Explainable Recommendations in a Personalized Programming Practice System

Jordan Barria-Pineda[1], Kamil Akhuseyinoglu[1], Stefan Želem-Ćelap[2], Peter Brusilovsky[1], Aleksandra Klasnja Milicevic[2], and Mirjana Ivanovic[2]

[1] University of Pittsburgh, Pittsburgh, PA, USA
jab464@pitt.edu
[2] University of Novi Sad, Novi Sad, Serbia

Abstract. This paper contributes to the research on explainable educational recommendations by investigating explainable recommendations in the context of personalized practice system for introductory Java programming. We present the design of two types of explanations to justify recommendation of next learning activity to practice. The value of these explainable recommendations was assessed in a semester-long classroom study. The paper analyses the observed impact of explainable recommendations on various aspects of student behavior and performance.

Keywords: Educational recommender systems · Explanations.

1 Introduction

The popularity of recommender systems in everyday-life activities encouraged researchers from several areas to explore the applications of recommender technologies to education [20,8]. Over the years, this stream of research gradually expanded to cover a variety of recommendation types from recommending a discussion thread, to suggesting the next problem to solve, to recommending courses to take [7]. It can be observed, however, that a recent major trend in the field of recommender systems – explainable recommendations – is currently underrepresented in educational recommender systems. Our work attempts to bridge this gap by exploring explainable recommendations specifically adapted to an educational context, where the main reason to recommend content is the learner's knowledge state, rather than taste or interests. We present a design of knowledge-based recommendations augmented with visual and verbal explanations. These technologies were integrated into an online personalized practice system and explored a semester-long study in a classroom context. The study examined how students use recommendations and explanations and assessed the impact of these technologies on various aspects of the educational process.

2 Related Work

Over the last decade, explainable recommendations emerged as a major trend in the area of recommender systems [15]. Tintarev and Masthoff [19] define ex-

planations as "*item descriptions that help the user to understand the qualities of the item well enough to decide whether it is relevant to them or not*". Research has shown that presenting explanations to users can increase the persuasiveness of the recommended items as well as users' trust and satisfaction with the recommender system [10,14]. While being popular in domains where recommendations are based on interests and taste such as movies or songs [15], explainable recommendations remain understudied in the online learning domain.

One of the challenges of transferring explanation approaches accumulated in the area of traditional recommender systems [10,15] to the area of educational recommendations is the different nature of these recommendations, which are typically based on the learner's knowledge state rather than user interests or taste. Given this, explanations of educational recommendations have to be designed afresh rather than re-used from the taste-based domains. At the same time, the "knowledge-based" nature of educational recommendations offers a new opportunity. *Learner models* that are used for generating educational recommendations carry a much higher explanation potential than *user profiles* applied by traditional recommender systems [9]. While user profiles are notoriously hard for users to understand or control [2], the long history of AI-Ed research on *open learner models* (OLM) demonstrates that learner models could serve as a means to understand and control the behavior of adaptive educational systems [5,4]. Moreover, OLMs have shown their effectiveness in facilitating navigation and supporting metacognitive processes of planning, monitoring, and reflection [5].

We believe that the large body of OLM research offers an excellent starting ground for the design of explainable educational recommendations. Moreover, as argued by [6,17] insights learned from OLM research can be used for a broader goal to improve interpretability in AI-driven educational systems. Yet to complete the task, a layer of visual or verbal explanations should be built on the top of OLM to produce explainable recommendations. The work in this direction is at the very beginning and we could cite only a few motivating cases. Putnam and Conati [16] investigated the value of having explanations for automatically generated hints in an Intelligent Tutoring System (ITS). Barria-Pineda et al. [3] and Abdi et al. [1] were the first to explore the effects of using an OLM as the basis of justifying learning content recommendations. Most recently Zhou et al.[21] found that explaining the decisions of an ITS to students could improve the student-system interaction in terms of their engagement and autonomy.

