Data Analytics and Visualization Approaches for Online Learning of Math and Computational Thinking

HKUST VisLab
The Hong Kong University of Science and Technology
http://vis.cse.ust.hk/groups/e-learning/

1 November 2020

TABLE OF CONTENTS

DATA ANALYTICS AND VISUALIZATION APPROACHES FOR ONLINE LEARNING OF MATH AND COMPUTATIONAL THINKING

ABSTRACT

With online education becoming popular in the past decades, there has been an increasing number of online learning platforms that provide students with online questions to cultivate their problem-solving skills. For example, various MOOC platforms (e.g., Khan Academy) and online question pools (e.g., LeetCode, LearnLex) offer interactive intellectual and programming exercises. However, the enormous learning resources and the lack of educators in the online learning settings make it hard for students to achieve personalized learning. There are two major challenges for people to remove these obstacles. The first is how to collect students' fine-grained behavior data during the problem-solving process. The second is to utilize the collected data to infer students' cognitive state in the dynamic problem-solving process and further solve downstream tasks, such as student learning performance prediction and personalized learning path recommendation.

In this white paper, we introduce a general way to collect students' fine-grained mouse movement data during their problem-solving process. Based on the data collected, we analyze the detailed problem-solving sequences. First, we propose QLens, a visual analytics system to help question designers analyze students' problem-solving behaviors in multi-step questions to improve question design at a micro-level. Second, we design, SeqDynamics, another visual analytics system to evaluate students' problem-solving dynamics among a series of questions from both cognitive and non-cognitive perspectives at a macro level. Third, we extract the interaction features from students' mouse movement trajectories and develop advanced deep learning algorithms (i.e., GNN) to predict their performances on any given questions. Furthermore, we try to improve their self-learning skills by proposing PeerLens, an interactive visual analytics system that enables peer-inspired learning path planning and visual interventions to regulate their problem-solving behaviors. We conduct extensive quantitative evaluations, such as case studies, user studies, and expert interviews, to demonstrate the effectiveness and usefulness of our proposed data analytics and visualization approaches.

CHAPTER 1

INTRODUCTION

With online education becoming increasingly popular in the past decades, various online learning platforms have been developed to provide students with all kinds of learning materials, including online exercises. For example, various MOOC platforms (e.g., Khan Academy), online question pools (e.g., LeetCode, LearnLex), and intelligent tutoring systems (e.g., SimStudent) offer interactive maths questions and programming exercises (Xia et al., 2019a). Compared with traditional learning, students can access these learning materials more flexibly and conveniently. However, the huge amount of learning resources and the imbalance in the number of educators and students make it challenging for students to achieve personalized learning. In this white paper, we attempt to bridge the gap between educators and students in students' problem-solving behaviors.

Problem-solving processes can reflect students' cognitive abilities as well as noncognitive traits. Cognitive skills refer to the core skills that the brain uses to think, read, learn, remember, reason, and pay attention, while noncognitive traits and behaviors refer to specific characteristics such as motivation, conscientiousness, perseverance, self-regulation, and collaboration (Farkas, 2003). In the online problem-solving scenario, for educators and question designers, they need to understand students' problem-solving logic when solving a multi-step question and their learning habits when solving a series of questions. With a deep understanding of students' learning habits and problem-solving behaviors, they can improve question designs and give customized instructions to student groups with different cognitive abilities and non-cognitive traits. Meanwhile, students on online learning platforms need to enhance their self-learning skills, such as learning paths planning and learning habits regulation.

However, two challenges are encountered when educators and question designers are trying to handle the above tasks in online learning platforms. The first is how to collect students' fine-grained behavior data during the problem-solving process. The second is how to utilize the data to infer students' cognitive state in the dynamic problem-solving process and further solve downstream tasks (e.g., student performance prediction and personalized learning paths planning). For example, each learner's problem-solving behaviors, is ultimately multi-dimensional time-series data and how to represent and interpret such data with meaningful semantics (i.e., cognitive and non-cognitive traits and behaviors) is non-trivial; summarizing students' problem-solving dynamics in different levels of details to facilitate convenient comparison can be difficult; mining and representing peers' learning data to assist students to regularize their learning behaviors and better plan their online learning is hard to achieve.

With the sponsor from Innovation and Technology Fund of HKSAR (Project No.: ITS/388/17FP) and our industry collaborator, Trumptech (Hong Kong) Ltd, we conduct a series of research on students' problem-solving process in online learning. Fig. 1.1 provides an overview of our research on employing data analytics and visualization techniques to facilitate online learning, which mainly consists of three modules. The first module is data collection, which introduces a general way to collect students' fine-grained mouse movement data and submission records in the problem-solving process. Based on the data collected, the second module is the problem-solving analysis and its evaluation. It analyzes the detailed problem-solving sequences, extracts the interaction features from students' mouse movement trajectories to predict students' performance, and further assists students in path planning and behavior regulation. In particular, two visual analytics systems, QLens and SeqDynamics, are proposed to analyze students' problem-solving behaviors at a micro-level (multi-step within questions) or at a macro level (among a series of questions), respectively. Moreover, student mouse movement interaction features and new GNN models are proposed to predict students' performance on any given question. The third module is evaluation, where we conduct various quantitative assessments, case studies, user studies, and expert interviews to demonstrate the effectiveness and usefulness of our proposed data analytics and visualization approaches. Table 1.1 lists the publications in this white paper.

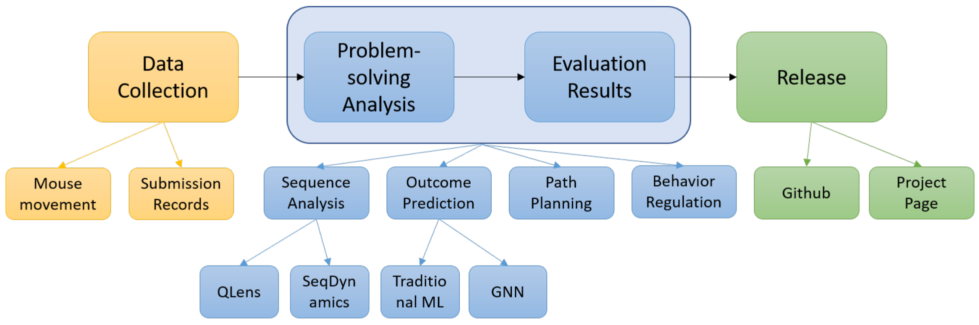


Figure 1.1: The framework of the white paper contains three modules. Data collection module collect students' fine-grained mouse movement data. Data analytics & evaluation module introduces and evaluate visual analytic systems to analyze students' problem-solving processes, advanced machine learning techniques to predict students' performance, and visual interventions to assist students' learning path planning and self-regulation. Techniques release module publishes the github repositories and the project page.

NO.	Title	Venue	Section
1	Visual Analytics of Student Learning Behaviors on K-12 Mathematics E-learning Platforms	IEEE VIS 2019 (poster)	2.2.1
2	QLens: Visual Analytics of Multi-step Problem-solving Behaviors for Improving Question Design	IEEE VIS 2020	2.2.1
3	SeqDynamics: Visual Analytics for Evaluating Online Problem-solving Dynamics	Euro VIS 2020	2.2.2
4	Predicting Student Performance in Interactive Online Question Pools Using Mouse Interaction Features	Learning Analytics & Knowledge 2020	2.3.1
5	Peer-inspired Student Performance Prediction with Graph Neural Network	CIKM 2020	2.3.2
6	PeerLens: Peer-inspired Interactive Learning Path Planning in Online Question Pool	CHI 2019	2.4
7	Using Information Visualization to Promote Students'Reflection on"Gaming the System"in Online Learning	Learning @ Scale 2020	2.5

Table 1.1: Our publication list in this white paper.

CHAPTER 2

OUR METHODS

2.1 Data Collection

2.1.1 Mouse Movement Data Collection

We design and implement a general interaction data collection technique which could be applied in different platforms. Its implementation logic is shown in Fig. 2.1.

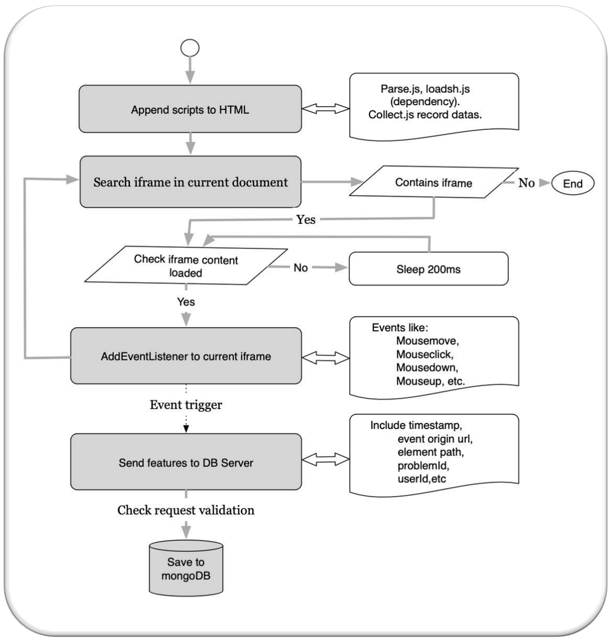
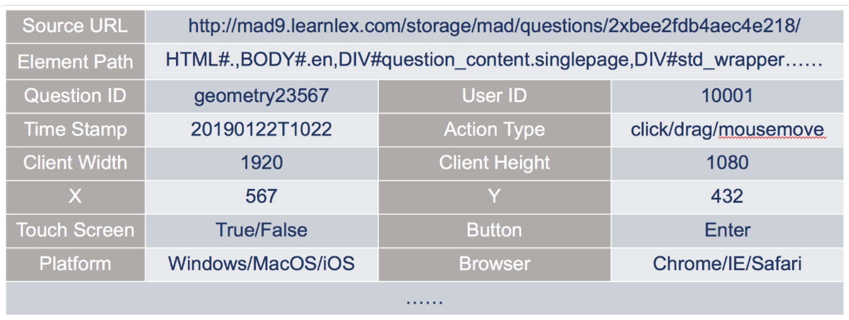

Append scripts to HTML
Parse.js, loadsh.js (dependency). Collect.js record datas.
Search iframe in current document
Contains iframe
No
End
Yes
Check iframe content loaded
No
Sleep 200ms
Yes
AddEventListener to current iframe
Events like: Mousemove, Mouseclick, Mousedown, Mouseup, etc.
Event trigger
Send features to DB Server
Include timestamp, event origin url, element path, problemId, userId,etc
Check request validation
Save to mongoDB

Figure 2.1: Framework of interaction data collection.



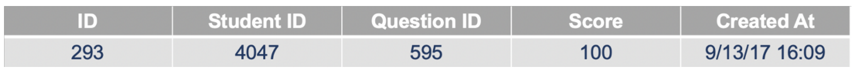
Source URL	http://mad9.learnlex.com/storage/mad/questions/2xbee2fdb4aec4e218/		
Element Path	HTML#.,BODY#.en,DIV#question_content.singlepage,DIV#std_wrapper……		
Question ID	geometry23567	User ID	10001
Time Stamp	20190122T1022	Action Type	click/drag/mousemove
Client Width	1920	Client Height	1080
X	567	Y	432
Touch Screen	True/False	Button	Enter
Platform	Windows/MacOS/iOS	Browser	Chrome/IE/Safari
……			

Figure 2.2: Mouse movement data format

Fig. 2.2 is a format example of our collected data and it shows all dimensions of the data, such as the trajectories and the drag/click actions of the mouse, user login time/frequency, user average scores and time spent on questions. These data dimensions are visualized in various forms for learning behavior analysis. The analysis can help the instructors as well as the users to better understand the learning behaviors and patterns, so as to improve the design of lecture materials and the learning efficiency accordingly.

2.1.2 Submission Data Collection

Apart from the mouse movement data, we required students' submission data from an E-Learning platform, which format is shown in Fig. 2.3. The dimensions of the submission data contain the submission ID, student ID, questions ID, score, and the submission time, which give us a baseline about students' performance and could be used as labels to predict students' future performance.



ID	Student ID	Question ID	Score	Created At
293	4047	595	100	9/13/17 16:09

Figure 2.3: Submission data format

2.2 Problem-solving Sequence Analysis

2.2.1 QLens: Problem-solving Sequence Analysis within Multi-step Questions

There has been an increasing number of learning platforms that provide students with multi-step questions to cultivate their problem-solving skills. To guarantee the high quality of such learning materials, question designers need to inspect how students' problem-solving processes unfold step by step to infer whether students' problem-solving logic matches the design intents of the questions. They also need to compare the behaviors of different groups (e.g., students from different grades) to distribute questions to students with the right level of knowledge. The availability of fine-grained interaction data, such as mouse movement trajectories from the online platforms, provides the opportunity to analyze problem-solving behaviors. However, it is still challenging to interpret, summarize, and compare the high dimensional problem-solving sequence data. In this chapter we present a visual analytics system, *QLens*, to help question designers inspect detailed problem-solving trajectories, compare different student groups, distill insights for design improvements.

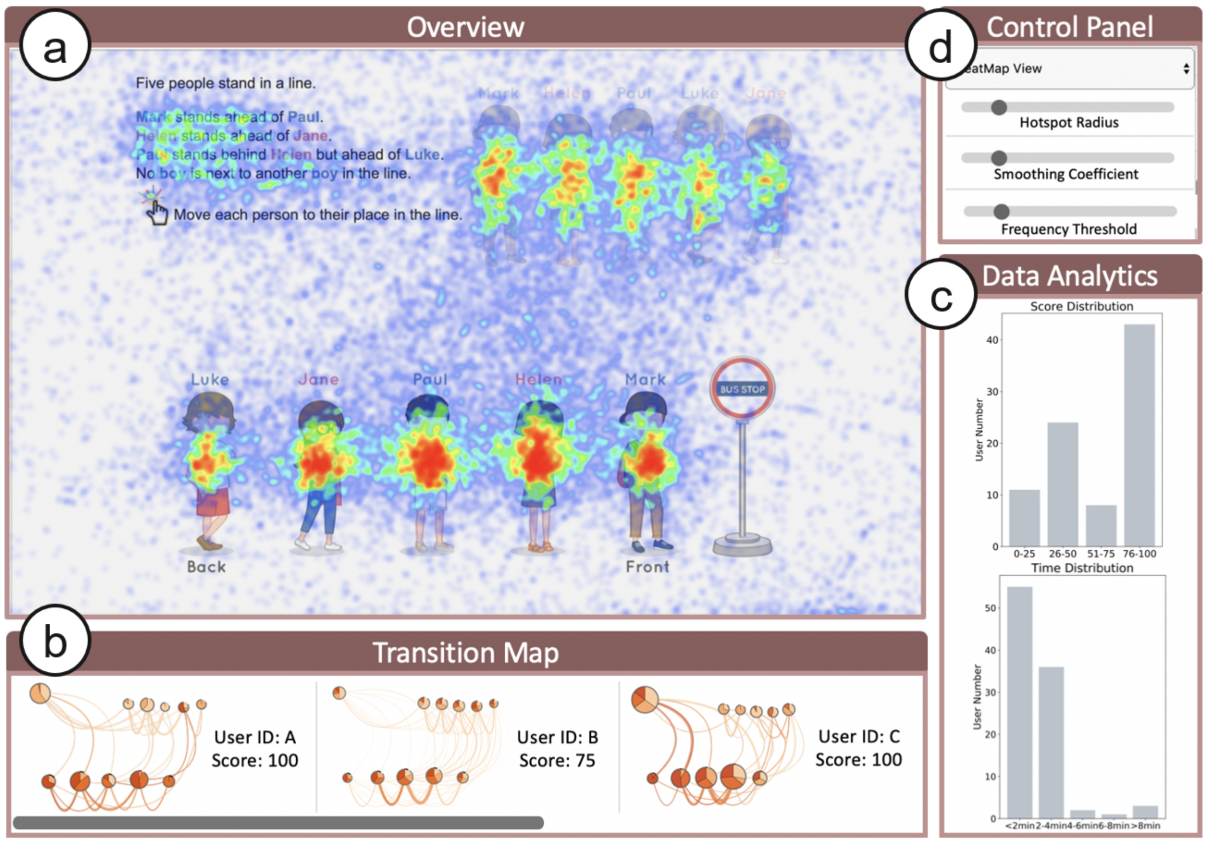


Figure 2.4: The alpha version of QLens. (a) Overview of the problem and the heat map shows the problem-solving interaction data; (b) Transition map view reveals the detailed problem-solving steps; (c) Data analytics view displays the correlation between learning attributes and learning performance;(d) Control panel offers the function to adjust the parameters of Region of Interests (ROIs).

