An Intelligent Pedagogical Agent for In-The-Wild Interaction in an Open-Ended Learning Environment for Computational Thinking

Rohit Murali, Sébastien Lallé, Cristina Conati

An Intelligent Pedagogical Agent for In-The-Wild Interaction in an Open-Ended Learning Environment for Computational Thinking

Rohit Murali
muralirohit@gmail.com
The University of British Columbia
Vancouver, BC, Canada

Sébastien Lallé
sebastien.lalle@lip6.fr
LIP6, CNRS, Sorbonne Université
Paris, France

Cristina Conati
conati@cs.ubc.ca
The University of British Columbia
Vancouver, BC, Canada

ABSTRACT

Adaptive support can help learners in *Open-Ended Learning Environments* (OELEs), where the free-form nature of the interaction can be confusing to students. In this paper, we design and evaluate an *Intelligent Pedagogical Agent* (IPA) for an OELE designed to foster *Computational Thinking* (CT). Specifically, we design help interventions for an *in-the-wild* scenario where students interact with the OELE in an unmonitored, self-directed manner. We build a student model by extracting meaningful student behaviors on real-world interaction data obtained during interaction in online classrooms and including expert insights. We show that these student models perform better than a baseline and have the potential for adaptive support in self-directed interaction with the OELE. We design an IPA with the help of teachers, leveraging the student behaviors extracted from data. Lastly, we get insights into the value of these help interventions by empirically evaluating the IPA in a formal user study.

CCS CONCEPTS

• **Human-centered computing** → **User models**; *User studies*; **Field studies**.

KEYWORDS

Intelligent pedagogical agent, Adaptive hints, Computational thinking, In-the-wild user study

ACM Reference Format:
Rohit Murali, Sébastien Lallé, and Cristina Conati. 2024. An Intelligent Pedagogical Agent for In-The-Wild Interaction in an Open-Ended Learning Environment for Computational Thinking. In *Proceedings of ACM Intelligent Virtual Agents Conference (IVA'24).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3652988.3673948

1 INTRODUCTION

OELEs are interactive systems designed to allow students to engage in learning with minimal constraints [13]. Previous work has shown the value of OELEs in fostering students' learning (eg. [12, 13]) by engaging learners through their perspectives [14] and promoting reflection. Success in OELEs, however, is dependent on the learner's voluntary cognitive engagement [29]. For some learners, the exploratory nature of OELEs can often be confusing and it can be difficult to quantify progress, leading learners to detrimental interaction with the environment such as incorrectly interpreting visual cues or making imprecise inferences from the learning content [11, 23]. Research has shown that adaptive support can alleviate these detriments in OELEs, by monitoring and responding to the learners' difficulties in real-time, e.g., [2, 16, 25, 26, 31].

In this paper, we build and evaluate an intelligent pedagogical agent (IPA) for adaptive support in a specific OELE designed to foster computational thinking (CT) skills, Unity-CT, developed by UME Academy. CT is defined as the ability to express problems and their solutions computationally, for which there have been advocated efforts [1, 9, 34] by K-12 educators to focus on, akin to reading, writing, and arithmetic. Unity-CT was originally designed for in-person interaction with UME Academy's teachers. From 2020-2022 onward, Unity-CT moved online, but still involved interaction with teachers. Since 2023, UME Academy offers self-directed online lessons where students interact with Unity-CT without the presence of a teacher. Here, we build adaptive support, for the first time for fully unmonitored interaction in Unity-CT without the presence of researchers or teachers from UME Academy. Adaptive support can help provide personalized assistance to struggling students and is even more important for students interacting with OELEs *in-the-wild*. Specifically, we use *in-the-wild* to refer to fully *unmonitored*, self-directed interaction with an OELE.

To build the student model driving our IPA, we apply the existing data-driven Framework for User Modelling and Adaptation (FUMA) [15] to Unity-CT's interaction data. FUMA has shown success in user modeling for open-ended learning activities like Massive Open Online Courses [22] (MOOCs), and interactive learning simulations, [8, 16] and even in Unity-CT [20]. Lallé et al. [20] previously leveraged FUMA to build interpretable data-driven student models in Unity-CT for adaptive support using real-world data collected in 2019 during in-person lessons with Unity-CT. Unfortunately, since in-person lessons were halted in 2020 due to the SARS-CoV-2 pandemic, and the timeline of student interaction was altered when Unity-CT moved online, we do not know to what extent their student model is still applicable to the current lessons in Unity-CT. Here, we leverage FUMA similarly, but build our student model from scratch using data from online interaction with Unity-CT. In this process, we refine the data-driven student model from FUMA with expert insights from teachers at UME Academy, allowing us to build a final data-informed student model for adaptive support. We delineate here between data-driven (let data speak for itself) and data-informed (let us figure out what data tell us) [33] in the

IVA'24, September 2024, Glasgow, Scotland

ACM ISBN 123-4567-24-567/08/06
https://doi.org/10.1145/3652988.3673948

sense that the data-driven models rely on data alone whereas data-informed models build on the purely data-driven student model by considering it along with other factors (e.g., teachers' insights).

In previous work [21, 35, 36], we have already designed an IPA for Unity-CT, but for two very specific, teacher-informed errors. These studies had positive results in terms of usability but no impact on learning performance. Here, we develop an IPA by considering for adaptation data-driven behaviors that are more nuanced as compared to predefined behaviors, and could better help struggling students. Thus, we leverage the data-informed student model to deliver real-time adaptive help interventions to students. Finally, we evaluate the impact that our IPA has on student learning performance by conducting a real-world user study in online self-directed lessons in Unity-CT. To do this, we conduct a formal evaluation between groups of students who receive adaptive help interventions and those who do not. Our results show that the IPA improves student learning performance. Moreover, we analyze the impact that specific help interventions had on learning performance.