3 Java Personalized Programming Practice System

To explore the value of explainable recommendations in online learning, we implemented content recommendations for Java Personalized Programming Practice System (JP^3). JP^3 is an online personalized system offering students in introductory Java programming courses to practice their skills using several types of interactive learning content. The system is designed as a non-mandatory practice and self-assessment tool that each student could use for individual needs. In this section, we present the design of content recommendations in JP^3 and the mechanism for generating recommendations, and their explanations.

3.1 Explainable Recommendations in JP^3

JP^3 provides access to three types of Java learning content: worked examples, challenges - faded examples where students have to complete missing parts of the code [12], and short coding problems [22]. The learning content in JP^3 is grouped into topics (e.g., variables, if-else, etc.) that follow the chronological order of the course. To start practicing, the student has to select one of the topics as the current goal. After opening a topic, the student can see the list of available practice content for this topic along with *personalized guidance* for choosing the most appropriate activity (Fig. 1). The personalized guidance is based on a concept-level overlay model of student's Java knowledge. The model is built by observing student behavior in the system and represents the probability of students knowing each Java concept. To make this learner model "open" to the student, it is visualized as a bar chart on the bottom part of the activity selection interface. Each bar depicts one concept. The height and the color of the bar indicate the estimated mastery of this concept: the higher and the greener is the bar, the higher is the mastery estimation; whilst the lower (below the origin) and the redder is the bar, the lowest is the estimated mastery. Concepts are arranged along the x-axis following the order of topics where they are introduced. Concepts introduced in the current topic are emphasized by a dashed rectangle.

The personalized guidance is offered by recommendations and explanations. Recommendations suggest the three most relevant learning activities in the topic

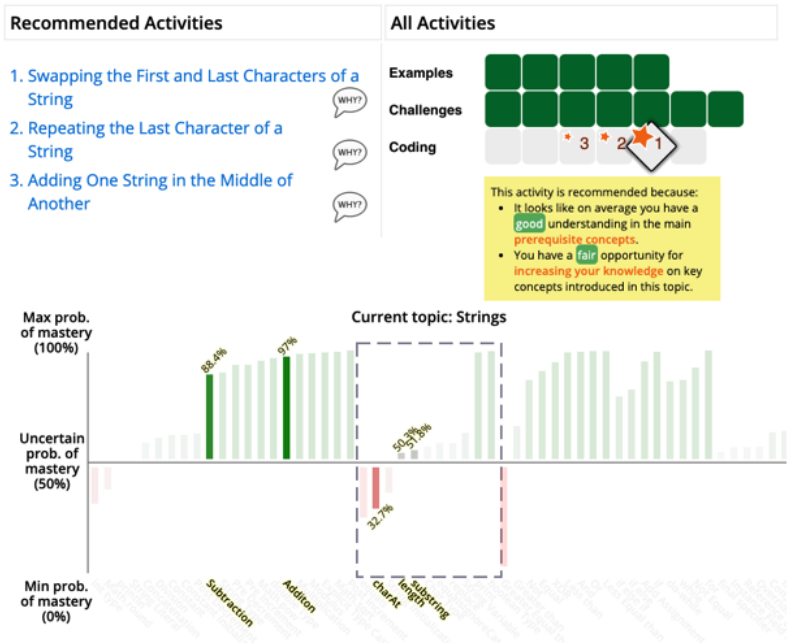

Fig. 1: Visualization for *Strings* topic in JP^3 with recommended content shown as a list (left) and with stars. The learner mouses over the top recommendation and JP^3 (1) highlights in the OLM the estimated knowledge of concepts linked to this item (see bar chart) and (2) shows a verbal explanation (see yellow box).

given the current state of the learner model. Recommendations are provided as a ranked list on the left and also in navigation support form as stars of different sizes placed over the recommended activities. As explained in the next section, the recommendation approach favors activities that combine sufficient levels of prerequisite knowledge (concepts to be learned in earlier topics) with a good opportunity to master target knowledge (concepts introduced in the topic).