Alpha Version of QLens

The system shown in Fig. 2.4 is the alpha version of *QLens*. In the alpha version, we first extract the regions of interests according to the density of students' mouse movements trajectories (Fig. 2.4a) and then use the transition map to represent in which order and frequency a student interacts with different regions of interests.

As for the transition map in Fig 2.5, regions with intensive interactions are extracted as ROIs and visualized through pie charts connected by arcs according to mouse movement streams to form a transition map. The size of the pie charts represents the number of interactions over the ROIs; the color of the arcs and sectors represents the time order. In this way, the map reveals students problem-solving steps. We can see that the third and fourth pie at the bottom are larger than others and the lines between them are much thicker, which means large amounts of transitions between these two ROIs. This transition map can be used to summarize a group of students' behaviors and also to compare the transition maps of two students or two groups of students' problem-solving behaviors.

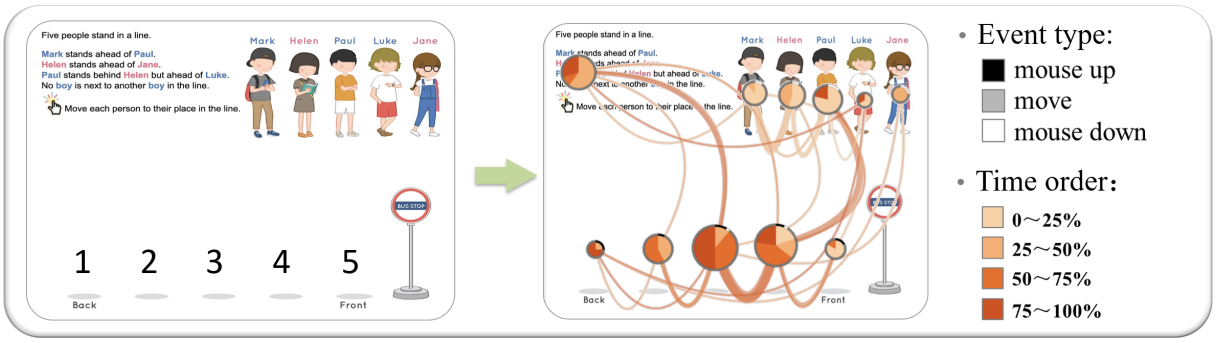
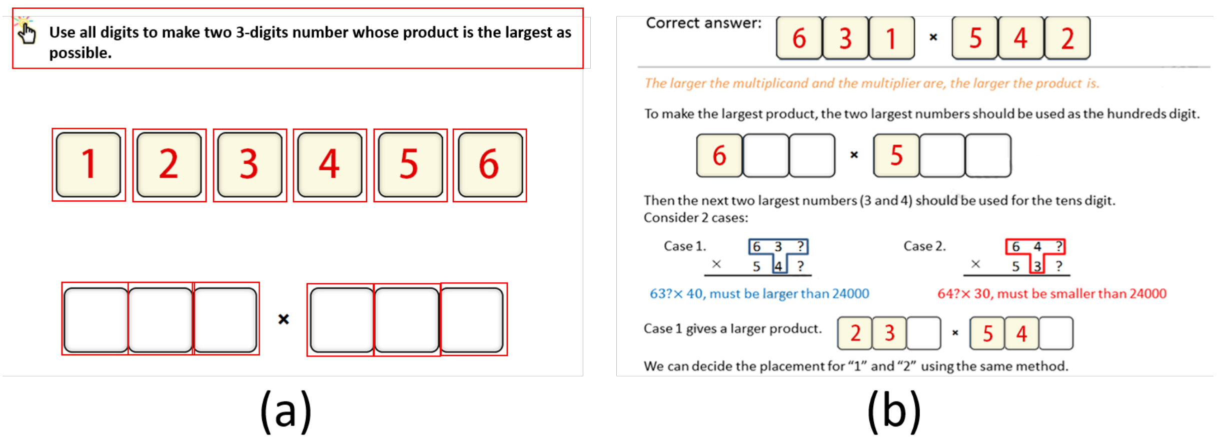

Five people stand in a line.
Mark stands ahead of Paul.
Helen stands ahead of Jane.
Paul stands behind Helen but ahead of Luke.
No boy is next to another boy in the line.
Move each person to their place in the line.
Mark
Helen
Paul
Luke
Jane
1
2
3
4
5
Back
Front
BUS STOP
Event type:
■ mouse up
▨ move
□ mouse down
Time order:
0～25%
25～50%
50～75%
75～100%

Figure 2.5: The transition map demonstrates the detailed problem-solving steps with event types and time information.

More details can be referred in the paper (Xia et al., 2019b).

Problem-solving Behavior Modeling


Use all digits to make two 3-digits number whose product is the largest as possible.

Correct answer: 631 × 542

The larger the multiplicand and the multiplier are, the larger the product is.

To make the largest product, the two largest numbers should be used as the hundreds digit.

Then the next two largest numbers (3 and 4) should be used for the tens digit.
Consider 2 cases:

Case 1. 63? × 54?

Case 2. 64? × 53?

63?× 40, must be larger than 24000

64?× 30, must be smaller than 24000

Case 1 gives a larger product.

We can decide the placement for “1” and “2” using the same method.

Figure 2.6: (a) An example of interactive question. (b) The solution of (a).

To further reflect the problem-solving logic, engagement level, and difficulties students encountered and facilitate further visualization tasks, we use a hybrid state Markov Chain to model students’ problem-solving behaviors (Abate et al., 2008). Before introducing the detailed problem-solving behavior modeling, we first define the following terms as below:

- **Intermediate answer:** the answer a student fills in blanks at a step. For example, for the Fig 2.6a question, $6, null, null, null, null, null$ is an intermediate answer.
- **Condition:** one criteria that students need to fulfill to get part of the score. For example, in Fig 2.6a, placing one of the six digits in the correct position accounts for one condition being satisfied.

- **Condition array:** a string consisting of 0 and 1 to indicate whether a set of conditions are fulfilled or not. The length of the string is the number of the conditions. For example, the length of the question in Fig 2.6a is six.
- **Step:** the result of an action (e.g., deletion, insertion, exchange or translation) that a student makes to change his/her answer.
- **Stage:** the number of conditions the current answer fulfills. For example, for the Fig 2.6a question, $6, null, null, null, null, null$ fulfills one condition at Stage one.
- **Time elapse:** the time period between the time a student starts to solve the problem and the time he/she reaches a certain step.
- **Trajectory length:** the length (pixels) of the cursor moving on screen from the time student starts to solve the problem to the time he/she reaches a certain step.

Based on the above definitions, **State (S)** is modeled as a two-level hybrid structure:

- **Level1:** {Step, Stage} + {Condition array, Time elapse, Trajectory length}
- **Level2:** {Intermediate answer}

For a particular student, the change of the conditions satisfied along the way can reflect the problem-solving logic. If a student's stage cannot steadily go up as the step continues (e.g., the stage remains the same for several steps or drops continuously), it may indicate that the student encounters difficulties, as suggested by the domain experts.

Data-driven Feedback Construction

As required by domain experts, they need to evaluate the feasibility of using existing data to provide data-driven feedback to guide students' problem-solving process. When constructing data-driven feedback, the basic idea is to recommend the paths conducted by students who can correctly solve the question to students who cannot. We implemented a popular path recommendation algorithm based on the path similarity applied in previous research (Du et al., 2016) (Xia et al., 2019a). More specifically, for a student who submitted an incorrect answer, we find all the problem-solving paths (1) that fulfill all the conditions at the end and (2) whose states cover the final states resulting the incorrect answer. If more than one path satisfies this constraint, then we rank them according to their similarities with the current path. The similarity is calculated by the total steps, the total time, and the total trajectory length. Finally, we recommend the solution path with the highest similarity.

Visual Design

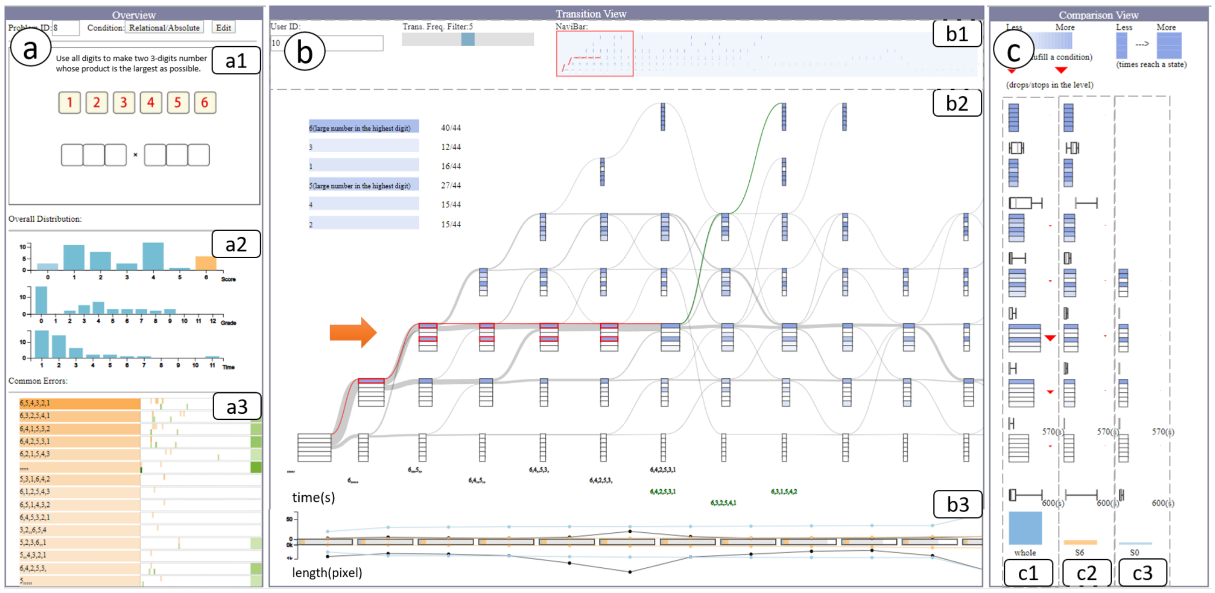


Figure 2.7: *QLens* enables question designers to analyze students' multi-step problem-solving behaviors for design improvements from levels of detail.

We present a novel visual analytics system *QLens* to accomplish the aforementioned tasks. This system can aid educators and question designers in exploring, analyzing, and understanding the problem-solving processes of students in multi-step questions. The visual analysis module of *QLens* incorporates the following three views. 1) **Overview**, as shown in Fig. 2.7a, displays basic information about the students, which includes the background of students, score distribution, time spent by students and information about common errors. 2) **Transition View**, as shown in Fig. 2.7b, intuitively visualizes the steps and stages involved in solving a problem which help in understanding the different approaches and problem-solving logic that the students apply in order to solve the problem. This view also displays the data-driven recommended paths of common errors for question designers to evaluate the feasibility for providing data-driven feedback. 3) **Comparison View**, as shown in Fig. 2.7c, facilitates a detailed comparison of the problem-solving process of two or more groups of students based on users' selection. Moreover, a collection of interactions, such as filtering, highlighting, and tooltips, is also available for the users to explore the dataset freely.

Transition View

The Transition View (Fig. 2.7b) aims to provide a holistic view of the multi-step problem-solving behavior to reflect students' problem-solving logic, engagement, and difficulties encountered and has three parts: control panel (Fig. 2.7b1), transition graph (Fig. 2.7b2), and engagement chart (Fig. 2.7b3).

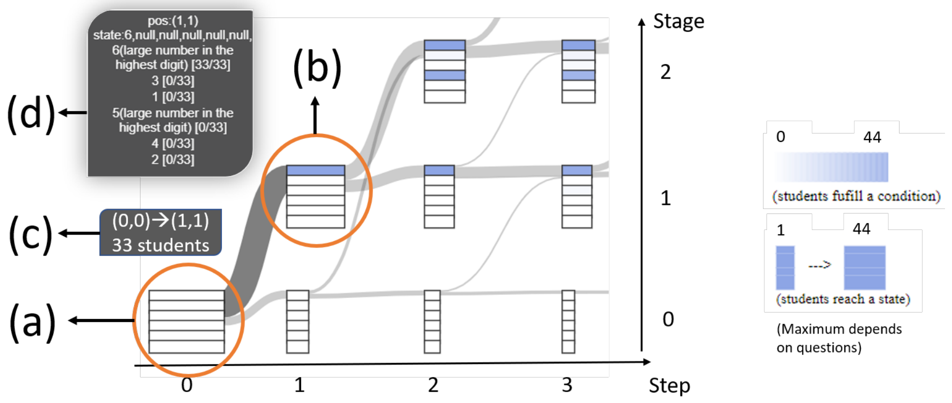

pos:(1,1)
state:6,null,null,null,null,null,
6(large number in the highest digit) [33/33]
3 [0/33]
1 [0/33]
5(large number in the highest digit) [0/33]
4 [0/33]
2 [0/33]
(d)
(b)
(c)
(0,0)→(1,1)
33 students
(a)
Stage
2
1
0
Step
0
1
2
3
0
44
(students fufill a condition)
1
44
--->
(students reach a state)
(Maximum depends on questions)

Figure 2.8: The enlarged part of the transition graph in Fig. 2.10b2

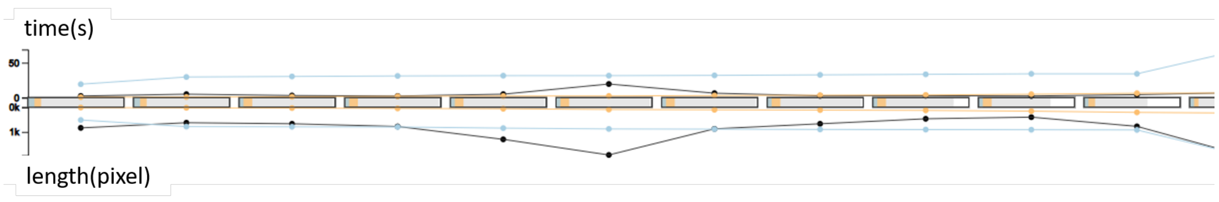


Figure 2.9: The contextual line chart with dual axes. The upper line chart shows the average time spent (seconds) on each step and the lower shows the trajectory length of the cursor (pixels) on each step. Lines of different colors represent different groups.

Control Panel, as shown in Fig. 2.7b1, provides filter options to facilitate exploration. The leftmost filter enables question designers to select a single student and analyze his/her problem-solving path. The middle filter filters the transitions less than a certain number and makes the common path appealing. It also incorporates a navigation bar to provide question designer an overall impression of the length of total steps students take by highlighting the current page using a red rectangle.

Transition graph, as shown in Fig. 2.8, intuitively visualizes how a group of students fulfill a set of conditions step by step in order to reach the final answer (i.e., the problem-solving logic). The x-axis represents the step and a student makes a step when he/she changes the answer by deleting/inserting/exchanging/translating element(s). The y-axis represents the stage, where the stage is the number of conditions fulfilled in each step. For example, if a student has a sequence starting with $(null, null, null, null, null, null)$, $(6, null, null, null, null, null)$, he/she moves from Fig. 2.8a (Step zero, Stage zero) to Fig. 2.8b (Step one, Stage one). By placing 6 in the correct position, one condition is fulfilled. A student moves to a higher stage only when one or more conditions are satisfied otherwise the students remains in the same stage or drops to a lower stage if he/she breaks one or more condition(s) no matter how many steps he/she makes. The transition lines between the glyph show the number of students going from one state (step, stage) to another. For example, as shown in Fig. 2.8a, all students fulfill zero conditions at the beginning. The 33 students (Fig. 2.8c) move from Stage zero (zero conditions fulfilled) to Stage one (one condition fulfilled) at the second step. By hovering the mouse to the glyph, we can check the details like the number of students fulfilled each condition (Fig. 2.8d).