Specifically, our work provides two novel contributions to research in adaptive support in OELEs. First, to the best of our knowledge, there is only limited work for IPAs in OELEs that foster CT skills [2, 21, 36], where adaptive support was provided based on obvious errors and behaviors that are solely pre-defined by experts with extensive domain knowledge but this process is laborious, and may not capture subtle student behaviors that may go unnoticed to experts. In our work, we design and evaluate data-informed adaptive support based on student behaviors extracted from real-world data combined with expert insights from teachers. Second, most research to develop IPAs in OELEs in other domains examine data-informed support [16, 25] but these were not for in-the-wild interaction. In most research in adaptive support for OELEs, while interaction may be self-directed, it is usually monitored by teachers or researchers in laboratory or classroom settings [2, 3, 16, 25]. This is reasonably done to limit the number of extraneous factors that often occur in the real-world for model development. However, it is important to evaluate IPAs in fully unmonitored environments in order for robust adaptive support. To the best of our knowledge, we are the first to build and evaluate an IPA for in-the-wild interaction with OELEs. In this regard, we provide additional empirical evidence to research in OELEs to foster CT skills that adaptive scaffolding can improve student learning performance.

The rest of the paper is organized as follows. In section 2 we discuss related work. In section 3, we describe Unity-CT and the datasets used in this paper. In section 4, we describe FUMA and how we process data from Unity-CT. In section 5, we leverage FUMA to build a student model for adpative support in Unity-CT. In section 6, we discuss the design and implementation of the IPA. In section 7, we evaluate the IPA through a formal user study. Finally, we present our conclusions and future work in section 8.

2 RELATED WORK

IPAs have been previously shown to have great potential for improving an OELE's ability to provide adaptive support to students [24], albeit to the best of our knowledge, only three studies have evaluated IPAs in CT OELEs [2, 36]. Namely, Basu et al. [2] designed an IPA in the CTSiM OELE to provide adaptive support to middle school students in interactive simulations for model building activities (e.g., modeling a car's speed based on its mass and engine force). The adaptive hints delivered by the IPA, which are based on desired students' behaviors and solutions in the simulation, were found to have a positive effect on their learning performance. An important difference with our work is that the behaviors for the adaptations are pre-defined upfront by experts, which is not possible in free-form Game Design (GD) as in Unity-CT where students design games as they like, due to the extremely large solutions and behaviors space. Thus, we contribute to [2] by showing that data-driven adaptive support delivered by an IPA can be valuable to teach CT in another activity, free-form GD, which is arguably more engaging to a very young audience than model building, albeit much more unconstrained. In [21, 36], they designed an IPA for Unity-CT, which provided adaptive hints on two specific erroneous behaviors predefined upfront by teachers. The IPA hints were found to significantly reduce error rates, but only for one of the two target behaviors, and with no impact on the students' grades at the end of the lesson. Here, we modify this IPA by leveraging a data-driven approach to consider for adaptation more and richer suboptimal behaviors than the ones predefined by the teachers, which successfully improved students' grades in the end. This provides new insights into the applicability and value of data-driven IPA hints in CT OELEs.

Other works have studied the value of data-informed support in OELEs for learning domains others than free-form GD and CT, with positive results in terms of learning outcomes and/or engagement. In particular, Munshi et al. [25] designed a conversational IPA that provides adaptive feedback while middle school students create causal maps about science topics in Betty's Brain. The adaptation is driven by suboptimal behavioral patterns both mined from data and predefined by experts. The adaptation mechanism, however, requires to continuously assess the correctness of ongoing students' causal maps by comparing them to experts' ones, which would not be possible with Unity-CT due to the virtually unlimited space of correct solutions and no clear way to assess correctness until the student is done. Kardan & Conati [16] designed adaptive support driven by suboptimal behaviors discovered from data in a simulation to learn about constraint satisfaction algorithms. Unlike Unity-CT, they targeted university students, whereas we focus on elementary and middle school students in a different learning domain (free-form GD to foster CT). Importantly, the aforementioned works were evaluated in controlled user studies, either in laboratories [16] or in-classroom [2, 21, 25, 36], whereas we focus on self-directed, unmonitored usages entirely in the wild, as stated in the introduction. Thus, our work also provides novel insights on the feasibility and value of an IPA used in such *in-the-wild* context.

3 UNITY-CT

Unity-CT (*ume.games*), developed by UME Academy, a Vancouver-based educational company, empowers students to engage in interactive game construction while fostering the development of CT skills. Students learn how to design a platformer game by interacting with the open environment. Figure 1 provides a visual snapshot of the environment embedded in the game engine, Unity.

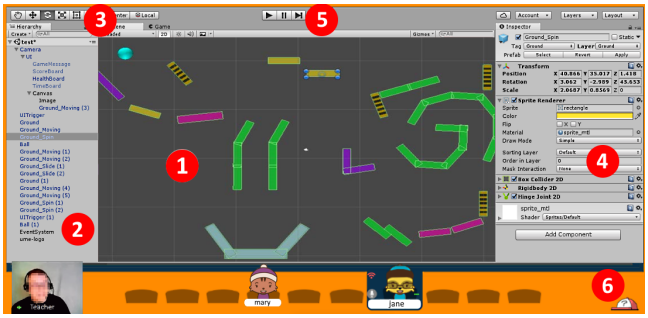


Figure 1: Unity-CT Environment

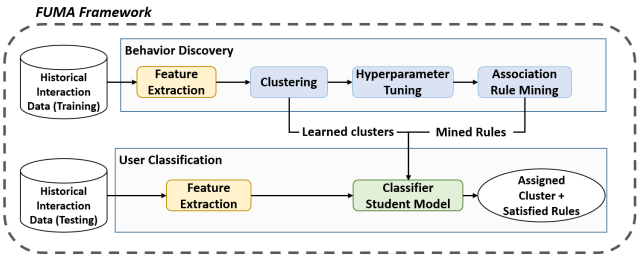


Figure 2: FUMA Framework

Within the Unity-CT environment, students are immersed in a scene view (fig. 1 - area 1). Here, students can interact with the scene and game objects organized in a hierarchical structure (fig. 1 - area 2) using various manipulators (fig. 1 - area 3). Students can interact with the scene using 4 actions - Pan, Zoom, Execute, and Save. They can also interact with game objects using 4 manipulators - Select, Move, Rotate, and Scale. Furthermore, students have the ability to directly modify object properties through the inspector panel (fig. 1 - area 4), affording them fine-grained control over their game elements - although this is not explicitly taught during the first lesson. All these actions can be done when the student is in Edit Mode. Furthermore, students can test their creations by switching to Play mode (see play button in fig. 1 - area 5). Students can switch back and forth between Play/Edit mode by clicking on the play button. In online classes conducted by UME Academy, teachers can navigate through each student's screen using a dedicated dashboard and offer help to students who may need it. The online version of Unity-CT had standardized the timeline for both the online classes from 2020-2022 and the self-directed lessons from 2023 onward. This timeline consists of a 15-minute tutorial[1] followed by up to an hour of interaction with the environment.