Explanations are offered in visual and verbal form. Following Tintarev and Masthoff's guidelines for explanations of recommendations [18], explanations attempt to increase the *transparency* of the recommendation process. Given the nature of JP^3 recommendations, both types of explanations focus on highlighting the balance between prerequisite and target knowledge associated with an activity. ***Visual explanations*** are provided when student mouses over an activity cell by highlighting names and bars of concepts that can be practiced through this activity. The visualization stresses whether the student has sufficient prerequisite knowledge to attempt the activity and how much this activity could improve the target knowledge. As we see in Fig. 1, the top recommended problem (large star) pointed by the mouse involves five concepts. Two prerequisite concepts (*addition* and *subtraction*) to be learned in the earlier topics are already mastered making the student ready to attempt the problem. At the same time, three target concepts (*substring*, *charAt*, *length*) are not yet well-learned making the problem a good opportunity to improve this knowledge.

Verbal explanations attempt to convey the same idea of readiness and relevance through natural language (yellow box on the right of Fig. 1). A typical explanation was composed of two sentences where the first explains the system's assessment of the prerequisite knowledge for the examined activity while the second assesses its learning opportunity. To stress how positive is each part of the assessment, the focus keyword of each part (e.g., good, fair) is marked by the green color of different intensity. The darker the green is, the more positive is the assessment. Our original intention was to make verbal explanations accessible along with visual explanations on mouse-over, however, we were concerned that it will make it hard to examine the usage of each type separately. To support our study needs, we implemented two ways to access verbal explanations: *explanations on mouse-over* (*expOnMouseover*) where verbal explanations were presented along with visual by mousing-over the recommended activity in the grid and *explanations on-click* (*expOnClick*) where verbal explanations were accessed by clicking on *why* icon next to the recommended activity in the list.

3.2 The Implementation of Explainable Recommendations

Knowledge Modeling: JP^3 uses an ontology of Java concepts as the core for its knowledge representation. Each learning content item in JP^3 is linked to the ontology concepts automatically using a concept parser [13]. A Bayesian Network [11] is used to maintain a probabilistic overlay student model for these concepts. The network was initially trained with data collected through JP^3 in earlier classroom studies. The knowledge estimates are seeded based on students' performance in the pretest. Every time a student attempted a challenge or a

problem while practicing with JP^3, the system updated the probability estimates related to the concepts linked to that activity. Simply, the probability estimates increase when the student's answer was correct and decreased otherwise.

Recommendation Approach: JP^3 content recommendation algorithm maximizes the balance between the opportunity to improve knowledge of target concepts and the necessity of sufficient knowledge on prerequisite concepts that are needed to solve the activity correctly. A concept associated with an activity is labeled as a prerequisite if it is expected to be mastered in the chronologically earlier topics and as a target if it is the topic where the concept is first introduced. The recommendation algorithm uses the concept-level knowledge estimates taken from the student model and generates content recommendations according to the following rules: (1) only non-completed activities are recommended; (2) examples have recommendation priority when they introduce a new concept that has not been practiced before; (3) for challenges and problems, a recommendation score is calculated using the Equation 1,

$$\text{rec score}_{ij} = \frac{1}{NW}\left(\sum_{p} w_p * \theta_{pj} + \sum_{t} w_t * (1 - \theta_{tj})\right) \quad (1)$$

where p represents the prerequisite concepts and t represents the target concepts associated with activity i. θ_{pj} and θ_{tj} are the knowledge estimates of student j for both types of concepts. w denotes the topic-level importance of the concepts (either p or t) calculated by the *tf-idf* approach (i.e., the more unique a concept in a topic, the higher its importance) and W is the sum of the weights for the associated concepts (both prerequisite and target ones). Finally, N denotes the total number of concepts associated with activity i. Learning activities are sorted based on these scores and top-3 items are recommended to the learner.