Engagement Chart (Fig. 2.9) shows the efforts students pay on each step when solving the question. It consists of a dual-axis line chart (top and bottom). The upper line chart shows the average time spent in seconds on each step, and the lower line chart shows the trajectory length of the cursor in pixels in each step. This contextual line chart helps users in understanding how much effort the students make or how much they engage in solving the problem based on the time spent and the interactions. When one or more subgroups are selected, a line with a different color is added to show the average time and interaction in each step of that particular group. In Fig. 2.9, the light blue line represents students fulfilling zero conditions and the orange line represents students fulfilling all the six conditions. We can see that students got zero score actually exert more efforts for each step.

2.2.2 SeqDynamics: Problem-solving Sequence Analysis among Multiple Questions

Problem-solving dynamics refers to the process of solving a series of problems over time, from which a student's cognitive skills and non-cognitive traits and behaviors can be inferred. For example, we can derive a student's learning curve (an indicator of cognitive skill) from the changes in the difficulty level of problems solved, or derive a student's self-regulation patterns (an example of non-cognitive traits and behaviors) based on the problem-solving frequency over time. Few studies provide an integrated overview of both aspects by unfolding the problem-solving process. In this paper, we present a visual analytics system named *SeqDynamics* that evaluates students' problem-solving dynamics from both cognitive and non-cognitive perspectives. The system visualizes the chronological sequence of learners' problem-solving behavior through a set of novel visual designs and coordinated contextual views, enabling users to compare and evaluate problem-solving dynamics on multiple scales. We present three scenarios to demonstrate the usefulness of *SeqDynamics* on a real-world dataset which consists of thousands of problem-solving traces. We also conduct five expert interviews to show that *SeqDynamics* enhances domain experts' understanding of learning behavior sequences and assists them in completing evaluation tasks efficiently.

Establishing Problem-Solving Attributes

We establish problem-solving attributes from both the cognitive and non-cognitive perspectives. Based on previous research (Ausubel et al., 1968) and domain experts' suggestions, we consider three learning attributes to evaluate individual's cognitive skills: the number of problems solved (l_1), the percentage of hard problems solved (l_2), and the diversity of problems solved (l_3). In addition, we consider three learning attributes related to noncognitive traits and behaviors: diligence, willingness to take on challenges, and perseverance. Previous research (Farkas, 2003) summarized a series of non-cognitive skills from two aspects, conscientious work habits (e.g., perseverance, discipline) and other personality traits (e.g., opening to new experience, aggressive, etc). The roles of those skills in schooling and employment outcomes had been examined. We discussed these attributes one by one with domain experts to check how they are valued and measured in our scenario. We concluded that the total number of submissions (l_4) can reveal whether a student has expended effort on the programming platform, which we recognize as diligence. The time starting to solve hard problems (l_5) shows when a student has adapted to the learning process and is confident enough to try new problems. This indicates one's willingness to take on challenges and is highly emphasized by the experts when selecting the best candidates. l_6, calculated by #(active days)/ #(days after registration), uncovers a student's ability from the time perspective, which we called perseverance, to differentiate good candidates from other students who attempt many problems with enthusiasm only over a short period.

Problem Difficulty and Learner Ranking

According to domain experts (E1, E2), problem difficulty is a key feature in our context when evaluating and selecting learners, based on which the students' ranking is defined. The current method of determining the rank of a learner is purely based on the number of problems he/she solves (Revilla et al., 2008). However, a student who solves ten easy questions is different from a student who solves ten difficult ones. Inspired by the ELO ranking algorithm used in areas (e.g., chess, education, work allocation) to define the relative ranking dynamically (George Trimponias & Yang,) (Molenaar et al., 2019) (Goodspeed, 2017), we modify it to calculate the learner ranking. The detailed ranking method can be found in paper (Xia et al.,)

Visual Design

We present a novel visual analytics system for instructors to evaluate the problem-solving dynamics and select the best candidates from the candidate pool. The visual analysis

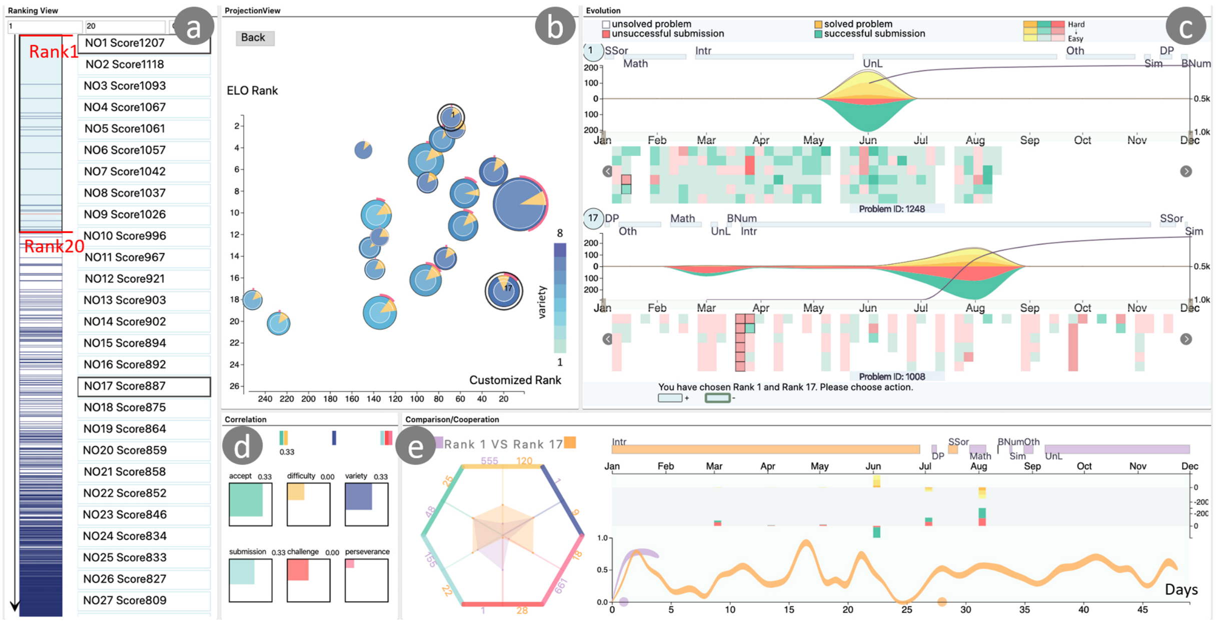


Figure 2.10: *SeqDynamics* facilitates the evaluation of learners' problem-solving dynamics via levels of analysis: (a) Ranking View displays the overall performance (macro level), (b) Projection View plots a subset of learners with key learning attributes (meso level), (c) Evolution View and (e) Comparison/Cooperation View magnify the details of a problem-solving sequence unfolded over time (micro level). Correlation View (d) displays the correlation of different learning attributes with the performance ranking and enables users to customize the weights.

module of *SeqDynamics* includes: 1) **Ranking View** (Fig. 2.10a) that displays an overall distribution of the learners' scores and ELO ranking in an ascending order (macro); 2) **Projection View** (Fig. 2.10b) and **Correlation Panel** (Fig. 2.10d) that facilitate instructors to customize their own evaluation criteria and compare the synoptic problem-solving features of a subset candidates on a 2D canvas. 3) **Evolution View** (Fig. 2.10c) that expands the selected learners' problem-solving features over time for detailed inspection (micro); 4) **Comparison/Cooperation View** (Fig. 2.10e) that facilitates explicit comparisons of two problem-solving sequences and demonstrates the complementarity of two learners. A collection of interactions, such as querying, highlighting, tooltips, and brushing, is also available for users to explore the dataset freely. More details can be referred in the paper (Xia et al.,).

2.3 Problem-solving Outcome Prediction

2.3.1 Predicting Student Performance in Interactive Online Question Pools Using Mouse Interaction Features

We propose a novel method by introducing a set of new features based on interactions between students and questions to predict student performance in interactive online question pools. The mouse movement trajectories, which consist of the mouse interaction timestamp, mouse event type and mouse coordinates, are basic factors in prediction and represent the problem-solving path of a student for each question. Inspired by prior researchers in education and psychology fields (Stahl, 1994; Row, 1974), which shows that student's "think" time on questions affect their performance, we propose a set of novel features (e.g., think time, first attempt, first drag-and-drop) based on mouse movement trajectories to predict student performance in interactive online question pools.

We define the time between when students see the question and the time that they start solving the questions as the "think time" and also extract attributes related to the first attempt and first drag-and-drop. Further, since there is no predefined question order in interactive online question pools, we apply a heterogeneous information network (HIN) to calculate the similarity among questions and incorporate students' problem-solving history to enhance performance prediction. We evaluated our approach on a real-world dataset that are collected from a K-12 interactive mathematical question pool Learnlex. We tested our new feature set on four typical multiple-classification machine learning models such as Logistic Regression (LR), Gradient Boosting Decision Tree (GBDT), Support Vector Machine (SVM) and Random Forest(RF).

Prediction Framework

We propose our prediction framework as Figure 2.11, which mainly contains three modules: data collection and preprocessing, feature extraction, and prediction and evaluation.

In the feature extraction module, we extract the historical performance statistical features and recent performance statistical features from the records. In addition, we summarize each question's basic information (e.g., grade, difficulty) and compute mouse movement trajectories to representation features including think-time, first drag-and-drop, first attempt, and other problem-solving related features. In addition, we combine these features and statistical features (e.g., score) and build a problem-solving HIN on them to calculate the question similarity matrix..

In the prediction and evaluation module, we test these features on four typical machine learning models to compare their performances with and without the mouse movement

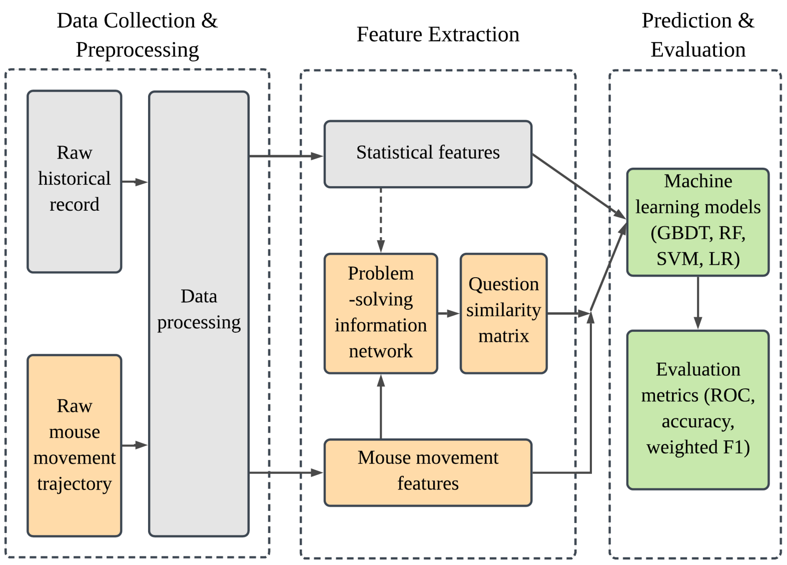

Data Collection & Preprocessing
Feature Extraction
Prediction & Evaluation
Raw historical record
Data processing
Raw mouse movement trajectory
Statistical features
Problem -solving information network
Question similarity matrix
Mouse movement features
Machine learning models (GBDT, RF, SVM, LR)
Evaluation metrics (ROC, accuracy, weighted F1)

Figure 2.11: The prediction framework

features. Similar to previous research (Gardner et al., 2019; ?), we compare prediction accuracy, weighted F1 and ROC curves and feature importance score in GBDT to evaluate and discuss our method.

Feature Table

We extract three sets of features: change points, TFF (think time, first attempt, and first drag-and-drop) and MDSM (mouse drag statistical measurements). We use change points, TFF and MDSM to model students' initial and overall behaviors in a problem-solving process, respectively.

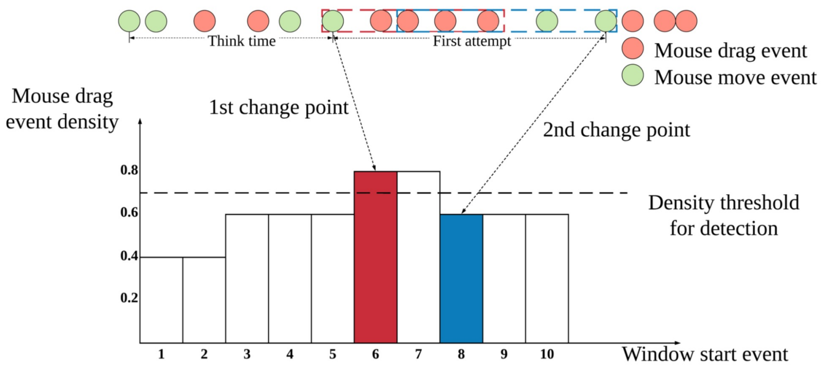


Figure 2.12: A sample mouse movement trajectory and the scheme diagram of change point detection algorithm. The 1st change point is both the think time end point and the first attempt start point. The 2nd change point is the first attempt end point.

The change points are the time points where there is an abrupt change in the mouse movement trajectory to distinguish different problem-solving stages in our context and

Table 2.1: TFF feature table. TFF: think time, first attempt, and first drag-and-drop.

Type	Feature	Description
Think time	Time length	Time between opening the web page and the first change point.
	Time percent	Percentage of think time in time length of the whole trajectory.
	Event length	Mouse event number in think time.
	Event percent	Percentage of mouse event number in the whole trajectory.
First attempt	Event end index	Mouse event number during the first attempt.
First drag-and-drop	Time length	Time between opening the web page to the first drag-and-drop.
	Time percent	Percentage of first drag-and-drop in time length of the whole trajectory.
	Event start index	Mouse event number before first drag-and-drop starts.
	Event percent	Percentage of event number before first drag-and-drop starts in the whole trajectory.
	Event end index	Mouse event number when first drag-and-drop ends.
	K	First drag-and-drop trajectory curvature.
	D	First drag-and-drop trajectory length.
	Delta	First drag-and-drop chord length.

they could infer students' problem-solving capabilities from the mouse movement trajectories. As shown in Figure 2.12, we split the mouse movement trajectory into three sub-parts using two change points: the think time stage, the first attempt stage (i.e., the first action episode after the thinking period) and the following actions stage.

The detailed attributes of the TFF are shown in Table 2.1. Drag-and-drops is a series of consecutive drag events that start with the mouse down and end with the mouse up and thus the first drag-and-drop is the first drag with a mouse down and a mouse up. More details can be referred in the paper (Wei et al., 2020).

As for MDSM, we define the following features to represent mouse drag statistical measurements as Figure 2.13 shows.

- D: Drag-and-drop trajectory length.
- $Delta$: Drag-and-drop chord length.

Table 2.2: Three statistical measures for six features. IQR: Interquartile range. D: Drag-and-drop trajectory length, $Delta$: Drag-and-drop chord length, K: Drag-and-drop trajectory curvature, T_{drag}: Drag-and-drop duration, T_{Idle}: Drag-and-drops interval and t_{Idle}: Mouse events interval.

	K	D	T_{drag}	$Delta$	t_{Idle}	T_{Idle}
Median	The feature's median value in its value list of the whole trajectory.					
IQR	The feature's IQR in its value list of the whole trajectory.					
Mean	The feature's mean value in its value list of the whole trajectory.					

- K: Drag-and-drop trajectory curvature (Delta/D).
- T_{Drag}: Drag-and-drop duration.
- T_{Idle}: Drag-and-drops interval.
- t_{Idle}: Mouse events interval.