UME Academy's educational curriculum integrates CT design practices in alignment with the well-established Brennan and Resnick's framework [4]. In this paper, we focus on the first lesson in the curriculum which introduces the following challenge for students in Unity-CT: "*Create a ramp and bucket with different types of platform objects. Add a ball that must hit at least once each type of platform before landing in the bucket*". See an example solution in fig. 1. This challenge focuses on two high-level CT design practices. The first one, *Being Incremental and Iterative*, is defined as the ability to design algorithmic solutions in a step-by-step manner, which is fostered by requiring students to discover new game objects, and imagining how to use them to solve the challenge at hand. The objects introduced in this challenge include the Ball, the Ground Platform, and 4 special platforms - Ground_Spin, Ground_Moving, Ground_Slide, and Ground_Bouncy. These special platforms either spin around, move side to side, slide a ball or bounce a ball, respectively. The second one, *Testing and Debugging*, is defined as the ability to use trial-and-error to identify problems with the current solution and resolve them. This is fostered in this challenge by the need to repeatedly modify the ramp and the position of the ball until it ends in the bucket.

[1]Either via a teacher in online classrooms, or an online video otherwise.

4 FUMA FRAMEWORK

FUMA, fully described in [15], consists of two main parts - *Behavior Discovery* and *User Classification* as shown in fig. 2. In Behavior Discovery, users' interaction data is pre-processed into feature vectors. Next, a Genetic k-means clustering algorithm is applied on these features to identify users with similar interaction behaviors. FUMA automatically determines the optimal number of clusters based on three measures of cluster quality - C-index [18], Calinski-Harabasz index [5], and Silhouettes [28]. The resulting clusters are statistically compared to see if there are significant differences in learning performance, thus identifying groups of students with varying degrees of learning from the interaction with the target OELE. Then, behaviors in each cluster are mined using the Hotspot association rule mining algorithm from WEKA [10]. Hotspot extracts behaviors in the form of feature trees. These feature trees are composed of nodes in the form $X \rightarrow c$, where X is a feature-value pair and c is the predicted class label for the data points where X applies.

During User Classification, the clusters and corresponding association rules learned during Behavior Discovery are used to build a classifier. As new users interact with the OELE, they are classified in real-time into one of the identified clusters, based on a membership score that summarizes how well the user's behavior matches the association rules of a cluster. Thus, in addition to classifying students in one of the clusters, this phase returns the specific association rules describing the user's behaviors that caused the classification. These rules are then used to design real-time interventions that encourage behaviors conducive to learning and discourage those that are suboptimal for learning. Here, we point out that since FUMA is entirely data-driven, the behaviors mined in the rule-mining process may not always be interpretable or directly related to specific aspects of learning (eg. CT skills). Thus, it is crucial that we evaluate the interpretability and quality of the association rules in order to leverage them for adaptive interventions.

To formally evaluate the performances of the user classifier, FUMA is trained and tested using internal nested k-fold *cross-validation* (CV). Namely, the Behavior Discovery phase is run on the training data at the inner loop of the nested CV, to fine-tune the FUMA hyper-parameters (e.g., minimum support of the rules) and learn the clusters and rules. The trained classifiers are then tested on the test set at the outer loop of nested CV, in the User Classification phase. If the performances of the user classifier are

Table 1: Action features used for training FUMA models

Features on all action-object pairs together	• Total Number, Frequency of Actions • Average, SD of intervals in-between actions
Features on each action-object pair	• Frequency, Count of tuple, Longest repetition of this tuple • Average, SD of time intervals in-between next action • Time to first, Time to last tuple occurrence
Pausing	• Number, frequency of pauses • Average and SD of pause duration

deemed sufficient, FUMA is trained one more time on the entire data, to extract a final set of association rules that can be used to drive the design of adaptive support. For brevity, we shall refer to this as the *student model* obtained by FUMA on a dataset.

4.1 FUMA application to Unity-CT

We track student interaction logs in Unity-CT, and use this data to build student models for adaptive support. To apply FUMA on Unity-CT's interaction data, we first transform action logs into features in a similar manner to a previously successful application of FUMA on Unity-CT [20]. We generated action features for each action-object pair (as tuples) available in Unity-CT, as described in table 1. The features in table 1-(a) indicate summative statistics over all pairs, and indicate overall student engagement, while the features in table 1-(b) are generated for each pair, indicating behaviors specific to individual actions and target objects. We scale time-based features for all students by dividing these features by their interaction lengths to obtain comparable measurements despite variations in total interaction length.

To evaluate the student model built by FUMA, we generate twenty data windows corresponding to incremental percentages (5%, 10%, 15%... up to 100%) of student data. This allows us to verify how early during the interaction, the student model can perform accurate classification. This is important for our goal of providing adaptive support in real-time to students who exhibit ineffective behaviors. We evaluate student models through 10-fold nested cross-validation (CV) and evaluate its accuracy in assigning students to their final label, i.e., their cluster at the end of the interaction.