Verbal Explanations: To generate verbal explanations for recommendations, we calculate the average proficiency for both the top three prerequisite and target concepts ($\bar{\theta_p}$ and $\bar{\theta_t}$). Based on these proficiency estimations, we generate short paragraphs for each part of the verbal explanations. Table 1 presents samples of verbal explanations for several thresholds. The thresholds and wording were selected to offer a qualitative explanation of numerical values and were not used to drive the recommendation process. A recommended example was justified by stating that *"it presents concept(s) that are new to you (e.g. concept_name)"*.

Table 1: Rules for generating explanations for educational recommendations

Verbal explanation template for prerequisite concepts	$\bar{\theta_p} \geq .6$	$\bar{\theta_p} \geq .75$	$\bar{\theta_p} \geq .95$
It looks like on average, you have a ... understanding in the main prerequisite concepts.	good	proficient	excellent
Verbal explanation template for target concepts	$\bar{\theta_t} \leq .6$	$\bar{\theta_t} \leq .4$	$\bar{\theta_t} \leq .2$
You have a ... opportunity for increasing your knowledge on key concepts introduced in this topic.	fair	good	excellent

4 Study

To assess the impact of explainable educational recommendations, we performed a semester-long classroom study. The study was conducted with 86 undergradu-

ate students taking a Java Programming course at a European university. After taking an online pretest designed to assess their prior knowledge, each student was given access to the JP^3.The use of JP^3 was non-mandatory, so there was no penalty to those who did not use the system. In contrast, to encourage JP^3 use, 10% extra credit was added to their final course grade if they viewed at least 80% examples, solved at least 70% of challenges, and 60% of coding problems. At the end of the term, an online post-test (isomorphic to the pretest) was taken. Pretest scores show that a high proportion of students had an medium level of proficiency in Java, given that the median grade in the pretest was 5 out of 10. Students were randomly assigned to one of two explanatory treatments described in section 3.1: *expOnMouseover* ($n = 45$) or *expOnClick* ($n = 41$). Student actions in JP^3 were logged. The logs included content openings, problem solving attempts, mouse-overs (with duration) on recommended and non-recommended activities, and access to verbal explanations on-click. As the summary of activities shows (Table 2), on average, students opened and attempted a large fraction of available learning activities.

4.1 Value of Recommendations

To assess the impact of recommendations on engagement, we contrasted student engagement with recommended and non-recommended activities on two levels: (1) attempting the learning activity once it is opened (*conversion rate*) and (2) keep working on the activity after the first attempt until solving it correctly (*persistence rate*). To reliably assess the impact of recommendations, we considered (1) topics with more than 6 activities and (2) students' actions until more than 3 activities left to be completed in a topic to eliminate cases where students had *no other choice* than to select a recommended item.

We fit a series of mixed-linear models to predict conversion and persistence rates by using the pretest score, the type of learning activity (recommended or non-recommended) and the interaction (*int*) between these two variables as independent variables. We add student ids as a random effect. In this and following analyses we use common notation to report significance levels of variables within models: $p < .05 \rightarrow$*, $p < .01 \rightarrow$**, $p < .001 \rightarrow$***. We found a significant model for the conversion rate in coding problems ($p < .001$), which revealed that the probability of attempting a problem once it is opened depends on the starting knowledge of the student measured by the pre-test (β_{int}=-.025*). We also found significant main effects ($\beta_{pretest}$=.04***,β_{rec}=.13*). More exactly, low-pretest learners exhibited significantly higher conversion rates on recommended items while high-pretest students demonstrated higher chances of attempting

Table 2: Summary of students' activity in JP^3 (Mean(SD))