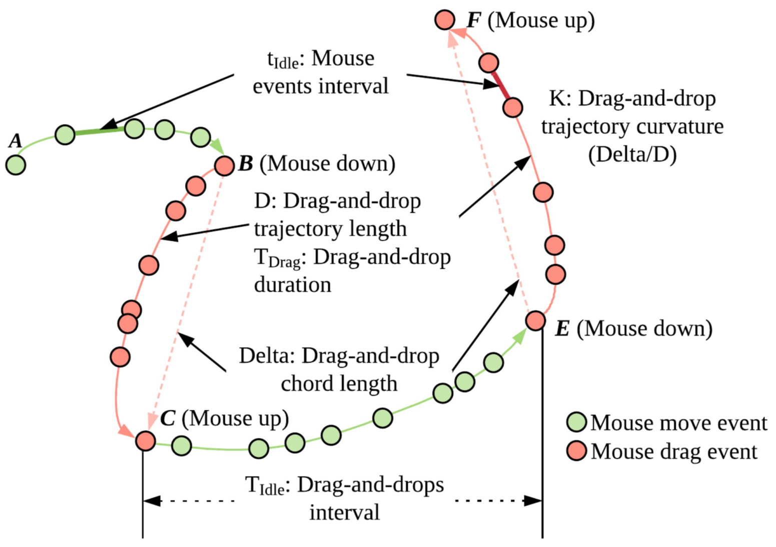

F (Mouse up)
t_{Idle}: Mouse events interval
K: Drag-and-drop trajectory curvature (Delta/D)
A
B (Mouse down)
D: Drag-and-drop trajectory length
T_{Drag}: Drag-and-drop duration
E (Mouse down)
Delta: Drag-and-drop chord length
C (Mouse up)
Mouse move event
Mouse drag event
T_{Idle}: Drag-and-drops interval

Figure 2.13: The example of defined features for MDSM.

Information such as score records and score distribution on questions can also reflect a student's ability and the question difficulty for all students. Features extracted from such records are widely applied in prediction of students' performance. Crossing combinations of features provide predictive abilities beyond what the features can provide individually (Crosses, 2019). Thus, we apply the cross feature method, which is a synthetic feature formed by multiplying (crossing) two or more features, to questions' and students' basic statistics. As Table 2.3 shows, we have three parts of statistical features: question statistics, student statistics, and the recent statistics of a student.

Table 2.3: Other statistical features: questions statistics, student statistics and student recent statistics. Symbol: #: Number of records, %: Proportion of records. Expression A $\times$ B: cross features of A and B. Expression C in [E]: calculate C for each category or dimension in E.

Feature	Description
Question statistics	
Math dimension	Question's domain knowledge (e.g, area).
Grade	Student's grade that the question suggests.
Difficulty	Question's difficulty given by experts.
#Total submissions	Total number of submissions in question.
#2nd submissions	Total number of second submissions in question.
%Submissions in [score class]	Proportion of submissions in each score class.
Student statistics	
#Total submissions	Student's total submissions in history.
#2nd submissions	Student 's total second submissions in history.
%Submissions in [math dimension $\times$ grade $\times$ difficulty]	Student's proportion of submissions in each specific math dimension, grade and difficulty.
1stAvgScore in [math dimension $\times$ grade $\times$ difficulty]	Student's first submission average score in each specific math dimension with assigned specific grade and difficulty.
Student recent statistics	
#Submissions in [math dimension]	Number of submissions in each math dimension in past N days.
#Submissions in [grade $\times$ difficulty]	Number of submissions in each grade and difficulty in past N days.
Average score in [math dimension]	Average score in each math dimension in past N days.
Average score in [grade $\times$ difficulty]	Average score in each grade and difficulty in past N days.
Score std in [math dimension]	Score standard deviation of each math dimension in past N days.
Score std in [grade $\times$ difficulty]	Score standard deviation of each grade and difficulty in past N days.

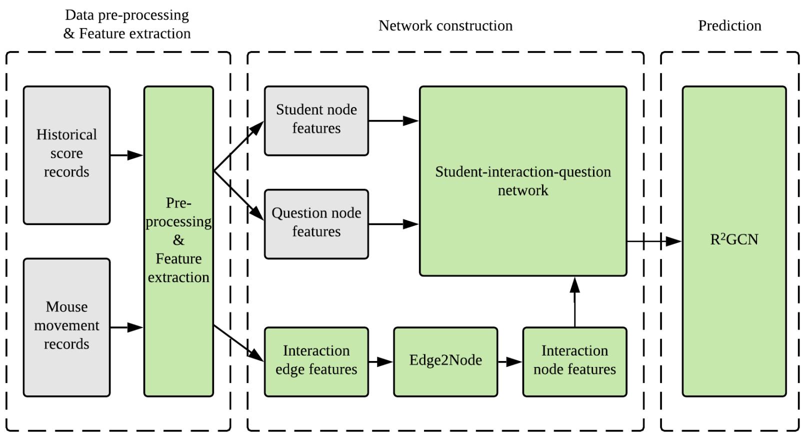

Data pre-processing & Feature extraction
Network construction
Prediction
Historical score records
Mouse movement records
Pre-processing & Feature extraction
Student node features
Question node features
Interaction edge features
Edge2Node
Interaction node features
Student-interaction-question network
R²GCN

Figure 2.14: The framework of the proposed method. The blocks highlighted in green are our major contributions.

2.3.2 Peer-inspired Student Performance Prediction with Graph Neural Network

Interactive online question pools, an essential part of online learning, attempts to make it a joyful process for students to practice their knowledge on a collection of interactive questions. However, since interactive online question pools are different from MOOC platforms and there is no predefined sequential order for the learning materials (i.e., questions) or accurate question knowledge labelling, former successful performance prediction methods designed for MOOCs are not suitable to be applied on interactive online question pools. Therefore, we are motivated by the crucial research question: how can we achieve effective student performance prediction in interactive online question pools?

We propose a peer-inspired approach for student performance prediction in interactive online question pools. It extensively considers the historical problem-solving records of both a student and his/her peers (i.e., other students working on the question pool) to better model the complex relationship among students, questions, and student performances and further enhance student performance prediction, which is achieved by using a GNN-based model. Figure 2.14 shows the framework of our approach, which consists of three major modules: data processing & feature extraction, network construction, and prediction. The module of *data processing & feature extraction* is designed to process the related data and extract features of the historical data that will be further used for network construction and student performance prediction. We considered three types of features: *statistical features of students* reflecting students' past performance, *statistical features of questions* indicating the question popularity and their average scores, and *mouse movement features* representing the characteristics of students' problem-solving behaviors. The *network construction* module builds a network consisting of both students and questions, where the interactions between them are also considered and further integrated into the network. This network also incorporates the three types of features to extensively model the various performance of different students on different questions. Finally, the constructed network, along with the extracted features, is input into the *prediction* module, where we propose R^{2}GCN, a novel Residual Relational Graph Neural Network model that is adapted from R-GCN (Schlichtkrull et al., 2018) by adding residual connections to hidden states, to predict a student's score level on the unattempted questions in interactive online question pools.

Our peer-inspired performance prediction method could provide the interactive online question pools with a accurate prediction result and benefit lots of downstream tasks including adaptive learning path construction, personal question recommendation and students retention rate improvement. For more details, please refer to our paper (Li et al., 2020).

2.4 Learning Path Planning

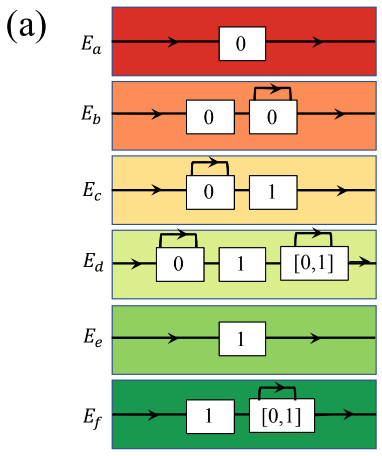
Online question pools like LeetCode provide hands-on exercises of skills and knowledge. However, due to the large volume of questions and the intent of hiding the tested knowledge behind them, many users find it hard to decide where to start or how to proceed based on their goals and performance. To overcome these limitations, we present PeerLens, an interactive visual analysis system that enables peer-inspired learning path planning. PeerLens can recommend a customized, adaptable sequence of practice questions to individual learners, based on the exercise history of other users in a similar learning scenario. We propose a new way to model the learning path by submission types and a novel visual design to facilitate the understanding and planning of the learning path. We conducted a within-subject experiment to assess the efficacy and usefulness of PeerLens in comparison with two baseline systems. Experiment results show that users are more confident in arranging their learning path via PeerLens and find it more informative and intuitive.

2.4.1 Path Planning Engine

The learning path planning engine models the learning path, groups similar learning paths according to four important attributes and further forms learning path suggestions.

Learning Path Modeling

We use *submission type* to describe how a learner solves a specific problem P_i. Figure 2.15 shows the six submission types $E = \{E_a, E_b, E_c, E_d, E_e, E_f\}$ defined in this paper, where E_a denotes one failed attempt without success, E_b denotes multiple failed attempts without success, E_c denotes multiple failed attempts followed by one success, E_d denotes



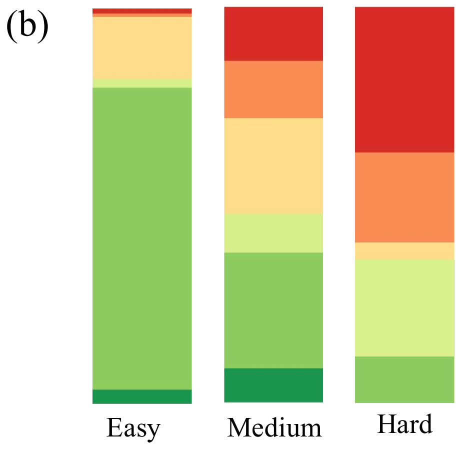


Figure 2.15: Submission types and their distribution: (a) the six submission types: 1 means *solved*, 0 otherwise; (b) submission type distributions of *Easy*, *Medium* and *Hard* questions.

multiple failed attempts followed by one success and more attempts, E_e denotes one success without further attempts, E_f denotes one success followed by more attempts.

Peer Learning Path Grouping

The grouping of peer learning paths is done in three steps. First, following prior studies (Shephard, 1968; Burnett et al., 2007),we choose four attributes to group learning paths of learners in question pools. They are learning duration (the time span between the first submission event and the last submission event), learning frequency (how often a submission event appears), learning intensity (the number of submission events per day) and learning proficiency ($(E_e + E_f)/\#\{\text{submission events}\}$). Second, we plot histogram overviews to inspect the user distribution along each attribute. Third, domain experts are involved to specify meaningful ranges for each attribute based on the histograms. Two factors are considered in this process: user number within each range and behavior differences between ranges. The grouping results are shown in Table 2.4. Combining these attributes, we further extract three typical learning scenarios (i.e., regular learning, intensive learning and, advanced learning), as shown in Figure 2.10(a). This grouping can be reused if the size of the new data is relatively small to that of the existing dataset and when the value distribution is largely unaffected. If the distribution of an attribute has changed considerably or new scenarios are introduced, we rerun the second and third step to update the grouping.

Path Suggestion

We achieve learning path planning based on Markov Chain (MC), as it is more intuitive for human beings to understand. Specifically, we define the state s as a set of problems which have been solved (Piech et al., 2015; ?), e.g., $s = \{X_0, X_1, ..., X_n\}$. Note that we do

Table 2.4: The four performance-based attributes are empirically divided into four ranges.

	Range 1	Range 2	Range 3	Range 4
Duration(months)	0~1	1~3	3~6	>=6
Frequency	0~0.1	0.1~0.2	0.2~0.3	>=0.3
Intensity	0~1	1~2	2~5	>=5
Proficiency	0~0.25	0.25~0.5	0.5~0.75	>=0.75

not consider the order in which the problems are solved. Based on this definition, a given peer path $[(X_{i_0}, E_{i_0}, t_{i_0}), ..., (X_{i_n}, E_{i_n}, t_{i_n})]$ corresponds to a state $s = \{X_{i_0}, X_{i_1}, ..., X_{i_n}\}$. State s_i transits to State s_j only when $s_j = s_i \cup X_k$, where X_k is the extra problem in s_j compared with s_i. To generate the component $P_{ss'}$ in the transition matrix P, we count the number of transitions $N_{ss'}$ from s to s' and the number of all transitions N_s from s within the given peer paths, which are included in the peer group selected by the learner. Then $P_{ss'}$ is defined as the ratio of $N_{ss'}$ to N_s. This transition probability matrix P captures the common behaviors of problem solving in the peer group.

Based on this P, we can plan learning paths for a given learner. The most natural path is the most popular path taken by the selected peer group. Given the learner's history path, we first find the corresponding state s_u, and then query P to find the state transition from s_u with the highest possibility. This query is conducted recursively until a path of a certain length is found. This path is then recommended to the learner as the **popular path**.

We also derive two variants from the popular path to meet different learners' needs and characteristics. The first variant is the **challenging path**, which is generated by skipping problems of similar difficulty. To this end, we use cosine similarity to measure the similarity of submission type distributions among the consecutive problems in the popular path and select only one problem from cosecutive and similar problems. Another path variant is the **progressive path**. We reorder the problems from the popular path based on the problem's difficulty level from easiest to hardest, which is also inferred by submission type distributions.

2.4.2 Visual Design

Design The interface of *PeerLens* is composed of three coordinated views: the peer selection view (Figure 2.10(a)), the learning path view (Figure 2.10(b)), and the problem archive view (Figure 2.10(c)).

Peer Selection View As shown in Figure 2.10(a), the peer selection view further consists of four radar charts arranged horizontally. The left three charts indicate peer groups in regular, intensive, and advanced learning scenarios, separately. Meanwhile, the rightmost chart allows learners to manually customize their learning scenarios.

Learning Path View

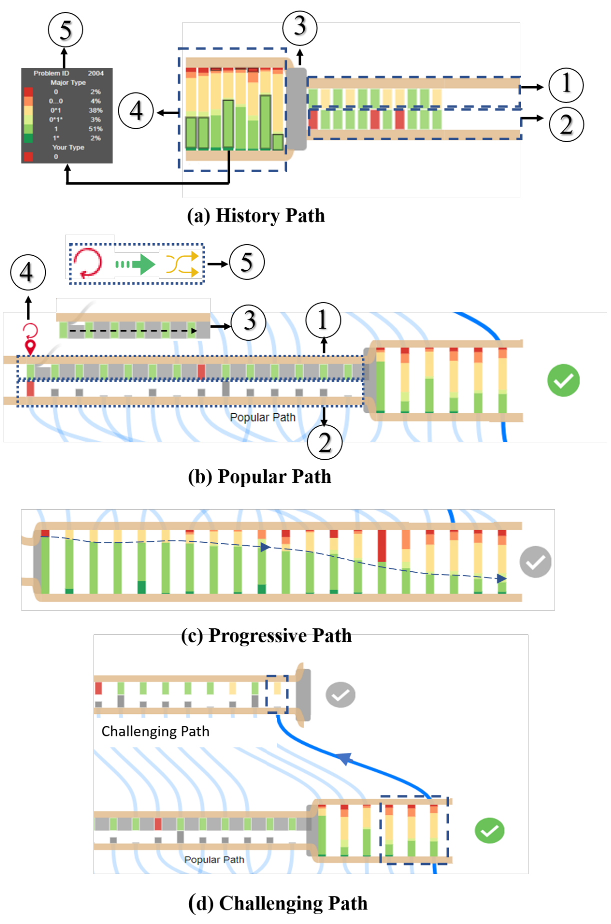


Figure 2.16: Learning path view. (a) History path, (b) Popular path, (c) Progressive path, and (d) Challenging path.

The learning path view compares the learner's learning path with those of the selected peer group (Requirement R2), and offers diverse learning path suggestions (Requirement R3). In this view, a zipper-like visual metaphor is proposed as part of the main design to help learners understand the context of one's learning path by answering "where have I came from?", "where am I?", and "where do I go?". As shown in Figure 2.10(b), the history path on the left displays which questions have been tried by the learner (Figure 2.10(b1)), and the future path on the right displays three learning paths suggested by the system (Figure 2.10(b2, b3, b4)). A location marker in the middle indicates the question currently being worked on (Figure 2.10(b5)). In all these paths, each question is represented by a tooth on the zipper design.

The history path in Figure 2.16(a) is made up of three components: the array of upper teeth 2.16(a-1), the array of lower teeth 2.16(a-2), and the slider 2.16(a-3). The upper teeth represent the performance of the selected group. The lower teeth represent current learner's performance on each question. Both upper and lower teeth are arranged in chronological order. To prevent visual clutter, we make use of the slider in the middle to control the amount of information displayed in this view. The slider is placed in the leftmost position by default when the zipper is closed. Each tooth in the lower array represents the submission type of the learner towards a specific question. Each tooth in the upper array shows the major submission type of the selected group in completing that question. In case the learner wants to see the detailed group performance, s/he can drag the slider rightward to open the zipper. The opened part of the teeth will then show the distribution of submission types by the stacked bar chart Figure 2.16(a-4). The learner's submission type is highlighted on the bar to show his/her position in the group. Submission types are encoded using a sequential color scheme from red to green in Figure 2.15 to indicate different submission types because we consider that different submission types can differentiate a good performance from a bad performance. The detailed information of the submission types can be assessed when hovering over each bar in the array (Figure 2.16(a-5)). The problem ID, the distribution of the submission type of each question and the learner's submission type will be given.