Finally, in order to analyze the differences in student learning in the clusters, we grade student solutions based on 4 binary criteria defined by the teachers who taught this lesson - whether the solution has a ramp, whether the solution has a bucket, whether the ball hits all special platforms, and whether the ball ends up in the bucket. These criteria are evaluated with either 1 or 0 if they complete it or not. Finally, the 4 criteria are averaged and students get a *solution score* of 0, 0.25, 0.5, 0.75 or 1. While these solution scores do not measure CT directly, they capture student learning performance as these criteria are necessary to complete the lesson and it provides a measurement of student progress in the lesson. Solution scores in each cluster are statistically compared to identify whether FUMA identifies clusters with a significant difference in student learning performance.

5 STUDENT MODELING FOR THE IPA

In this section, we describe the process of building a student model for adaptive support in Unity-CT. We first describe the potential datasets for our student model model. Then we apply FUMA to each of these datasets to assess whether any of these data-driven student models are accurate, and if they comprise of association rules that are interpretable and feasible for providing adaptive support in practice. Finally, we extend the data-driven student model derived by FUMA and leverage teachers' insights to build a final data-informed student model.

5.1 Datasets

All students in each of the datasets described below were recruited by UME Academy as part of their regular educational activities. These students were from grades 4-6 living in North America. The classes were held by UME Academy as extracurricular activities, as they normally do, with no involvement from researchers to ensure high ecological validity. Within each dataset, we discarded data from students who interacted less than 10 minutes with Unity-CT, based on recommendations from UME Academy's teachers who stated that students require at least 10 minutes to complete the challenge.

Unmonitored Dataset: This dataset was obtained from in-the-wild online sessions conducted in 2023. Recall that in-the-wild refers to fully, unmonitored interaction where students interact with Unity-CT online without a teacher. Students had the option to view a pre-recorded tutorial video before their interaction with the environment if they chose to do so[2]. Students were allowed to interact freely with Unity-CT for up to 60 minutes and could end the lesson at any point once they completed the challenge, which led to variable interaction length. In total, this dataset consists of 282 students with an average interaction length of 29.40 minutes (std.dev 13.69 min). We graded all 282 student solutions in this dataset. The average grade for students was 0.45 (std.dev 0.39).

Monitored Dataset This dataset was obtained from online classes conducted in 2021 and has been studied in previous work [36]. Each class includes a teacher and up to 12 students. No researcher attended the remote classrooms nor interacted with the students or teachers in any way. Each class lasts 60 minutes and includes an initial tutorial of around 20-30 minutes led by the teacher describing the basics of using the Unity editor and the steps involved to achieve the goal for the initial lesson. The remaining time consisted of the student's interaction with Unity-CT to complete the challenge. Teachers monitored students' progress and provided help to students if they needed it. Students could signal to the teachers that they are done at any point, using as much time as they needed for the lesson until the end of the 60-minute class. In total, this dataset consists of 205 students with an average interaction length with Unity-CT of 28.39 minutes (std.dev 6.23 minutes). The average grade for students was 0.61 (std.dev 0.49).

5.2 FUMA Student Model

We apply FUMA to both datasets after processing the data into features (table 1). FUMA found 2 clusters of students in each dataset.

[2]The video is on Youtube, so we cannot know for sure whether they fully watched it.

Table 2: Comparison of FUMA applied to datasets

Dataset	N	Cluster Sizes	Kruskal Wallis test of Learning Performance on Clusters	Mean classification accuracy	Comparison against baseline	Interpretability of rules
Unmonitored	282	114, 168	$p < 0.01, \eta^2 = 0.107$	85.65% (std.dev 3.32)	$p < 0.0001, d = 11.20$	Rules are too complex
Monitored	205	60, 145	$p < 0.0001, \eta^2 = 0.124$	83.80% (std.dev 4.54)	$p < 0.0001, d = 7.90$	Rules are interpretable

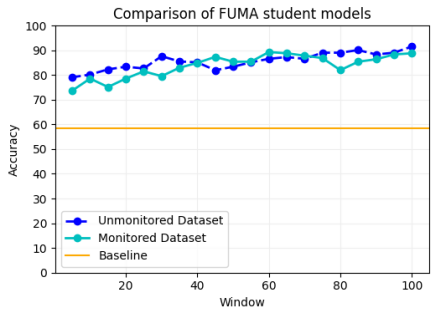


Figure 3: Accuracy of FUMA over 5% incremental data windows trained and tested within the monitored dataset.

Within each dataset, to assess the differences in learning performance between the two clusters, we use the Kruskal-Wallis test to test differences in learning performance using the students' solution scores and report p-values with eta-squared effect size [32]. We describe the test statistics in table 2. The Kruskal-Wallis test showed significant differences in learning between the clusters in both datasets with a medium-large effect size[3] in both datasets. These clusters shall be referred to as the *high-performing* (HP) and *low-performing* (LP) cluster for this analysis.

We plot in fig. 3 the total accuracy (i.e., *correct_predictions* / *all_predictions*) for the student model built in each dataset over incremental data windows against a stratified baseline, which makes predictions based on the class probability distributions in the training set. We run t-test comparisons of total accuracy over-time against the baseline and describe the average accuracy of the student models in table 2. We see that both student models beat the baseline significantly with a large effect size[4]. To see if we could use these student models in practice, we look next at the association rules extracted by FUMA in each of these student models.

Rules extracted from the Unmonitored dataset: There were 11 nodes in the HP feature-tree, but the LP feature-tree was inherently complex, with over 458 nodes. The maximum depth of the tree was 6, and the total number of leaf nodes was 214. This resulted in 163 distinct rules, which is much larger than previous applications of association rule-mining for adaptive support (e.g., 15 LP rules in ACSP [16], 12 LP rules in CCK [8], 8 LP rules in Betty's Brain [17]).

Rules extracted from the Monitored Dataset: The rule-trees were simple and we were able to extract 3 interpretable rules for the HP cluster and 4 for the LP cluster. These rules were interpretable in the sense that a researcher who was familiar with Unity-CT and its curriculum extracted rules that they deemed more understandable so as to avoid overwhelming teachers with too many complicated rules when we present them for their validation.