	Number of mouse-overs	Mouse-overs' duration (sec)	Explanations' clicks	Activities opened	Activities attempted	Activities solved
Coding (n=46)	37.3(27.3)	1.42(.31)	2.98(2.19)	19.5(13.3)	13.7(10.3)	8.1(7.7)
Challenges (n=76)	53.5(37.1)	1.34(.36)	3.18(1.98)	33.2(12.7)	30(11.9)	28.4(11.4)
Examples (n=55)	39.3(30.1)	1.36(.31)	4.86(3.05)	25.4(11.1)	22.3(8.7)	20.5(8.12)

non-recommended ones. On one hand, it indicates that students with lower domain knowledge relied considerably on system recommendations when selecting content to practice. On the other hand, it hints that JP^3 underestimated the knowledge of high-pretest students since a considerable proportion of their Java learning happened before the course and was not accurately modeled. No significant model was found for conversion on challenges and examples. Conversion rates were uniformly very high for all students indicating that they were less picky when selecting low-effort activities.

We found similar results when checking the impact of recommendations on the *success rate* of attempted coding problems (i.e., where conversion was reported). We fit the same model as used for conversion rates, but including success rate as the dependent variable ($p < .05$). We found a significant interaction between pretest and the presence of recommendation (β=-.016*). We also found significant main effects ($\beta_{pretest}$ =.018*,β_{rec}=.13**). The data shows that learners with lower pretest exhibited much higher success rates when working on recommended coding problems while for students with high pretest scores there was almost no difference. A similar model for challenges ($p < .001$) failed to reveal an impact of recommendations. The only factor that affected the success rate of students was the pretest (β=.03***) – high-pretest students exhibited higher success rates with challenges.

For *persistence rate*, only pretest acted as a significant predictor ($\beta = 0.08$**) for the model ($p < .001$) – the higher the pretest, the more persistent students are in coding problems, regardless if they were recommended or not. No significant model was found for explaining persistence on challenges and examples.

4.2 Value of Explanations

The first step of assessing the value of explanations is to examine whether they were used or not. As Table 2 shows, on-click explanations were requested for about 16% of attempted activities, which is a considerable usage. To assess whether students were "processing" mouse-over explanations for recommended content as well, we contrasted the duration of mouse-overs on recommended and non-recommended activities. For this analysis, we excluded "short" mouse-overs ($<$ 1 sec.), which were likely generated "on passing" and provided too little time to pay attention to either visual or verbal explanations. We found that students took on average more time on mousing over recommended items than non-recommended ones, which suggests that they paid attention to explanations of recommended activities ($p < .001$). Moreover, we found that this difference was lower for *expOnClick* who can only check visual explanations on mouse-over ($M_{diff} = .051$) and higher for *expOnMouseover* ($M_{diff} = .181$), who receive both explanation types on mouse-over. A greater additional time spent by *expOnMouseover* students ($p = .033$) hints that they paid attention to both verbal and visual explanations.

Next, we checked whether inspection of explanations (measured as the mean duration of mouse-overs on learning content) was associated with adoption of either recommended or non-recommended activities. We fit two multiple regression models: (a) *predictors*: pretest score and mouse-overs' duration on recommended

items, *outcome*: percentage of the total items accessed by the learner which were recommended ones; and (b) *predictors*: pretest score and mouse-overs' duration on non-recommended items, *outcome*: percentage of the total items accessed by the learner which were non-recommended ones. For (a), we found that both pretest score (β=-2.01*) and mean mouse-over duration on recommended activities (β=31.8*) were significant predictors of commitment with these type of items ($F(2, 58) = 10.93,\ adj.R^2 = .25,\ p < .001$). Similarly, for (b) we found that pretest score (β =2.09*) and average mouse-over duration on non-recommended activities (β=-37.19*) were correlated with the adoption of non-recommended content for practicing ($F(2, 58) = 6.78,\ adj.R^2 = .16,\ p = .002$). The data shows that the ability to examine explanations of recommended problems increases student's motivation to attempt these problems. On the other hand, the ability to inspect explanations of non-recommended problems decreases learner's chances to attempt those items. These results also reiterate that low-pretest students chose to work more on recommended content, in contrast to high-pretest students who decided to perform a self-guided exploration of the content instead.We did not find any influence of mouse-over duration on challenges or examples.