The future path displays three suggested learning paths: the challenging path (Figure 2.10(b2)), the popular path (Figure 2.10(b3)), and the progressive path (Figure 2.10(b4)). When the zipper is closed, as shown in Figure 2.16(b-1), the array of upper teeth encodes the selected learning group's major submission type on each question while the array of lower teeth shows the predicted difficulty of each question for the learner. To avoid overusing colors, we use the height of the grey bar to show the difficulty Figure 2.16(b-2). The higher the grey bar, the more difficult the question.

We also design visual cues to reason each path, and help learners understand the difference among the three suggested paths. On the popular path, we use the flow inside the path to show the probability of taking the next question in the selected group (Figure 2.16(b-3)). The branch of the flow indicates the minor number of people going on to do the other questions. Moreover, we use the lines to link the same question on different paths which helps to reason the challenging path and the progressive path which are derived from the popular path. For the progressive path (Figure 2.16(c)), learners see the detailed distribution of problems on the progressive path showing the growing difficulty from the easiest to the hardest, in accordance with the reordering of problems in Figure 2.16(b), in terms of difficulty level. The problems on challenging path only link to some of the problems on the popular path. By referring to Figure 2.16(d), learners can find that only one problem is chosen from three concecutive problems with similar submission distributions (difficulty).

When a learner selects a path, a location marker appears on that path, indicating the problem that the learner is solving. A hint is also shown above it, as depicted in Figure 2.16(b-4). There are three types of hints: “do this problem again”, “move to the next problem”, “change a path”, as shown in Figure 2.16(b-5). The suggested path will be updated when the learner changes path or customizes a new learning scenario.

Problem Archive View

The problem archive view in Figure 2.10(c) is designed to allow learners to quickly map the questions on the learning path with the original question in the pool. When hovering over any bar on the learning path, the corresponding question will be highlighted on the problem list. Learners can click the question on question list to enter original question page. The previous records and hints are shown on the left-hand side of each problem. More details can be referred in the paper (Xia et al., 2019a).

2.5 Learning Behavior Regulation

“Gaming the system” is the phenomenon where students attempt to perform well by systematically exploiting properties of the learning system, rather than learning the material. Frequent gaming tends to cause bad learning outcomes. Though existing studies tackle the problem by redesigning the system workflow to change students’ behaviors *automatically*, gaming students discover new ways to game. We instead propose a novel way, reflective nudge, to *reflectively* influence students’ attitudes by conveying reasons not to game via information visualizations. Particularly, we identify three common gaming contexts and involve students and instructors in co-designing three context-specific persuasive visualizations. We deploy our information visualizations in a real online learning platform. Through embedded surveys and in-person interviews, we find some evidence that the designs can promote students’ reflection on gaming, and suggestive data that two of them can reduce gaming compared with control groups. Furthermore, we present insights into reflective nudge designs and practical issues concerning deployment.

2.5.1 Visualization Designs

V1 (Fig. 2.17) is designed to convey that gaming the system through frequent submissions may result in spending a much longer time reviewing questions before an exam (G1). There are three rows in this visualization. The first row describes how good students spend time on this question, the second row shows how struggling students spend time on this question, and the third row is the student’s own submission sequence. Each bar on the row is a submission, whose colors, red and green, represent incorrect and correct submission, respectively. Yellow blocks mean the reflection time while blue ones represent

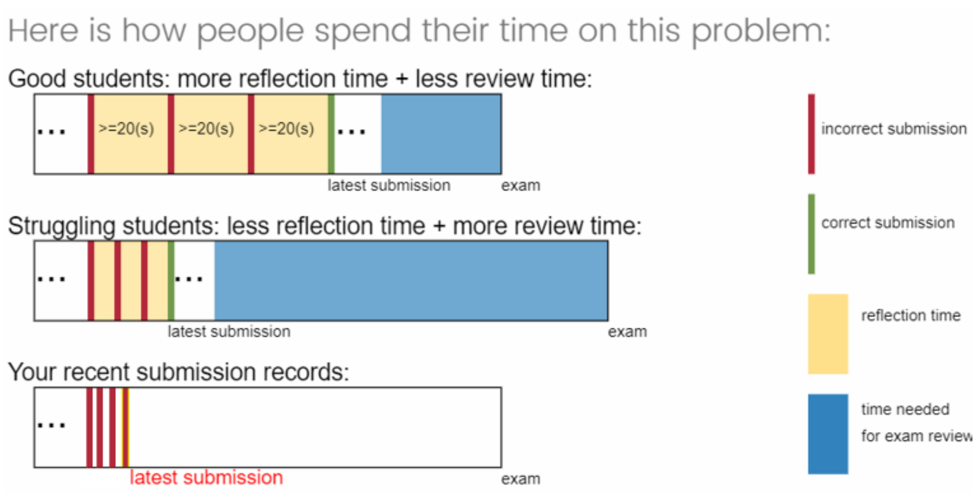

Here is how people spend their time on this problem:
Good students: more reflection time + less review time:
>=20(s)
>=20(s)
>=20(s)
latest submission
exam
Struggling students: less reflection time + more review time:
latest submission
exam
Your recent submission records:
latest submission
exam
incorrect submission
correct submission
reflection time
time needed for exam review

Figure 2.17: V1 shows how good students and struggling students spend their time on solving the problem and reviewing before the exam in order to convey gaming the system by submitting frequently may need to spend much more time reviewing the question before an exam.

the reviewing time before the exam. To achieve G2, first, by comparing the second row with the first row, we see that the yellow block is smaller between each bar while the blue trunk is much longer, by which we want to enhance the persuasiveness on spending more time between two submissions. Second, students can compare his/her records with the good students and struggling students to remind themselves of reflection instead of gaming. As shown in Fig. 2.17, this student has four recent submissions, and the time interval between each submission is short. Therefore, he/she tends to be similar to a struggling student, which may make it possible to persuade him/her to spend a longer time on reflection. We try to make it intuitive by using rectangles and basic colors (G3).

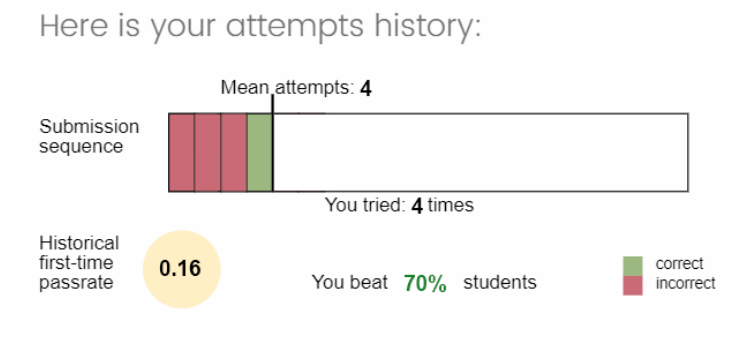


Figure 2.18: V2 shows the attempt times of peers by mean attempts and first-time pass rate from last semester in order to convey that difficult problems take considerable effort for other students to solve and that it is unnecessary to give up quickly by resorting to gaming the system.

V2 (Fig. 2.18) is designed to convey that difficult problems take considerable effort for other students to solve and that it is unproductive to give up quickly by resorting to gaming the system (G1). According to instructors' suggestions, we use mean attempts and the first-time pass rate to indicate the difficulty level of this problem. Additionally, we use "beat" how many percentages of students to make this design more persuasive. We use the red and green bars to represent incorrect and correct submissions, respectively (G3). To resolve the problem that no first-time pass rate exists when no one submits their answer in the exercise, we use the data of this exercise from the last semester. E.g., in Fig. 2.18, the historical first-time rate is only 0.16, and the number of mean attempts is four, indicating that this problem is quite difficult. Although this student solved the problem on the fourth attempt, he/she is better than or as good as 70% students with more trials. In addition, if the student tries to guess the correct answer by submitting many times, he/she will beat fewer students, which persuades him/her to think carefully about being competitive in the class (G2).

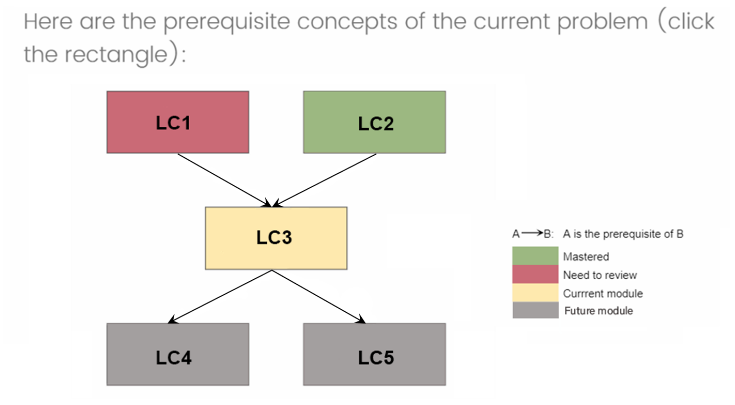

Here are the prerequisite concepts of the current problem (click the rectangle):
LC1
LC2
LC3
LC4
LC5
A⟶B: A is the prerequisite of B
Mastered
Need to review
Currrent module
Future module

Figure 2.19: V3 shows one example of the prerequisite relationship among learning concepts in order to convey that learning concepts are connected and that the negligence of one concept may hinder the mastery of later concepts due to the cumulative nature of the course.

V3(Fig 2.19) is designed to convey that learning concepts are connected and that the negligence of one concept may hinder the mastery of later concepts due to the cumulative nature of the course (G1). Each rectangle represents a learning concept, and the arrow from A to B means A is the prerequisite of B. We constructed the concept hierarchy based on the prerequisites relationship labeled by the instructors for each question. The background color of the rectangle represents how much the student has mastered the learning concept. The green one means students' average score on that learning concept is high enough, indicating the student has mastered the concept, while the red one means that the corresponding learning concept needs to be reviewed. Furthermore, the yellow one means the current learning module, and the grey ones require an understanding of the current learning concept. The reason we use this structure is that existing research reported that the tree structure is easy to understand (Long et al., 2017) (G3). We attempted to use this concept graph to show the connections between learning concepts and the importance of the current one. We also use red rectangles to persuade students to review previous learning concepts for solving the current problem if they can not solve them correctly (G2). These rectangles are clickable and can direct students to the corresponding pages.

CHAPTER 3

EVALUATION

3.1 Data Collection

3.1.1 Mouse Movement Data Collection

The interaction data, such as the trajectories, the drag and click actions of the mouse, are collected and visualized as interaction heat map. It virtually shows the elements of the lecture materials which the users pay more attention to.

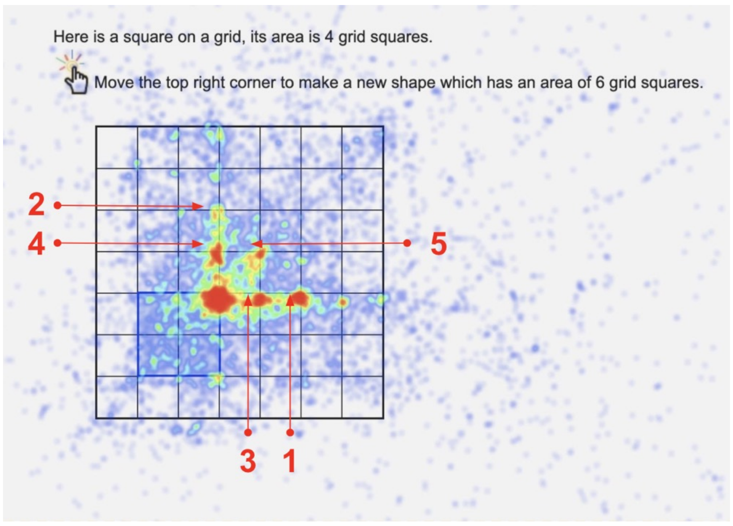

Here is a square on a grid, its area is 4 grid squares.
Move the top right corner to make a new shape which has an area of 6 grid squares.
2
4
5
3
1

Figure 3.1: Mouse movement data format

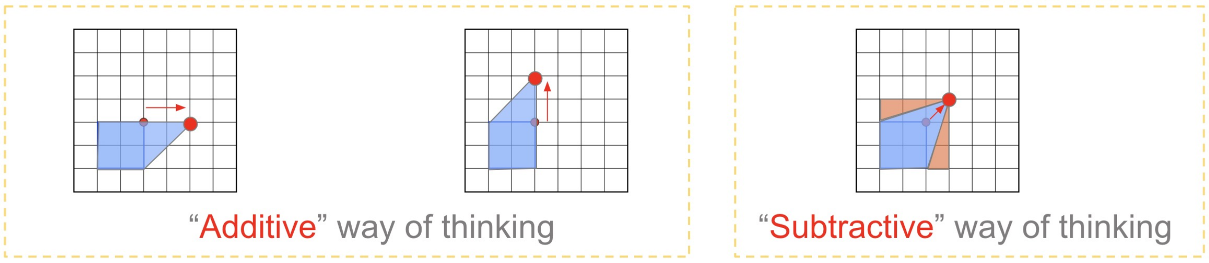


Figure 3.2: Mouse movement data format

- Many students got the correct answer (Points 1, 2 and 5).
- More students preferred to move the point horizontally than vertically (Points 1,3 vs 2,4).
- Some students did solve the problem in a "subtractive" way.

3.1.2 Grading Data Collection

The average score of each question could imply the questions' difficulty levels. It helps improve the reasonability of question assignment. For example, the difficulty level of the bottom left questions may not be reasonable.

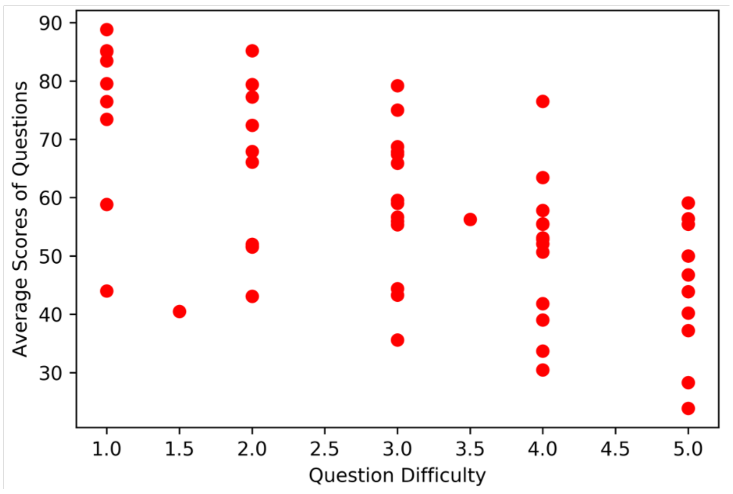


Figure 3.3: Mouse movement data format

The average scores in different login periods could reflect the students' study patterns and efficiency. A bursting number of students study on the E-learning platform around lunch time, but their performance drops.