Overall, while both student models were accurate, the excessive number of LP rules make the student model trained on the unmonitored dataset non-interpretable and infeasible for providing adaptive support in practice. For the student model trained on the monitored dataset, we were able to extract interpretable behaviors. We will proceed with this student model and refine it through expert insights by presenting the association rules to UME Academy's teachers. Once this is done, in section 5.4, we test the final student model on the unmonitored dataset to understand the implications of using them for adaptive support in the real-world setting.

5.3 Final Association Rules

We presented the association rules to two UME Academy's teachers in a randomized order and a blind manner, concealing which FUMA cluster they come from. We obtained their input on whether they believe the behavior exhibited would be by an LP student or an HP student, and comments on how adaptive interventions could be provided if needed. Through this process, we narrowed down a set of FUMA rules that align with the teachers, and subsequently, we used feedback from teachers to design adaptive help interventions in the form of an IPA (as elaborated in section 6). Table 3 presents the interpretable behaviors mined by FUMA in the LP and HP clusters and the responses of teachers to the respective behaviors.

From table 3, we find that there is an agreement in all the HP rules (behaviors 1,2 and 3) found by FUMA indicating that HP students get started on the ramp early in the challenge by manipulating special objects. Behavior 7 acts as a corollary to these behaviors, in the sense that LP students do not manipulate special platforms early in the challenge but rather wait till the middle of the challenge. It is necessary to work with the special objects in order to build a ramp, and not manipulating (move, select, rotate or rescale) any special objects could indicate that the student is struggling and needs assistance. This behavior is associated with the CT skill, Being Incremental and Iterative, and requires assistance to guide the student to the initial steps needed to complete the lesson. Moreover, the teachers unanimously mentioned that we should extend these behaviors and provide adaptive support to students who fail to do any type of manipulation of a special ground platform. The other high-level behavior that stands out is that LP students press the Play Button Infrequently reflected in behaviors 4 and 5, subsequently testing and debugging their game inadequately. This behavior relates to the CT skill, Testing and Debugging, and for success with Unity-CT, students must test out their games often. Teachers suggested that help interventions could be provided to students who do not test their games often enough. The only rule that was in disagreement with the teachers was the rule relating

[3]Guidelines for η^2 are 0.01 (small), 0.06 (medium),0.14 (large) as per [27]
[4]Guidelines for d are 0.2 (small), 0.5 (medium) and 0.8 (large) as per [6]

Table 3: Teacher Assessment of Behaviors identified by FUMA. T1 and T2 are the assessments of the behaviors by teacher 1 and teacher 2. Behaviors that are in agreement with teachers are highlighted in bold.

Behavior	Cluster	T1	T2
1. Student moved a Ground_Moving platform for the first time early in the challenge	HP	HP	HP
2. Student moved a Ground_Slide platform for the first time early in the challenge	HP	HP	HP
3. Student selected a Ground_Slide platform for the first time early in the challenge	HP	HP	HP
4. Student presses the Play button to enter Play mode Infrequently	LP	LP	LP
5. Student presses the Play button to enter Edit mode Infrequently	LP	LP	LP
6. Student Duplicates balls for the first time during the middle of the challenge	LP	HP	HP
7. Student rotated a Ground_Moving platform for the first time during the middle of the challenge	LP	LP	LP

to duplicating balls. Teachers suggested that this behavior actually reflects a HP student who is possibly testing various ways to make balls land in the bucket.

By disregarding the rule that was in disagreement with the teacher and incorporating the suggestion on expanding the first high-level behavior, we refine our data-driven student model into what we shall call the *data-informed* student model. The data-informed student model looks at the two high-level behaviors - not manipulating any special platform, and pressing the play button infrequently and deems the student as LP if either of these behaviors occur, or HP otherwise.

5.4 Testing Data-informed Student Model on the unmonitored dataset

We plot the comparison of the performance of the data-informed student model against the student model trained on the unmonitored dataset and a baseline in fig. 4.

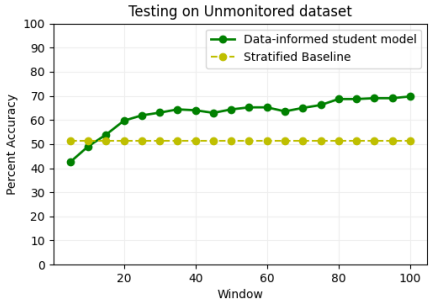
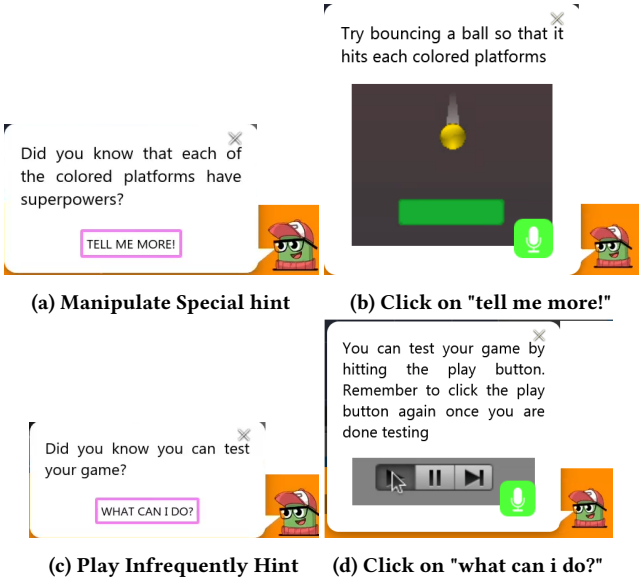


Figure 4: Evaluation of data-informed student model

The data-informed student model achieves an average accuracy of 62.81 (std.dev 6.78) and a t-test comparison show that it significantly outperforms both the baseline overall with a large effect size ($p < 0.0001, d = 2.28$). Moreover, McNemar's test [19] over each incremental window show that the data-informed student model significantly outperforms the baseline as early as the 20% data window with a large effect size ($p < 0.05, w = 0.70$)[5]. In practice, when using this for real-time adaptive support, we could start providing help interventions at this point, which is 5.7 minutes on average.