To assess if the difference in how to access explanations in the *expOnClick* group influenced students' engagement with the learning activities, we analyzed if the number of clicks performed by a learner correlated with students' engagement in working on JP^3. We divided students into low and high "explanation requesters", according to the median of explanations' clicks (considering the three types of learning content). We found that these two different groups significantly differ on the number of attempts on coding problems ($p < .01$), where high explanation requesters exhibited a much higher average number of attempts on coding problems ($Med = 33$) than low explanation requesters ($Med = 20$).

4.3 Students' Work in JP^3 and Course Performance

To assess the educational value of practicing with JP^3, we examined the correlation between the work of students within JP^3 and their performance in the course throughout the term. In this study, we had access to two classroom test scores that evaluated students' performance (*Test1* in the first half and *Test2* in the second half of the course), and the post-test scores that were not graded. To prepare data, we counted the number of successful and failed attempts on learning activities and calculated the average success rate per week. To account for the regularity of practice performance, we calculated the skewness of the distribution of weekly success/failure attempts. We repeated the same calculations for the number of sessions in JP^3 per week. Skewness can tell us if the work/performance of students was concentrated at the start (positive skewness) or the end (negative skewness) of the course. In our multiple regression models, we added pretest scores to control for the prior knowledge. We performed a step-wise feature selection process for each prediction model. While we considered these metrics for all the types of learning content, only performance on coding problems added predictive power to the models, while variables related to work on challenges and examples did not. It is consistent with the fact that,

given the incentives for getting extra-credit and the lower efforts associated with completing challenges and examples, all learners achieved a uniformly high completion level at the end of the term.

We first predicted scores in *Test1* and found a significant overall model ($F(3, 42) = 6.328,\ adj.R^2 = .26,\ p < .001$) where average success rate was the only significant predictor (β=.44***). Second, we fitted a model to predict *Test2* scores and results indicated a significant model ($F(3, 41) = 5.616,\ adj.R^2 = .24,\ p = .003$) with only pretest-score as a significant predictor (β =.44) among other predictors. Finally, we predicted post-test scores ($F(3, 52) = 8.018,\ adj.R^2 = .28,\ p < .001$) and found that pretest scores (β =.36*) , skewness of incorrect coding attempts (β =-.58*) and average success rate (β=.2*) were significant predictors of post-test scores.

5 Summary and Discussion

In this paper, we presented the design of an online programming practice system JP^3 augmented with explainable recommendations of learning content to practice. The recommendations were generated by optimizing the balance between the current level of prerequisite knowledge and the opportunity of practicing new concepts. As input, the explainable educational recommender module uses the state of student knowledge of Java concepts estimated by a student model based on the observable student performance. Explanations were generated in two different forms (see Fig. 1): (1) visual explanation through a concept-level OLM which showed the estimations of the learner's knowledge on the concepts associated with each learning item (2) verbal explanation describing the balance between prerequisites and potential for new knowledge acquisition (only for recommended content). We also presented a semester-long study where JP^3 was used as a learning support tool for an intermediate programming class. Our goal was to investigate how recommendations and explanations affect different aspects of student work with the learning content in a free practice mode.

Our data showed that students invested their time to access and inspect the explanations of the recommendations. The average time spent on mousing over activities was significantly higher for recommended activities than for non-recommended ones. Moreover, students in the *expOnMouseover* group exhibited longer mouse-overs on recommended items than *expOnClick* learners, since the first group was able to observe both visual and verbal explanations when mousing over recommended activities and needed more time to process it.