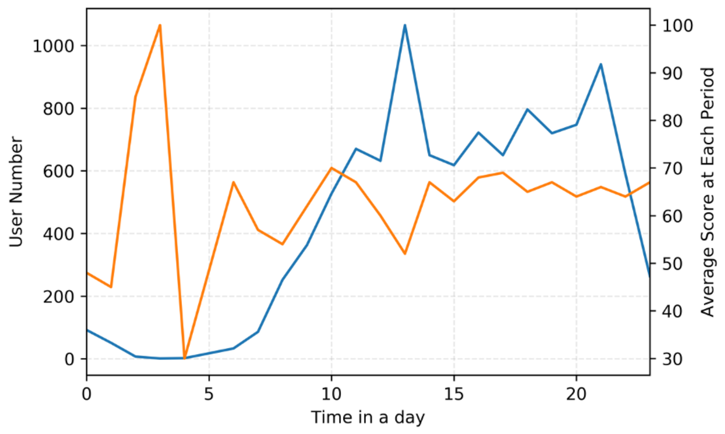


Figure 3.4: Mouse movement data format

Number change of login students in different login days in a week reflects the syllabus schedule. For example, the distribution of user numbers on Thursday implies an assignment deadline. Students are often more active in the afternoon (12pm-6pm) and evening (6pm-12am)

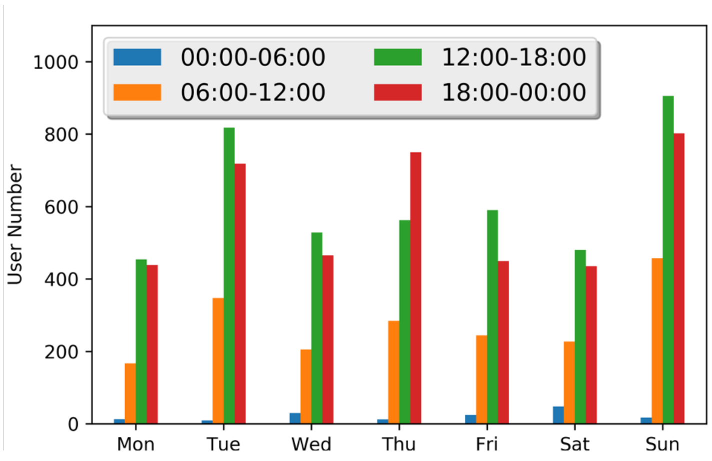


Figure 3.5: Mouse movement data format

3.2 Case Study of Problem-solving Sequence Analysis

3.2.1 QLens

To evaluate the visual analytics system, QLens, as introduced in Section 2.2.1, we use one case to describe how it can help question designers inspect students' problem-solving behaviors and check whether students' problem-solving logic matches the question designer's design intention. The design intention referred here is reflected in the step-wise solution provided by the question designer.

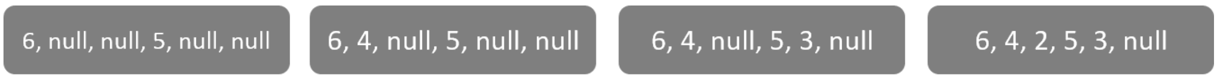


Figure 3.6: A sequence of popular answers.

The question in Fig. 2.10a1 asks students to position the six digits in a way such that the result of its product is as large as possible. E3 thought students may first figure out putting the largest two numbers in the hundred digits and estimated that students may encounter difficulties when dealing with the tenth digits, which was inferred from the post-question solution as shown in Fig. 2.6b. Then, E3 checked the Transition View (Fig. 2.10b2) to see whether students' practical behaviors met their design intention. He noticed a thick line in the middle of the Sankey diagram as indicated by the orange arrow. This thick line shows that most students got stuck at Stage two and cannot move further to higher stages (i.e., fulfilling more conditions) even after making several steps. Further, by checking the condition glyphs on Stage two, E3 saw that the first and fourth rectangles have darker shade than other rectangles, which represents that these two conditions are fulfilled by more students than other conditions. Referring to the condition description in the top left corner of Fig. 2.10b2, E3 confirmed that the students were clear that the first step is to position the largest numbers (5 and 6) to the hundredths place. He further checked the detailed information shown in the tooltip (Fig. 3.6) when hovering the cursor on each step of Stage two. E3 found that the most popular answers given by the students are as follows: $6, null, null, 5, null, null$, $6, 4, null, 5, null, null$, $6, 4, null, 5, 3, null$, $6, 4, 2, 5, 3, null$. It verifies that students were indeed confused in deciding the number for tenths place. In this case, we can safely conclude that the design intention is consistent with students' problem-solving behaviours.

3.2.2 SeqDynamics

To evaluate the visual analytics system, SeqDynamics, as introduced in Section 2.2.2, we describe one usage scenario, elite selection and analysis. Oliver, the coach for a competitive programming team who wanted to enlist three highly skilled individuals into his team's training camp from a pool of 1000 applicants. In the past, Oliver selected people by holding an examination. Nonetheless, he worries that individuals who perform poorly in the examination may still have great potential. Thus, he decides to use *SeqDynamics* to better examine individual students by evaluating their problem-solving history on the online judge.

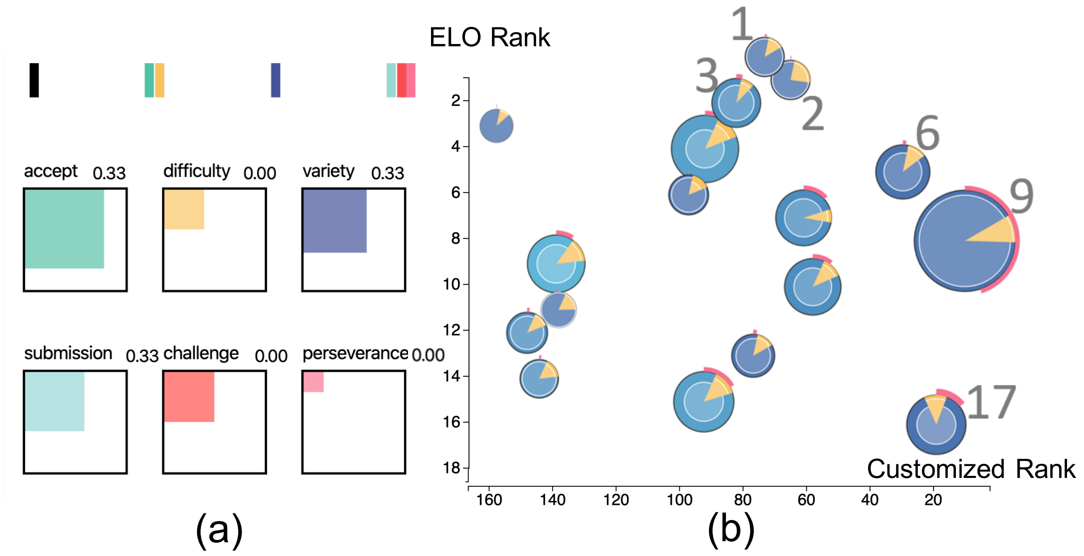


Figure 3.7: Enlarged subgraphs of Fig. 2.10. (a) Correlation Panel. (b) Enlarged section of the projection view in Fig. 2.10b.

Oliver first loaded all 1000 applicants into the system and found the ranking distribution sorted by their relative ELO scores (Fig. 2.10a). He then narrowed down to the top 20 applicants, since the other candidates had relatively lower scores, and then checked the Correlation Panel to see which attributes were positively correlated to the ELO ranking. He discovered that the accept, the submissions, and the variety had larger portions of the inner squares (Fig. 3.7a). By equally distributing the weights of the three features to one third each, the new customized ranking was generated, based on which he selected candidates with high potential. He then inspected the Projection View (Fig. 2.10b) for more detailed information and enlarged the top right-hand corner to get Fig. 3.7b, to see the students who topped both the ELO ranking and his customized ranking.

Oliver noticed the upmost three candidates, 1, 2, and 3, who were the top scorers calculated by ELO ranking and the rightmost three candidates, 6, 9, 17, who ranked top in the customized ranking.

From the first glance at the six candidate glyphs (Fig. 3.7b), 9 was much bigger (both inner and outer circle) than the others, indicating 9 had tried and solved more questions. Therefore 9 was selected. Then Oliver discovered 3 was more brightly colored and there was a relatively smaller yellow sector, implying that 3 had attempted less problem types and overall simpler questions compared to the rest of the candidates, and therefore could be eliminated.

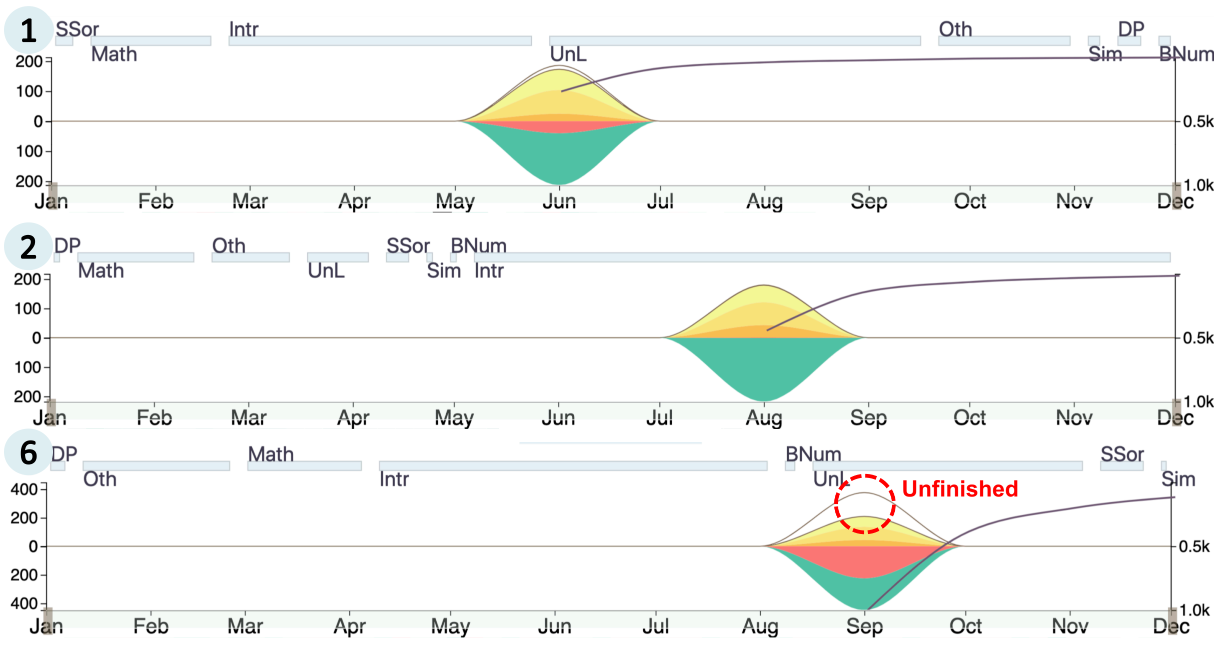


Figure 3.8: The bilateral stacked graphs of learners ranked 1, 2, 6.

Afterwards, Oliver inserted 1, 2, 6, 17 into the Evolution View (Fig. 3.8) to facilitate greater in-depth understanding and comparison. This view specifically allowed him to vividly visualize each candidate's progressive performance and behaviors. He found that 1, 2, 6 were fast learners and performed outstandingly within a short span. However, among them, 6 had an excessive amount of unfinished questions and was thus eliminated. By inspecting the downstream of the bilateral stacked graphs of 1 and 2, he found that 2 had no red branch while 1 had a distinct red branch, implying that 2 solved problems with a comparatively higher passing rate. Thus, Oliver selected 2 into the team.

With 2 and 9 chosen, Oliver needed to compare 1 and 17 to determine the allocation of the final position in his team of 3. Through the Comparison View of 1 and 17 (Fig. 3.9a), 1 tried more types of problems than 17.

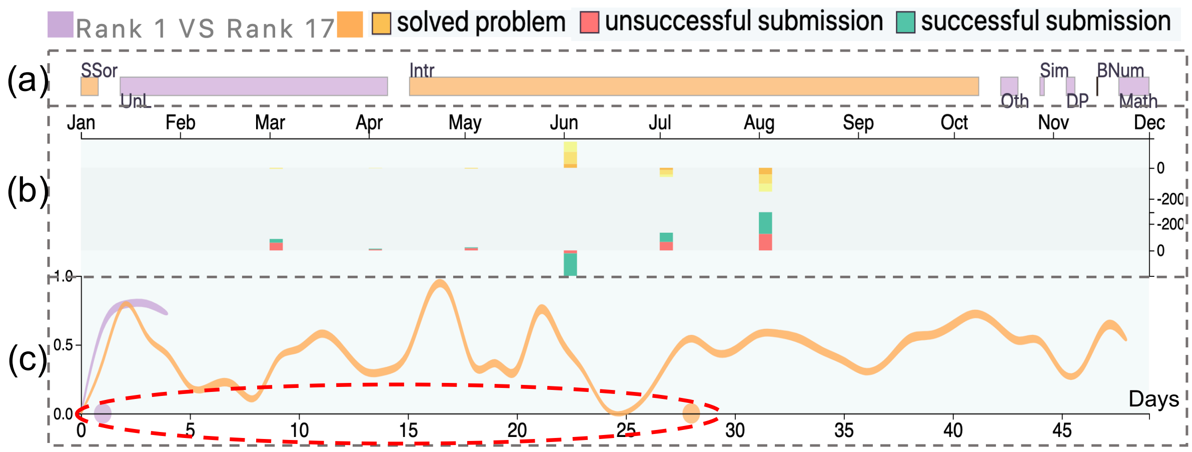


Figure 3.9: Comparison of 1vs17.

Also, he discovered that 17 started to attempt difficult problems 28 days later than 1 when aligning by start time as seen by the dot near the bottom of Fig. ??c. He then concluded that 17 was less unwilling to step out of his comfort zone and hence selected 1 as the final candidate to enlist. Overall, Oliver successfully selected three candidates, 1, 2, 9, through comprehensive user-driven evaluations.

3.3 Quantitative Evaluation of Student Performance Prediction

3.3.1 Predicting Student Performance in Interactive Online Question Pools Using Mouse Interaction Features

To evaluate the method proposed in Section 2.3.1, predicting student performance in interactive online question pools using mouse interaction features. We setup an experiment and compare how these features improve the prediction performance under four different traditional machine learning methods.

Experiment Setup

To make a 4-class classification, we applied four classical multi-class classification machine learning models, i.e., GBDT, RF, SVM, and LR, on our datasets. The features listed in Table 2.3 served as features in baseline method. Our experiment was conducted on the two datasets introduced in Table 3.1.

Table 3.1: AUC and ABROCA value in two datasets. ADD: area dimension question dataset. DGDD: deductive geometry dimension question dataset. Ours: our proposed method. ABROCA: Area between baseline curve and Ours curve.

Dataset	Method	GBDT	RF	SVM	LR
ADD	Ours	0.88	0.87	0.85	0.87
	baseline	0.79	0.85	0.77	0.83
	ABROCA	**0.09**	**0.02**	**0.08**	**0.04**
DGDD	Ours	0.94	0.90	0.91	0.91
	baseline	0.88	0.88	0.89	0.89
	ABROCA	**0.06**	**0.02**	**0.02**	**0.02**

Based on the proposed mouse movement features, similarity matrix M_{sim} of each dataset was extracted and then for each first submission $q_x - s_i$ in the dataset, we searched M_{sim} to find question q_y that was most similar to q_x and was solved by the same student s_m. If there was no similar question with q_x done by s_i, this submission record would be discarded. Finally, the feature vector of $q_x - s_i$ in our proposed method was composed of q_x's and s_i's historical statistical features, s_i's recent performance feature in Table 2.3, q_y's mouse movement features including features in Table 2.1 and Table 2.2, q_y's score class and the similarity score between q_x and q_y.

Because the dataset of proposed method for training and testing is not of the same size as the original dataset, we used the same submission records in the baseline and the proposed methods. In addition, to reduce the effect of imbalanced class distribution, we applied SMOTE over-sampling algorithm to increase the size of minority classes in the training set (Chawla et al., 2002). As for hyperparameter tuning in the four algorithms, grid-search was conducted in model training on our datasets (Manrique et al., 2019). The following parameters have been tested with the best values in bold:

- Number of trees for GBDT and RF: 50, 100, 150, 200, **250**, 300, 350
- Max depth of trees for GBDT and RF: **5**, 10, 15, 20, 25
- Learning rate of GBDT: 1e-4, **1e-3**, 1e-2, 5e-2, 0.1, 0.2
- Penalty parameter of SVM (C): 0.1, 1, **5**, 10

After fixing our hyperparameters, each model performed the task of predicting students' performance on two datasets using the baseline method and our proposed method for 10 times each and the average accuracy scores and weighted F1 scores are in Table 3.2.

Table 3.2: Results of the accuracy and weighted F1 over four typical machine learning algorithms (GBDT, RF, SVM, and LR) on the proposed method and the baseline method. ADD: area dimension question dataset. DGDD: deductive geometry dimension question dataset. Ours: our proposed method.