[5]Guidelines for Cohen's w [6] are small if w = 0.1, medium if w = 0.3, large if w = 0.5



(a) Manipulate Special hint **(b) Click on "tell me more!"**

(c) Play Infrequently Hint **(d) Click on "what can i do?"**

Figure 5: Hints by the IPA

6 DESIGN OF THE IPA

In this section, we will describe the design of adaptive help interventions for the two types of ineffective behaviors identified in section 5.3. We phrase both help provisions carefully using recommendations from the teachers by asking them what they would say to students exhibiting these ineffective behavior. We follow the design used in [36], and we structured the help content provided for each of the two behaviors in the progression of speech bubbles flagging the problem, and explaining how they can solve it as shown in fig. 5. The *Manipulate Special* hint is provided when the student has not manipulated any special objects yet, and the *Play Infrequently* hint is provided when students infrequently hit the play button. Infrequenty is defined as less than twice per minute, i.e., the interval learned by FUMA in the LP rule-tree for the monitored dataset.

We start detecting these ineffective behaviors and start providing hints only after 5.7 minutes in to the interaction as we found that the data-informed student model significantly outperformed the baseline in section 5.4. Following the methodology used in [16, 21] we provide a second hint to users if their behaviors do not change and the hint is triggered again. After a hint is delivered, if a hint of another type is triggered shortly after, we wait at least 14 actions after the initial hint was delivered to provide the new hint. If the same type of hint is triggered shortly after, we wait at least 28 actions after the initial hint to provide help in the form of a repeated hint. These values were obtained by scaling the thresholds used in the ACSP applet in [16] (10 and 20, respectively) proportionally with respect the number of actions per minute in Unity-CT[6]. We follow this mechanism because there was extensive pilot testing for these thresholds [16]. Once two hints of the same type are delivered, we do not provide any further hints of the same type (as in [21, 36]) to prevent help interventions from feeling intrusive.

7 EVALUATION OF THE IPA

In this section, we first describe the control and experimental groups that we use to evaluate the IPA. Then, to analyze the effectiveness of hints in Unity-CT, we look at the distribution of hints delivered by the IPA in the experimental group. We then compare student learning performance between the control group and the experimental group. Finally, we will look at the effect that rectifying student behaviors according to specific hints has on student learning.

7.1 User Study

To evaluate the effectiveness of the IPA, we conducted a user study in self-directed classes with Unity-CT in 2023 and 2024. We leverage two different cohorts as part of UME Academy's regular in-the-wild educational activities for our user study, one as a control group (without the IPA) and one as an experimental group (with the IPA), so that we can formally compare the impact of the presence of the IPA. The control group comprises the students in the unmonitored dataset fully described in section 5.1, who interacted with Unity-CT without the presence of an IPA in 2023. The experimental group consists of students interacting with Unity-CT with the IPA toggled on from January to March 2024. The IPA would provide the help interventions we designed in section 6. Comparing these two groups is deemed fair as the nature of the interaction with Unity-CT in both cases was in the wild, i.e., self-directed and fully unmonitored. Moreover, the lesson was advertised by UME Academy to similar students in both groups (North American students in grades 4-6), and the recruiting process and lesson content were kept exactly the same. For this evaluation, we discard data from students with less than 10 minutes of interaction data for consistency with what was done in the other datasets, as reported in section 5.1. As a result, we obtained 282 students in the control group and 84 students in the experimental group.

7.2 Results on Hint Delivery

We look at the distribution of hints delivered in the experimental group in fig. 6. Over 89% of students received at least one hint,

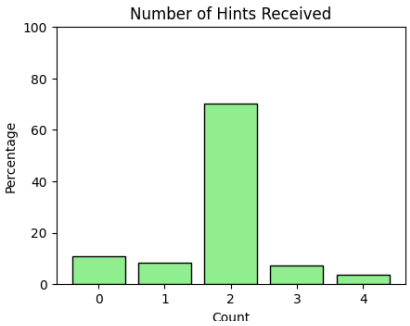


Figure 6: Number of hints received by the students.

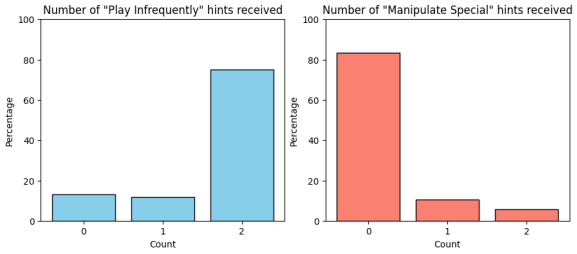


Figure 7: Distribution of specific hint types

indicates that behaviors derived in monitored online classrooms are still relevant in the unmonitored, self-directed setting. This result is promising as it showcases the ability of FUMA to leverage data from a constrained setting in order to provide data-driven adaptive support in self-directed lessons.

To further understand how specific hints were triggered, we plot the histograms in fig. 7. Only 16.7% of users receive the "Manipulate Special" hint. It is necessary that students move the special objects in order to build the ramp for the challenge. Thus, it is expected as most students who interact with the system would interact with the special objects, and this hint is targeted towards struggling students who do not touch the special objects. 76.9% of users receive at least one "Play Frequently" hint. This is much more common as it is a reminder to students to test their games out often. It looks like students in the experimental group did not test their games frequently enough. A reason for this could be that students skip the tutorial and are thus not aware of frequently testing the game.

7.3 Results on Hint Effectiveness

Comparing Learning: To objectively evaluate the effect of student ability to complete the challenge between the control group and experimental group, student solutions were graded as described in section 4.1. We perform statistical comparisons (see table 4) using a ranked Welch test to account for unequal variances and sample sizes [7, 37]. We find that the experimental group has statistically significantly higher learning performance as compared to the control group with a medium effect size[7]. This suggests that the IPA and the introduction of hints to students improved students' ability

[6]ACSP users performed on average 14.1 actions per minute as evaluated from [16], while students in Unity-CT performed 20 actions per minute.