By examining the effect of recommendations on student behavior, we observed that recommendations affected student selection and engagement with high-effort (coding problems) and low-effort activities (challenges and examples) differently. While for low-effort activities, learners' behavior was not influenced significantly by the recommendations, the *conversion* for coding problems (making at least one attempt on an opened problem) was significantly and positively influenced by the presence of a recommendation. This effect seems to be mediated by the students' prior knowledge. As we observed, low-pretest students, exhibited a higher level of trust in the recommendations and the willingness

to work with recommended problems. Moreover, we found that students with lower prior knowledge achieved a higher success rate on solving recommended coding problems than on non-recommended ones. These results indicate a better match of recommended problems to student knowledge. In contrast, students with a higher level of starting knowledge exhibited higher conversion rates on non-recommended problems. This situation might be explained by the fact that the learner model was initialized using results of a relatively small 10-problem pretest that underestimated the knowledge of students with a high level of proficiency in Java. Given the transparent OLM, these students might have noticed that the recommendations were generated using an incomplete model and preferred to select the content to practice themselves. In this sense, the explanations still achieved their goal to help students in selecting the right content to practice, in this case, revealing that recommended content is not adequate and helping them to make their own choice with OLM-based visualization.

On top of the effect of recommendations, we also observed that inspection of the explanations affected student engagement with learning content. The more time students spent while mousing over the recommended activities, the more they were willing to open them. In contrast, the more time student spent on inspecting visual explanations for non-recommended activities, the less they were inclined to open them. In this aspect, student behavior was also influenced by their starting level of knowledge. The students with high pre-test scores exhibited lower ratios of engagement with the recommended activities. This behavior is likely to have the same roots as discussed above for conversion data.

By putting together all these results, we can conclude that explainable recommendations can support students working with a programming practice system, most noticeably affecting the learners who are novices in programming and have the highest need for help in choosing activities to practice. The presence of recommendations and explanations could increase student engagement with knowledge-relevant learning content leading to a higher success rate and an increased opportunity to learn. In turn, it was exactly the success rate in problem-solving within JP^3 that impacted students' knowledge progress throughout the term, as it was positively correlated with student performance on intermediate evaluations and also on the post-test at the end of the class. Altogether, our explainable recommendation approach has the potential to positively impact students activity within JP^3 by pushing them to practice more, focusing on the most appropriate high-effort learning materials, and at the same time providing them with the opportunity for reflecting on the appropriateness of the content for supporting each step of their learning.

However, this study has several limitations. In particular, we were not able to reliably track student work with visual and verbal explanations using logs, as we use only mouse-over time as a proxy of attention. In the future, we need to better assess the impact of explanations by running studies where visual attention of students can be captured (e.g. eye-tracking controlled study). Also, more efforts are needed to define strategies that could make recommendations more relevant and useful for learners with higher initial levels of knowledge.