Dataset	Method	GBDT		RF		SVM		LR	
		Accuracy	Weighted F1	Accuracy	Weighted F1	Accuracy	Weighted F1	Accuracy	Weighted F1
ADD	Baseline	0.555	0.555	0.669	0.659	0.430	0.492	**0.650**	**0.670**
	Ours	**0.753**	**0.749**	**0.690**	**0.667**	**0.650**	**0.677**	0.649	0.659
DGDD	Baseline	0.600	0.597	0.664	0.663	0.600	0.611	0.720	0.731
	Ours	**0.833**	**0.805**	**0.780**	**0.767**	**0.633**	**0.643**	**0.733**	**0.744**

Performance Comparison

The results for our dataset ADD and DGDD are in Table 3.2. From the perspectives of accuracy and weighted F1 scores, we can see that in both two datasets, GBDT, RF, and SVM performed better in our proposed method than in the baseline.

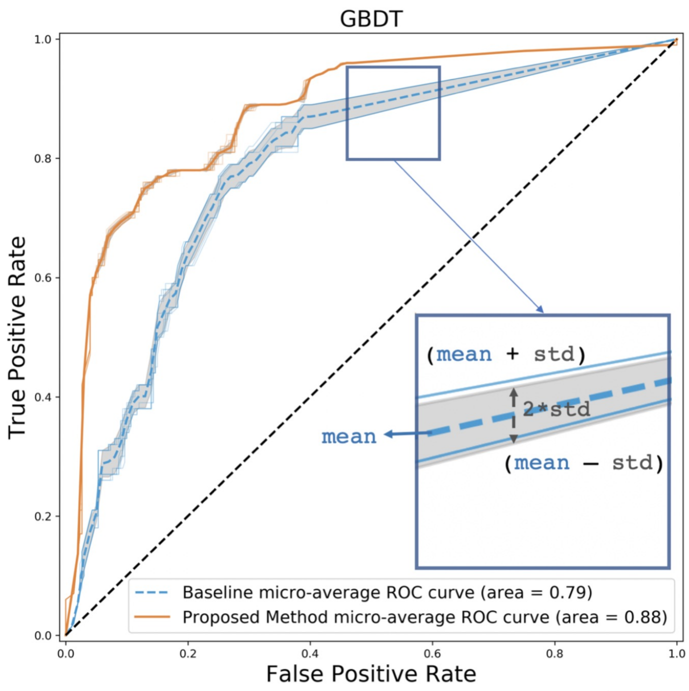


Figure 3.10: ROC-AUC curve of two methods on ADD (Area dimension question dataset).Gray borders represent mean+std or mean-std. Area between the two curves is ABROCA.

Besides the overall accuracy, we further evaluate the results based on the Receiver Operating Characteristic (ROC) curve. The ROC curve is a graph showing the performance of a classification model at all classification thresholds.

The ROC curve is a plot of the false positive rate and true positive rate. The area under the ROC curve (AUC) measures the entire two-dimensional area underneath the entire ROC curve from $(0, 0)$ to $(1, 1)$ (Curve & AUC, 2018; Gardner et al., 2019), which means AUC ranges from 0 to 1. AUC provides an aggregate measure of performance across all possible classification, so we use the area between the proposed method's ROC curve and the baseline ROC curve to further compare the performance of our proposed method with baseline's, which is called ABROCA in prior work (Gardner et al., 2019). To eliminate the randomness of the algorithms, we ran the program 10 times and made a $mean + std$ and $mean - std$ ROC-AUC graph (Figure 3.10). As Table 3.1 shows, the ABROCA value is always positive, this confirms that the aggregate performance of our proposed method is consistently better than the baseline across different models. Furthermore, we extended the student score prediction from a binary classification problem (correct or wrong) to a multiple classification problem (0-3). To further evaluate our method's performance in every score class, we selected ADD (area dimension question dataset) and drew a real-predicted heatmap (Figure 3.10) for each algorithm.

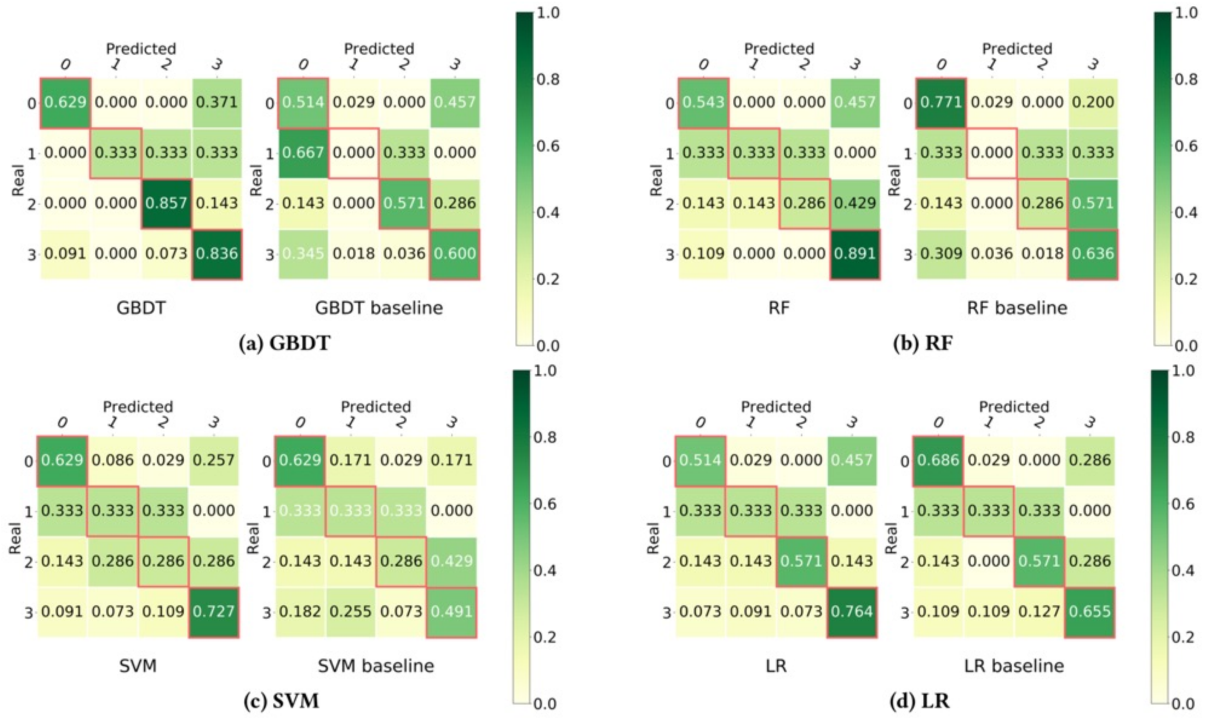


Figure 3.11: Predicted distribution heatmap of each score class with four algorithms on dataset ADD. The proportion of True Positive samples in each score class is in red box. In (a)-(d), the heatmap on the left shows the result of our proposed method and the right one shows the baseline method. ADD: area dimension question dataset.

Specially, the GBDT model provides an importance score for each feature, which is computed by the normalized summation of reduction in loss function of each feature and called Gini importance(Pedregosa et al., 2011). The importance score of each feature ranges from 0 to 1 and a larger importance score indicates that the corresponding feature is more important in training the model. This directly help to learn the importance distribution of our feature sets. In our method, the importance score of mouse movement feature set, most similar question's score class and the similarity score (39 features) reaches

27.4% among 483 features, which indicates the mouse movement feature set has significant contribution in GBDT model.

3.3.2 Peer-inspired Student Performance Prediction with Graph Neural Network

To evaluate the method proposed in Section 2.3.2, predicting student performance using graph neural network. We setup an experiment and compare the prediction performance with four baselines.

Experiment Results

We conducted experiments to evaluate the prediction accuracy and weighted F1 score of students' performance on each question and there are 4 question score levels. We compared the proposed approach with 5 baseline approaches including 3 classical machine learning models without peers' mouse movement features. Our baselines used are Relational Graph Convolutional Networks (R-GCN), Gradient Boosting Decision Tree (GBDT), Supporting Vector Machine (SVM) and Logistic Regression (LR). *E2N* denotes Edge2Node technique mentioned in 2.3.2.

Evaluation Metrics We use three different metrics to evaluate models comprehensively. Here we use s, n_c^s, n^s, $W\text{-}F1^s$ to denote a student, the number of correctly predicted questions for a student, the number of questions in the test set, and the weighted F1 of the prediction results.

Average personal accuracy (AP-Acc) evaluates a model's average prediction accuracy on different students:

$$AP\text{-}Acc = \tfrac{1}{N} \times \textstyle\sum_{s=1}^{N} \tfrac{n_c^s}{n^s}. \tag{3.1}$$

Overall accuracy (O-Acc) evaluates a model's average prediction accuracy on all predicted questions:

$$O\text{-}Acc = \textstyle\sum_{s=1}^{N} n_c^s / \sum_{i=1}^{N} n^s. \tag{3.2}$$

Average personal weighted F1 (APW-F1) evaluates a model's average weighted F1 score on different students:

$$APW\text{-}F1 = \tfrac{1}{N} \times \textstyle\sum_{s=1}^{N} W\text{-}F1^s. \tag{3.3}$$

Short-term Dataset The short-term dataset contains 43,274 mouse trajectories made by 3,008 students on 1,611 questions. Taking into account that too few labeled data will make it difficult to train the GNN models, we only conducted experiments for 47 students who have finished at least 70 questions. The experiment results are shown in 3.3.

Table 3.3: The comparison between our method R^2GCN and the baseline models on *short-term dataset.*

Model	AP-Acc	O-Acc	APW-F1
R^2GCN	**0.6642**	**0.6662**	**0.6148**
R-GCN (with E2N)	0.6302	0.6331	0.5737
R-GCN (without E2N)	0.6151	0.6198	0.5508
GBDT	0.5687	0.5750	0.4398
SVM	0.5734	0.5805	0.4470
LR	0.5928	0.5961	0.5414

Long-term Dataset The long-term dataset contains 104,113 mouse trajectories made by 4,020 students on 1,617 questions. Taking into account that too few labeled data will make it difficult to train the GNN models, we only conducted experiments for 1,235 students who have finished at least 20 questions. The experiment results are shown in 3.3.

Table 3.4: The comparison between our method R^2GCN and baseline models on *long-term dataset.*

Model	AP-Acc	O-Acc	APW-F1
R^2GCN	**0.5507**	**0.5671**	**0.5050**
R-GCN (with E2N)	0.5100	0.5313	0.4605
R-GCN (without E2N)	0.5119	0.5296	0.4535
GBDT	0.4836	0.4610	0.3686
SVM	0.4973	0.4718	0.3801
LR	0.4881	0.4904	0.4322

Expert Interview

We have been working closely with our industry collaborator Trumptech for about one year on this research project. When finishing the proposed peer-inspired student performance prediction approach, we interviewed 5 online question designers and course instructors from Trumptech and they showed great interests in our approach and a senior online question designer commented *"the prediction accuracy is impressive compared with other state-of-the-art approaches. We'd like to deploy it on our platforms to facilitate other tasks, e.g., an adaptive question recommendation system"*. Another senior system engineer commented *"The proposed method can fulfill our requirement for evaluating students' proficiency and will benefit our downstream tasks like adaptive recommendation"*

The experts from Trumptech also suggested that it will be interesting to extend the proposed approach to work for online learning platforms designed for touch screen devices, as they are becoming increasingly popular for online learning.

3.4 Learning Path Planning

3.4.1 User Studies of PeerLens

To evaluate how our system, PeerLens, assist students in planning their learning paths as introduced in Section 2.4, we conducted user studies with 18 participants and comparted with two baselines interms of informativeness, decision making, visual design, and usability.

Experiment Design

According to Weibelzahl's work (Weibelzahl, 2001), we adopted a four-layer taxonomy to evaluate our system. In particular, we conducted an user study to systematically assess the *informativeness* of the knowledge delivered, the *effectiveness* in facilitating the decision making, the *usability* of the proposed system, as well as the *visual design.*

Participants: We recruited 18 students (7 females, 11 males, age:24$\pm$2.85) from a local computer science department to conduct the user study. Each participant received a gift of $25 for their time after the study. All the participants have a basic knowledge of the online judge and question pool, and 9 out of 18 have had long experience using at least one of these online judge systems, e.g., LeetCode, TopCoder. We chose the participants with a computer science background as most of the popular online judges and question pools, as well as this study, are on a programming test topic, for which they could provide us more comprehensible insights.

Experiment setting and procedure: We compared *PeerLens* (namely the *full PeerLens*) with two alternative learning systems. One is the original online judge without an explicit design for the learning path planning (namely the *baseline* system). Questions in the baseline system are sorted by their problem IDs assigned upon creation, which is independent of contents and question difficulties; in a sense, the "recommendation" of the next question is almost random. The other one is a simplified version of *PeerLens* (namely the *primitive PeerLens*). The primitive system uses a truncated design of *PeerLens* with the same recommendation algorithm Figure **??**(b). The differences between the primitive and full version lie in: (1) the full version provides multiple learning paths for learners to select by themselves and the primitive version only provides one suggested learning path. (2) the full version makes use of several visual cues and hints to illustrate the semantics and statistics of suggested paths while no such cues are applied in the primitive version to interpret the learning path. According to (Sánchez-Ruiz et al., 2017), we designed three learning scenarios in this study: the basic programming practice, the coding qualification test for IT company interviews, and the International Collegiate Programming Contest preparation.To minimize the ordering and learning effect, we counterbalance the three systems in comparison with the three learning scenarios.

The actual experiment is composed of four sessions. In the first session, participants are briefed about the purpose and procedure of the experiment. Each following session lasts approximately 20 minutes and one of the three systems is presented and tested in one different learning scenarios.Each participant is required to conduct two tasks with the provided system. The first task is to determine the starting question under a specific learning scenario; The second task is to find the next question to solve given an existing historical learning path under the same learning scenario. Participants are asked to think aloud about their strategies to pick questions. After finishing all the tasks with a particular system, the participant is required to complete a questionnaire with 7-point Likert scale questions derived from the existing literature (Du et al., 2017; ?, ?), which is shown in Table 3.5.

Table 3.5: Our questionnaire focuses on 4 aspects: informativeness (Q1−Q3), decision making (Q4−Q6), visual design (Q7−Q8) and usability (Q10−Q12).

Q1	The information needed to plan a learning path is easy to access.
Q2	The information needed to plan a learning path is rich.
Q3	The information is sufficient to plan a learning path.
Q4	The system was helpful for me to find a proper learning path for a specific learning scenario.
Q5	I am confident that I find a suitable learning path for the learning scenario.
Q6	The system helps make adjustment according to previous performance.
Q7	The learning path design is intuitive.
Q8	The learning path design helps me understand the suggested path.
Q9	It was easy to learn the system.
Q10	It was easy to use the system.
Q11	I would like to recommend this system to others.

Results and Analysis

In summary, the results on informativeness and decision-making efficacy demonstrate that the proposed visual designs in *PeerLens* are informative as they provide accessible, rich,

Table 3.6: Repeated measures ANOVA of *Baseline*, *Primitive*, *Full* on informativeness, decision-making, *Primitive* and *Full* on visual designs and system usability.

		df	*F*	*Sig.*	η^2
Informativeness	accessibility	1	119.05	0.00	0.875
	richness	1	43.59	0.00	0.719
	sufficiency	1	153.86	0.00	0.364
Decision-making	confidence	1	79.12	0.00	0.823
	guidence	1	327.71	0.00	0.951
	adjustment	1	84.24	0.00	0.832
Visual design	intuitiveness	1	6.25	0.23	0.27
	comprehension	1	8.01	0.12	0.32
System usability	easy to learn	1	0.57	0.46	0.03
	easy to use	1	0.60	0.45	0.03
	recommendable	1	11.12	0.00	0.40

and sufficient information to learners. The submission distribution of the problem offers a clear visual cue of the difficulty of problems. In addition, the full-version *PeerLens* facilitates the decision-making process by providing more options, which allows the learners to make more adjustments, and offers more guidance to the learners. For example, the comparison of the learner's performance with the peer group's performance help learners choose which path to follow. The visual hints, such as reminding learners to try again, helped them to decide whether to move on. Hence, learners tended to be more confident when planning a path for their own studies. Overall, the proposed full-version *PeerLens* system is more intuitive and comprehensible for learner to learn and use, and is thus worth recommendation.