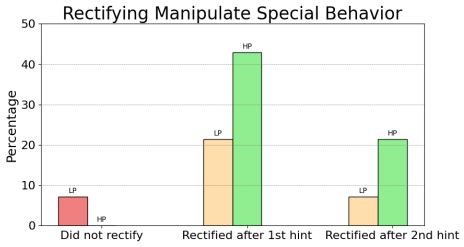
[7]Guidelines for d are 0.2 (small), 0.5 (medium) and 0.8 (large) as per [6]

Table 4: Learning Performance Comparison (mean +- std.err)

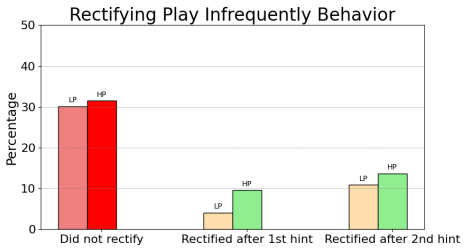
Experimental	Control	Ranked Welch T-test
0.62 +- 0.045	0.45 +- 0.023	p <0.005, d = 0.43

to learn and complete the lesson in Unity-CT. As opposed to previous work [21, 36], our work showcases that a data-informed IPA significantly improves student learning performance in Unity-CT.

Effect of specific hints on learning performance: To understand the effect that the two specific hints had on learning performance, we looked at whether users rectify their behavior after receiving hints. In fig. 8, we look at students who received hints and their learning performance outcome, categorized by whether they rectified their behavior. Here, we use a median-split to categorize students into HP or LP based on their solution scores, following the methodology in [16].



(a) Rectifying "Manipulate Special" behavior



(b) Rectifying "Play Frequently" behavior

Figure 8: Learning performance outcome (HP or LP) in relation with rectifying hints

Table 5: Correlation of students' learning performance with rectifying behaviors after receiving a hint.

Hint	#students (max 84)	Correlation (Pearson *r*)	p-value	95% confidence interval
Manipulate special	14	0.57	<0.05	(0.049, 0.84)
Play Infrequently	73	0.14	>0.1	(-0.12,0.32)

First, we look at students who received the "Manipulate Special" hint (see fig. 8a). Students who received this hint and did not rectify their behavior after receiving this hint, ended up being all LP. Moreover, over twice as many users who rectified their behavior after receiving a hint end up being HP rather than LP. To formally evaluate whether following these hints had an effect on learning performance, we perform a statistical analysis of pearson's correlation coefficient in table 5. We see that students who followed the "Manipulate Special" hint and rectified their behavior had a statistically significant positive impact on learning performance with a moderate[8] effect size. Now looking at students who received the "Play Infrequently" hint (see fig. 8b), while the correlation analysis yield a weak positive correlation, we still observe a trend that the majority of users who rectified their behavior after receiving a hint end up being HP rather than LP.

Overall, we see that students who receive and follow the help interventions delivered by the IPA are more likely to have increased learning performance. This showcases the value on the data-informed student model and the behaviors extracted by FUMA.

8 CONCLUSIONS AND FUTURE WORK

In this paper, we built and evaluated an IPA for unmonitored interaction with Unity-CT, an OELE that fosters CT skills in young learners. Our work provides a proof-of-concept that it is possible to build student models in a controlled setting and transfer them to an unmonitored setting. We build the student model for our IPA by leveraging data from monitored, online classes. We were able to extract interpretable student behaviors and validated these behaviors with teachers at UME Academy. We refined this student model through inputs from the teachers and evaluated the performance of this data-informed student model in the unmonitored setting. Given that the data-informed student model performed well, we designed our IPA help interventions for in-the-wild interaction with Unity-CT in collaboration with teachers. Lastly, we evaluated our IPA through a formal user study. We find that students who receive adaptive support in the form of help interventions delivered through an IPA had significantly higher learning performance as compared to students who do not receive any adaptive support. This is important as it is crucial to provide support to struggling students who may face difficulties due to the exploratory nature of OELEs. Moreover, our findings provide us insights on the effect of specific hints on student learning performance, which can be used to refine and improve future help provision.

Future work entails extending the IPA in Unity-CT to other lessons in UME Academy's curriculum, and possibly other OELEs. Future work also entails revising and further testing of our help interventions. In particular, while the "Manipulate Special" hint was effective, it was delivered to relatively few students, so revising the trigger for this hint may improve the quality to be applicable to more students. Further personalization of help interventions, such as to user profiles or characteristics may improve the quality of help provided and is part of future work.

Acknowledgement. This work was supported by the Natural Sciences and Engineering Research Council of Canada (Grant #ALLRP-567500-21) and UME Academy Ltd.

[8]Interpretation guidelines for correlation take from [30]

REFERENCES

[1] Valerie Barr and Chris Stephenson. 2011. Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *Acm Inroads* 2, 1 (2011), 48–54.

[2] Satabdi Basu, Gautam Biswas, and John S Kinnebrew. 2017. Learner modeling for adaptive scaffolding in a computational thinking-based science learning environment. *User Modeling and User-Adapted Interaction* 27 (2017), 5–53.

[3] Gautam Biswas, James R Segedy, and Kritya Bunchongchit. 2016. From design to implementation to practice a learning by teaching system: Betty's Brain. *International Journal of Artificial Intelligence in Education* 26 (2016), 350–364.

[4] Karen Brennan and Mitchel Resnick. 2012. New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada*, Vol. 1. 25.

[5] Tadeusz Caliński and Jerzy Harabasz. 1974. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods* 3, 1 (1974), 1–27.

[6] Jacob Cohen. 1988. Statistical power analysis for the behavioral sciences New York. *NY: Academic* 54 (1988).

[7] William J Conover and Ronald L Iman. 1981. Rank transformations as a bridge between parametric and nonparametric statistics. *The American Statistician* 35, 3 (1981), 124–129.