References

1. Abdi, S., Khosravi, H., Sadiq, S., Gasevic, D.: Complementing educational recommender systems with open learner models. In: Proceedings of the Tenth International Conference on Learning Analytics Knowledge. p. 360–365. LAK '20, ACM, New York, NY, USA (2020)
2. Ahn, J.W., Brusilovsky, P., Grady, J., He, D., Syn, S.Y.: Open user profiles for adaptive news systems: help or harm? In: the 16th international conference on World Wide Web, WWW '07. pp. 11–20. ACM (2007)
3. Barria Pineda, J., Brusilovsky, P.: Making educational recommendations transparent through a fine-grained open learner model. In: Workshop on Intelligent User Interfaces for Algorithmic Transparency in Emerging Technologies at the 24th ACM Conference on Intelligent User Interfaces, IUI 2019. vol. 2327. CEUR (2019)
4. Bull, S.: There are open learner models about! IEEE Transactions on Learning Technologies **13**(2), 425 – 448 (2020)
5. Bull, S., Kay, J.: SMILI: A Framework for Interfaces to Learning Data in Open Learner Models, Learning Analytics and Related Fields. International Journal of Artificial Intelligence in Education **26**(1), 293–331 (2016)
6. Conati, C., Porayska-Pomsta, K., Mavrikis, M.: AI in education needs interpretable machine learning: Lessons from open learner modelling. arXiv preprint arXiv:1807.00154 (2018)
7. Drachsler, H., Verbert, K., Santos, O., Manouselis, N.: Panorama of recommender systems to support learning. In: Recommender Systems Handbook, pp. 421–451. Springer, Berlin (2015)
8. Erdt, M., Fernández, A., Rensing, C.: Evaluating Recommender Systems for Technology Enhanced Learning: A Quantitative Survey. IEEE Transactions on Learning Technologies **8**(4), 326–344 (2015)
9. Gauch, S., Speretta, M., Chandramouli, A., Micarelli, A.: User Profiles for Personalized Information Access, Lecture Notes in Computer Science, vol. 4321, pp. 54–89. Springer-Verlag, Berlin Heidelberg New York (2007)
10. Gedikli, F., Jannach, D., Ge, M.: How should I explain? A comparison of different explanation types for recommender systems. International Journal of Human Computer Studies **72**(4), 367–382 (2014)
11. Hosseini, R.: Program Construction Examples in Computer Science Education: From Static Text to Adaptive and Engaging Learning Technology. Doctoral dissertation (2018)
12. Hosseini, R., Akhuseyinoglu, K., Petersen, A., Schunn, C.D., Brusilovsky, P.: Pcex: Interactive program construction examples for learning programming. In: Proceedings of the 18th Koli Calling International Conference on Computing Education Research. pp. 5:1–5:9. ACM (2018)
13. Hosseini, R., Brusilovsky, P.: Javaparser: A fine-grain concept indexing tool for java problems. In: The First Workshop on AI-supported Education for Computer Science (AIEDCS 2013). pp. 60–63 (2013)
14. Kulesza, T., Stumpf, S., Burnett, M., Yang, S., Kwan, I., Wong, W.K.: Too much, too little, or just right? Ways explanations impact end users' mental models. Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC pp. 3–10 (2013)
15. Nunes, I., Jannach, D.: A systematic review and taxonomy of explanations in decision support and recommender systems. User Modeling and User-Adapted Interaction **27**(3-5), 393–444 (2017)

16. Putnam, V., Conati, C.: Exploring the need for explainable artificial intelligence (XAI) in intelligent tutoring systems (ITS). In: Joint Proceedings of the ACM IUI 2019 Workshops co-located with the 24th ACM Conference on Intelligent User Interfaces (ACM IUI 2019), Los Angeles, USA, March 20, 2019 (2019)
17. Rosé, C.P., McLaughlin, E.A., Liu, R., Koedinger, K.R.: Explanatory learner models: Why machine learning (alone) is not the answer. British Journal of Educational Technology **50**(6), 2943–2958 (2019)
18. Tintarev, N., Masthoff, J.: Designing and Evaluating Explanations for Recommender Systems. In: Recommender Systems Handbook, Second Edition, vol. 2, pp. 479–510 (2010)
19. Tintarev, N., Masthoff, J.: Evaluating the effectiveness of explanations for recommender systems: Methodological issues and empirical studies on the impact of personalization. User Modeling and User-Adapted Interaction **22**(4-5), 399–439 (2012)
20. Verbert, K., Manouselis, N., Ochoa, X., Wolpers, M., Drachsler, H., Bosnic, I., Duval, E.: Context-aware recommender systems for learning: A survey and future challenges. IEEE Transactions on Learning Technologies **5**(4), 318–335 (2012)
21. Zhou, G., Yang, X., Azizsoltani, H., Barnes, T., Chi, M.: Improving student-system interaction through data-driven explanations of hierarchical reinforcement learning induced pedagogical policies. In: Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization. p. 284–292. UMAP '20, ACM, New York, NY, USA (2020)
22. Zingaro, D., Cherenkova, Y., Karpova, O., Petersen, A.: Facilitating code-writing in pi classes. In: Proceeding of the 44th ACM Technical Symposium on Computer Science Education. p. 585–590. SIGCSE '13, ACM, New York, NY, USA (2013)