666920X21000199

[28] James, S., Harbron, C., Branson, J. et al. Synthetic data use: exploring use cases to optimise data utility. Discov Artif Intell 1, 15 (2021). https://doi.org/10.1007/s44163-021-00016-y