[8] Lauren Fratamico, Cristina Conati, Samad Kardan, and Ido Roll. 2017. Applying a framework for student modeling in exploratory learning environments: Comparing data representation granularity to handle environment complexity. *International Journal of Artificial Intelligence in Education* 27 (2017), 320–352.

[9] Shuchi Grover and Roy Pea. 2013. Computational thinking in K–12: A review of the state of the field. *Educational researcher* 42, 1 (2013), 38–43.

[10] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter* 11, 1 (2009), 10–18.

[11] Michael Hannafin, Susan Land, and Kevin Oliver. 2013. Open learning environments: Foundations, methods, and models. In *Instructional-design theories and models*. Routledge, 115–140.

[12] Michael J Hannafin. 1995. Open-ended learning environments: Foundations, assumptions, and implications for automated design. In *Automating instructional design: Computer-based development and delivery tools*. Springer, 101–129.

[13] Michael J Hannafin, Craig Hall, Susan Land, and Janette Hill. 1994. Learning in open-ended environments: Assumptions, methods, and implications. *Educational Technology* 34, 8 (1994), 48–55.

[14] Janette R Hill and Susan M Land. 1998. Open-Ended Learning Environments: A Theoretical Framework and Model for Design. (1998).

[15] S Kardan and C Conati. 2011. A Framework for Capturing Distinguishing User Interaction Behaviours in Novel Interfaces. In *EDM 2011 4 th International Conference on Educational Data Mining*. 159.

[16] Samad Kardan and Cristina Conati. 2015. Providing adaptive support in an interactive simulation for learning: An experimental evaluation. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 3671–3680.

[17] John S Kinnebrew, Kirk M Loretz, and Gautam Biswas. 2013. A contextualized, differential sequence mining method to derive students' learning behavior patterns. *Journal of Educational Data Mining* 5, 1 (2013), 190–219.

[18] James A Koziol and Zhenyu Jia. 2009. The concordance index C and the Mann–Whitney parameter Pr (X> Y) with randomly censored data. *Biometrical Journal: Journal of Mathematical Methods in Biosciences* 51, 3 (2009), 467–474.

[19] Peter A Lachenbruch. 2014. McNemar test. *Wiley StatsRef: Statistics Reference Online* (2014).

[20] Sébastien Lallé, Özge Nilay Yalçın, and Cristina Conati. 2021. Combining Data-Driven Models and Expert Knowledge for Personalized Support to Foster Computational Thinking Skills. In *LAK21: 11th International Learning Analytics and Knowledge Conference*. 375–385.

[21] Sébastien Lallé, Özge Nilay Yalçın, and Cristina Conati. 2023. How to Repeat Hints: Improving AI-Driven Help in Open-Ended Learning Environments. In *International Conference on Artificial Intelligence in Education*. Springer, 760–766.

[22] Sébastien Lallé and Cristina Conati. 2020. A Data-Driven Student Model to Provide Adaptive Support during Video Watching Across MOOCs. In *Proceedings of AIED 2020, the 21st International Conference on Artificial Intelligence in Education*. Ifrane, Morocco.

[23] Susan M Land. 2000. Cognitive requirements for learning with open-ended learning environments. *Educational Technology Research and Development* 48, 3 (2000), 61–78.

[24] Ati Suci Dian Martha and Harry B Santoso. 2019. The design and impact of the pedagogical agent: A systematic literature review. *Journal of educators Online* 16, 1 (2019), n1.

[25] Anabil Munshi, Gautam Biswas, Ryan Baker, Jaclyn Ocumpaugh, Stephen Hutt, and Luc Paquette. 2023. Analysing adaptive scaffolds that help students develop self-regulated learning behaviours. *Journal of Computer Assisted Learning* 39, 2 (2023), 351–368.

[26] Thomas Price, Rui Zhi, and Tiffany Barnes. 2017. Evaluation of a Data-Driven Feedback Algorithm for Open-Ended Programming. *International Educational Data Mining Society* (2017).

[27] John TE Richardson. 2011. Eta squared and partial eta squared as measures of effect size in educational research. *Educational research review* 6, 2 (2011), 135–147.

[28] Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20 (1987), 53–65.

[29] Gavriel Salomon and Howard Gardner. 1986. The computer as educator: Lessons from television research. *Educational Researcher* 15, 1 (1986), 13–17.

[30] Patrick Schober, Christa Boer, and Lothar A Schwarte. 2018. Correlation coefficients: appropriate use and interpretation. *Anesthesia & analgesia* 126, 5 (2018), 1763–1768.

[31] Tasmia Shahriar and Noboru Matsuda. 2023. What and How You Explain Matters: Inquisitive Teachable Agent Scaffolds Knowledge-Building for Tutor Learning. In *International Conference on Artificial Intelligence in Education*. Springer, 126–138.

[32] Maciej Tomczak and Ewa Tomczak. 2014. The need to report effect size estimates revisited. An overview of some recommended measures of effect size. (2014).

[33] Karen L Webber and HY Zheng. 2020. Data analytics and the imperatives for data-informed decision making in higher education. *Big data on campus: Data analytics and decision making in higher education* (2020), 3–29.

[34] Jeannette M Wing. 2006. Computational thinking. *Commun. ACM* 49, 3 (2006), 33–35.

[35] Özge Nilay Yalçın, Sébastien Lallé, and Cristina Conati. 2023. The Impact of Intelligent Pedagogical Agents' Interventions on Student Behavior and Performance in Open-Ended Game Design Environments. *ACM Trans. Interact. Intell. Syst.* 13, 3, Article 11 (2023).

[36] Özge Nilay Yalçin, Sébastien Lallé, and Cristina Conati. 2022. An Intelligent Pedagogical Agent to Foster Computational Thinking in Open-Ended Game Design Activities. In *27th International Conference on Intelligent User Interfaces*. 633–645.

[37] Donald W Zimmerman and Bruno D Zumbo. 1993. Rank transformations and the power of the Student t test and Welch t'test for non-normal populations with unequal variances. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale* 47, 3 (1993), 523.