3.5 Learning Behavior Regulation

3.5.1 User Studies of Intervention

To evaluate the effect of information visualizations in promoting students' reflect on "Gaming the system" behavior as proposed in Section 2.5, we deployed our interventions (reflective nudges) on the research platform we designed for. This serves as a technology probe for understanding users' needs and experiences in a real-world setting to inform the proper design of technologies. We analyzed data from students' submission records, online questionnaires, and in-depth interviews to gain insights into the effectiveness of our approach on gaming reduction, information conveyance, reflection promotion, and visualization understanding.

3.5.2 Experiment Design

This part describes the deployment setting, online questionnaires, and the procedure of post-study interviews.

Deployment Setting We launched our reflective nudges on the previously mentioned online programming homework platform, which has been used by a university-level introductory programming course with 205 students from various departments. Since gaming behavior is affected by many contextual factors such as time (beginning of the semester, midterm period, etc.) and problem attributes (topic, difficult level, etc.) (Paquette et al., 2018), we selected problems from the last module of the course to ensure the relative consistency of these factors (time and topics). Also, it might be easier for us to observe the effect as a certain number of students are likely to game the system due to the time pressure in that period.

In particular, the last course module has four multiple-choice questions (P1-P4), and students have to finish them within two weeks. We applied our interventions to the last two questions and used the students' submission behavior on the first two questions to describe their gaming patterns before adding our interventions. We divided the students into four groups evenly (44) by randomizing their ids (three experimental groups, each receiving one of the three visualizations during the problem-solving process, and one control group without visualization intervention), but each group had some students who did not log in the system to do the exercises and thus the final numbers were uneven (39 students in the control group; 37, 44, and 38 students in V1, V2, V3 group, respectively).

Questionnaire Upon finishing all the questions, participants in the non-control groups were invited to fill out a post-study online questionnaire (optional) derived from a previous study (Xia et al., 2019a) and the requirements by the instructors that we were not supposed to disturb the process with a long and open-ended questionnaire. These questions, listed below, are to measure the efficacy of the proposed gaming interventions in terms of information conveyance (Q1), reflection promotion (Q2 and Q3), and visualization understanding (Q4) on a 7-point Likert scale (from "1. Strongly disagree" to "7. Strongly agree"). Note that Q1 has three versions (Q1-1, Q1-2, and Q1-3) corresponding to three types of reflective nudges, respectively. **Q1-1** This visualization clearly shows that 'gaming the system may cause a longer time to review before the exam to ensure the final performance'. **Q1-2** This visualization clearly shows 'peers also spend considerable efforts on the difficult problems and gaming the system may cost more attempts than peers'. **Q1-3** This visualization clearly shows 'a learning concept is related to other learning concepts, so gaming the system on the questions may impair the performance on the others'. **Q2** This visualization helps you reflect on your perception of "gaming the system" behavior. **Q3** This visualization helps you reflect on your question-answering behavior. **Q4** It is easy to understand this visualization.

In-depth Interviews Since four questions with 7-point scales may not gather all information, we conducted post-study interviews with students who had received the interventions to gather reasons behind their questionnaire ratings and suggestions on these intervention designs. Altogether, we interviewed eight students (S1-S8, six males, age: 24 ± 3.48, with backgrounds in computer science, political science, psychology, and economics), whom we recruited on-site after office hours and sent recruitment emails to the course mailing list. As nearly no new insight emerged after the sixth interviewee, we did not conduct further recruitment. We asked about which interventions they received and their ratings again since we could not cross-check their anonymous survey results. Four of them received V2, two V1, and two V3. All procedures were approved by the local university's Institutional Review Board. Each interview lasted about 60 mins and students were compensated with $8.

The interview proceeded as follows. (1) We collected the participant's consent by signing the consent form. (2) We provided a 5-min introduction of the project and then collected their responses to general questions such as "did you finish the last two questions in week 12?", "which visualization did you receive?". We also showed the platform to help interviewees to recall their experiences with the visualization designs. (3) We went over all the three designs with each interviewee and asked Q1 - Q4 for the elaboration on their ratings. For example, if a participant gives a rating 5 (Somewhat Agree) in Q2, we would ask them which part(s) of the visualization helped on their reflection on gaming behavior and how. In addition, we added a fifth question to gain insights into room for improvement: **Q5** How can it be improved to help you promote your reflection on your "gaming the system" behavior?

3.5.3 Results and Analysis

Table 3.7: Gaming proportions change over time on four problems, among three experimental groups (V1, V2, and V3) and two Baselines (control group and last-semester group). The number of student in each group is shown in the parentheses. The last row is the first-time pass rate of these four problems in this semester to indicate the difficulty levels.

	P1	P2	P3	P4
V1-time spending(37)	0.30	0.65	0.11	0.08
V2-attempt number(44)	0.34	0.63	0.16	0.07
V3-prerequisite graph(38)	0.37	0.63	0.21	0.16
Baseline1-control group(39)	0.26	0.59	0.21	0.23
Baseline2-last semester(138)	0.32	0.65	0.21	0.16
First-time pass rate	0.26	0.09	0.71	0.56

Table 3.8: Means and standard deviations (in parentheses) for ratings of agreement (1-strongly disagree, 4-neither agree nor disagree, 7-strongly agree) on questions: Q1. information conveyance, Q2. reflection on gaming, Q3. reflection on question-answering, and Q4. easy to understand by conditions (V1, V2, and V3).

	Q1-Information conveyance	Q2-Reflection on gaming	Q3-Reflection on question-answering	Q4-Easy to understand
V1	4.6(1.7)	4.3(1.8)	4.3(1.8)	3.7(1.9)
V2	5.2(1.5)	4.7(1.2)	4.4(1.3)	5.1(1.2)
V3	5.8(0.7)	5.0(1.2)	5.3(1.1)	4.7(1.8)

We have some suggestive data that V1 and V2 may have better gaming reduction effects compared with the control groups on the two problems we deployed. The online rating and in-person interviews show that all three visualizations (V1, V2, and V3) convey reasons not to game to a certain extent. They all encourage students to review the ramifications of gaming behavior to some degree. In particular, V1 and V2 promote reflecting more on gaming itself, whereas V3 seems to lead students to think more about their question-answering behavior. V2 and V3 are easier to understand than V1.

CHAPTER 4

RELEASE

4.1 GitHub

Some of the above work's code repositories have been open-source and could be acquired easily. Their links are as follows:

Data Collection Technique: https://github.com/huanhuanBOY/mousetrack

SeqDynamics: https://github.com/xiameng552180/SeqDynamics_V0.git

The data of this system cannot be public because of the privacy limits. If want to explore the system, please contact us.

Because the materials of the other work are regulated by NDA(Non Disclosure Agreement), they cannot be released to public .

4.2 Project Page

We create a web page for this E-Learning project. In the website, we summarize the content of the project(including all two phases) and introduce the manpower in this project. All of our publications and achievements related to the project also be listed. If interested, the link is http://vis.cse.ust.hk/groups/e-learning/.

CHAPTER 5

CONCLUSION AND FUTURE WORK

Online learning has become popular in the past decades, and an increasing number of online platforms have emerged and provided all kinds of learning materials. For example, various MOOC platforms (e.g., Khan Academy), online question pools (e.g., LeetCode, LearnLex), and intelligent tutoring systems (e.g., SimStudent) offer interactive maths questions and/or programming exercises (Xia et al., 2019a). Among them, online question pools, as an important complement to the MOOC courses, provide students with many online exercises for practicing their knowledge. However, it remains unclear about how to further improve students' learning effectiveness in online learning question pools. In this white paper, we introduced our research outputs to enhance students' online learning experience when working on online question pools. In Chapter 1, we introduced how we collect the students' fine-grained mouse movement data during the problem-solving processes, which is generalizable to different online learning question pools. In Chapter 2, we presented our methods on analyzing students' problem-solving behaviors within multi-step questions and their behaviors in solving solving a series of problems, which can help question designers and instructors quickly gain deep insight into students' problem-solving processes. In Chapter 3, we extracted semantic features from the mouse movement data and predicted students' problem-solving performance, which can help the learning platforms estimate the question difficulty level for different students and further provide students with personalized learning. In chapter 4, we proposed new ways for students to customize their learning scenarios and recommended diversified learning paths for them to choose from. In Chapter 5, we designed reflective information visualizations to encourage students to reflect on their problem-solving behaviors and avoid gaming the system.

Besides the above work we have done, we believe there are still many interesting research problems worth further exploration. Here are several directions that we plan to explore in the future:

Automated Difficulty Identification and On-the-fly Guidance for Students. We have developed the visual analytics system, *QLens*, to help question designers understand students' difficulties and guide the question design. However, such student difficulty identification is mainly based on manual checking by instructors and question designers. It remains unclear how to automatically detect the difficulties students face and provide them with on-the-fly guidance to students further. For example, if some appropriate hints can be appropriately supplied to students getting stuck in a question, it can significantly improve students' learning experience in online learning platforms. How to handle these issues is worth further investigation.

Multi-modal Analysis of Student Learning Behaviors. In Chapter 2.1, we proposed a general way to collect students' mouse movement interaction data during their problem-solving process. These mouse movement data and other problem-solving records are used in all our subsequent research on data analytics and visualization approaches for enhancing online learning. Such kind of information provides us with an understanding of students' problem-solving processes to some extent. However, it is not able to provide us with a comprehensive understanding of students' problem-solving processes. A multi-modal analysis of student learning behaviors that involve different dimensions (e.g., student facial emotions and eye movement) can enable a comprehensive and more in-depth understanding of student learning behaviors in online learning platforms. With such kind of multi-modal analysis, we will be able to analyze student learning behaviors and states (e.g., cheating and learning engagement level), which, otherwise, is difficult to be achieved in online learning platforms.

Personalized Question Recommendation. Our research on student performance prediction (Wei et al., 2020; Li et al., 2020) shows that we can incorporate students' mouse movement data to predict students' performance in solving questions. The prediction accuracy is also much better than prior methods. One of the major downstream tasks that can benefit student performance prediction is personalized question recommendation, i.e., recommending appropriate questions to students to work on next. However, considering these predicted student performances on each question and further recommend proper questions is still not clear. Specifically, what is an "appropriate" question for different students to work on next? What kind of knowledge should be tested in the subsequent questions? If multiple questions are available, which one should be recommended for different students? There are many such kinds of questions that need further investigation.

Acknowledgement

This work is partially sponsored by the Innovation and Technology Fund (ITF) with No. ITS/388/17FP. We would like to thank our industry sponsor, Trumptech (Hong Kong) Ltd, for providing with the data and the research platform. We are also grateful to our research collaborators, Dr. Una-May O'Reilly and Dr. Erik Hemberg from Massachusetts Institute of Technology and Prof. Nancy Law from Hong Kong University, for their insightful comments and constructive suggestions on our research.

Bibliography

Abate, A., D'Innocenzo, A., Di Benedetto, M. D., & Sastry, S. S. (2008). Markov set-chains as abstractions of stochastic hybrid systems. In *International Workshop on Hybrid Systems: Computation and Control*, pp. 1–15. Springer.

Ausubel, D. P., Novak, J. D., Hanesian, H., et al. (1968). Educational psychology: A cognitive view..

Burnett, K., Bonnici, L. J., Miksa, S. D., & Kim, J. (2007). Frequency, intensity and topicality in online learning: An exploration of the interaction dimensions that contribute to student satisfaction in online learning. *Journal of Education for Library and Information Science*, 21–35.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *J. Artif. Int. Res.*, *16*(1), 321–357.

Crosses, M. L. F. (2019). Accessed: 2019-9-20.

Curve, C. R., & AUC (2018). Accessed: 2019-9-20.

Du, F., Plaisant, C., Spring, N., & Shneiderman, B. (2016). Eventaction: Visual analytics for temporal event sequence recommendation. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 61–70. IEEE.

Du, F., Plaisant, C., Spring, N., & Shneiderman, B. (2017). Finding similar people to guide life choices: Challenge, design, and evaluation. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 5498–5544. ACM.

Farkas, G. (2003). Cognitive skills and noncognitive traits and behaviors in stratification processes. *Annual review of sociology*, *29*(1), 541–562.

Gardner, J., Brooks, C., & Baker, R. (2019). Evaluating the fairness of predictive student models through slicing analysis. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, LAK19, pp. 225–234 New York, NY, USA. ACM.

George Trimponias, X. M., & Yang, Q. Rating worker skills and task strains in collaborative crowd computing: A competitive perspective. In *The Web Conference 2019*, p. 10.1145/3308558.3313569.

Goodspeed, R. (2017). Research note: An evaluation of the elo algorithm for pairwise visual assessment surveys. *Landscape and Urban Planning*, *157*, 131–137.

Li, H., Wei, H., Wang, Y., Song, Y., & Qu, H. (2020). Peer-inspired student performance prediction in interactive online question pools with graph neural network. *ArXiv*, *abs/2008.01613*.

Long, L. K., Hui, L. C., Fook, G. Y., & Zainon, W. M. N. W. (2017). A study on the effectiveness of tree-maps as tree visualization techniques. *Procedia Computer Science*, *124*, 108–115.

Manrique, R., Nunes, B. P., Marino, O., Casanova, M. A., & Nurmikko-Fuller, T. (2019). An analysis of student representation, representative features and classification algorithms to predict degree dropout. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, LAK19, pp. 401–410 New York, NY, USA. ACM.

Molenaar, I., Horvers, A., & Baker, R. S. (2019). Towards hybrid human-system regulation: Understanding children'srl support needs in blended classrooms. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pp. 471–480. ACM.

Paquette, L., Baker, R. S., & Moskal, M. (2018). A system-general model for the detection of gaming the system behavior in ctat and learnsphere. In *International Conference on Artificial Intelligence in Education*, pp. 257–260. Springer.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Piech, C., Sahami, M., Huang, J., & Guibas, L. (2015). Autonomously generating hints by inferring problem solving policies. In *Proceedings of the second (2015) acm conference on learning@ scale*, pp. 195–204.

Revilla, M. A., Manzoor, S., & Liu, R. (2008). Competitive learning in informatics: The uva online judge experience. *Olympiads in Informatics*, *2*(10), 131–148.

Row, M. B. (1974). Wait-time and rewards as instructional variables, their influence on language, logic, and fate control: Part one-wait-time. *Journal of Research in Science Teaching*, *11*(2), 81–94.

Sánchez-Ruiz, A. A., Jimenez-Diaz, G., Gómez-Martín, P. P., & Gómez-Martín, M. A. (2017). Case-based recommendation for online judges using learning itineraries. In *International Conference on Case-Based Reasoning*, pp. 315–329. Springer.

Schlichtkrull, M. S., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., & Welling, M. (2018). Modeling relational data with graph convolutional networks. In *ESWC*, pp. 593–607.

Shephard, R. J. (1968). Intensity, duration and frequency of exercise as determinants of the response to a training regime. *Internationale Zeitschrift fuer Angewandte Physiologie Einschliesslich Arbeitsphysiologie*, *26*(3), 272–278.

Stahl, R. J. (1994). *Using "Think-time" and "Wait-time" Skillfully in the Classroom.* ERIC Clearinghouse.

Wei, H., Li, H., Xia, M., Wang, Y., & Qu, H. (2020). Predicting student performance in interactive online question pools using mouse interaction features. *ArXiv, abs/2001.03012*.

Weibelzahl, S. (2001). Evaluation of adaptive systems. In *International Conference on User Modeling*, pp. 292–294. Springer.

Xia, M., Sun, M., Wei, H., Chen, Q., Wang, Y., Shi, L., Qu, H., & Ma, X. (2019a). Peerlens: Peer-inspired interactive learning path planning in online question pool. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–12.

Xia, M., Wei, H., Xu, M., Lo, L. Y. H., Wang, Y., Zhang, R., & Qu, H. (2019b). Visual analytics of student learning behaviors on k-12 mathematics e-learning platforms. *arXiv preprint arXiv:1909.04749*.

Xia, M., Xu, M., Lin, C.-e., Cheng, T. Y., Qu, H., & Ma, X. Seqdynamics: Visual analytics for evaluating online problem-solving dynamics..