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

Huan Wei, Haotian Li, Meng Xia, Yong Wang, Huamin Qu
Department of Computer Science and Engineering, HKUST, Hong Kong SAR, China
{hweiad,hlibg,iris.xia,ywangct}@connect.ust.hk;{huamin}@cse.ust.hk

ABSTRACT

Modeling student learning and further predicting the performance is a well-established task in online learning and is crucial to personalized education by recommending different learning resources to different students based on their needs. Interactive online question pools (e.g., educational game platforms), an important component of online education, have become increasingly popular in recent years. However, most existing work on student performance prediction targets at online learning platforms with a well-structured curriculum, predefined question order and accurate knowledge tags provided by domain experts. It remains unclear how to conduct student performance prediction in interactive online question pools without such well-organized question orders or knowledge tags by experts. In this paper, we propose a novel approach to boost student performance prediction in interactive online question pools by further considering student interaction features and the similarity between questions. Specifically, we introduce new features (e.g., think time, first attempt, and first drag-and-drop) based on student mouse movement trajectories to delineate students' problem-solving details. In addition, heterogeneous information network is applied to integrating students' historical problem-solving information on similar questions, enhancing student performance predictions on a new question. We evaluate the proposed approach on the dataset from a real-world interactive question pool using four typical machine learning models. The result shows that our approach can achieve a much higher accuracy for student performance prediction in interactive online question pools than the traditional way of only using the statistical features (e.g., students' historical question scores) in various models. We further discuss the performance consistency of our approach across different prediction models and question classes, as well as the importance of the proposed interaction features in detail.

CCS CONCEPTS

• **Applied computing → E-learning**; **Interactive learning environments**; *Learning management systems*; • **Computing methodologies →** *Feature selection.*

KEYWORDS

Student performance prediction, question pool, mouse movement trajectory, heterogeneous information network.

ACM Reference Format:
Huan Wei, Haotian Li, Meng Xia, Yong Wang, Huamin Qu. 2020. Predicting Student Performance in Interactive Online Question Pools Using Mouse Interaction Features. In *Proceedings of the 10th International Conference on Learning Analytics and Knowledge (LAK '20), March 23–27, 2020, Frankfurt, Germany*. Frankfurt, German , 10 pages. https://doi.org/10.1145/3375462.3375521

1 INTRODUCTION

With the rapid development of digital technologies, online education has become increasingly popular in the past decade, which provides students with easy access to various online learning materials such as massive open online courses (MOOCs) and different online question pools. The availability of these online learning materials and student activity logs has made it possible to model student learning and further predict their performance [20]. It has become a well-established task in computer-aided education, since student performance prediction can help course designers to better handle the possible dropout, improve the retention rate of online learning platforms [22], and provide students with personalized education to enhance student learning by recommending different learning resources to them based on their different needs [12, 28, 38, 44].

Online question pools, consisting of a collection of questions, have become increasingly popular for helping students to practice their knowledge and skills [1]. For example, there have been many popular online question pools that provide students hands-on exercises to practice their programming skills (e.g., LeetCode[2], Topcoder[3]) and mathematics skills (e.g., Learnlex[4], Math Playground[5]). The *interactive online question pool*, as one kind of the online question pools, comprises interactive questions where interactions such as mouse movement and drag-and-drop are often needed. For example, some online educational games have been designed to involve user interactions to make it a fun way to practice important skills [18] [6]. Despite the popularity and importance of interactive online question pools, little work has been done to model student learning and predict their performance in online question pools.

Student performance prediction has been widely explored in education community and is a critical step for downstream tasks, such as recommending an adaptive learning path [39] or assisting students at an early stage [7]. However, most of them focus on student

[1]https://help.blackboard.com/
[2]https://leetcode.com/
[3]https://www.topcoder.com/
[4]https://mad.learnlex.com/login
[5]https://www.mathplayground.com/
[6]https://www.weareteachers.com/

performance prediction on the MOOCs platforms (e.g, Coursera, EdX, Khan Academy), and little work has been done on interactive online question pools. Compared with student performance prediction on MOOCs platforms, it is more challenging to predict student performance on interactive online question pools due to two major reasons. First, there is ***no predefined question order*** that users can follow; students need to explore the questions by themselves when working on the interactive online question pools. Moreover, for some interactive online question pools, their questions only have ***rough knowledge tags*** annotated by domain experts, which are not accurate enough to evaluate the similarity among questions. As will be introduced in Section 2.1, the major models for student performance prediction include Bayesian knowledge tracing [9, 25], deep knowledge tracing [28, 44] and traditional machine learning models [7, 22, 24]. However, almost all of these models intrinsically depend on the course curriculum or the predefined question order and Bayesian knowledge tracing models also require knowledge tags, making them inapplicable to student performance prediction on interactive online question pools.

To handle the above challenges, we propose a novel method by introducing a set of new features based on interactions between students and questions to perform student performance prediction in interactive online question pools. Specifically, we utilize the mouse movement trajectories, which consists of the mouse interaction timestamp, mouse event type and mouse coordinates, to predict student performance. Such mouse movement trajectories represent the problem-solving path of a student for each question. Inspired by prior researchers in education and psychology fields [31, 35], 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. Specifically, we define and measure the time between when students see the question and the time that they start solving the questions as the "think time". In addition, attributes related to the first attempt and first drag-and-drop are also extracted. Further, since there is no predefined question order in interactive online question pools, different students can work on the questions in different orders. We apply a heterogeneous information network (HIN) to calculate the similarity among questions, in an effort to incorporate students' problem-solving history to enhance performance prediction in online question pools. 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 – Logistic Regression (LR), Gradient Boosting Decision Tree (GBDT), Support Vector Machine (SVM) and Random Forest (RF).

The contributions of this paper can be summarized as follows.

- We introduce novel features based on student mouse movement trajectories to predict student performance in interactive online question pools. Features like the "think time" can reveal students' thinking details when working on a specific question.
- We propose a novel approach based on HIN to incorporate students' historical problem-solving information on similar questions into the performance prediction on a new question.
- We evaluate our approach using real-world dataset and compare with state-of-the-art baseline features on typical multiple-classification machine learning algorithms. The 4-class prediction result shows that our approach achieves a much higher performance prediction accuracy than the baseline features in various models, demonstrating the effectiveness of the proposed approach.

2 RELATED WORK

The related work can be categorized into three groups: student performance prediction, problem-solving feature extraction, and question similarity calculation.

2.1 Student Performance Prediction

There are mainly two ways in performance prediction: the knowledge tracing and the traditional machine learning approach (e.g., Multiple regression). Methods based on or extended from knowledge tracing usually utilize a computational model of the effect of practice on KCs (i.e. knowledge components, which may include skills, concepts or facts) as the way to individually monitor and infer students' learning performance [29]. Bayesian Knowledge Tracing (BKT) [9] was the most popular approach to model students' learning process, where each learning concept was represented as a binary variable to indicate whether or not the student has mastered the learning concept or not. However, this method gives little consideration to the individual students' learning ability. Learning Factors Analysis (LFA) [3] extended the basic formulation by adding the factor–learning rate. More factors were taken into consideration by later methods, such as Performance Factors Analysis (PFA) [26], which further incorporated the students' responses to the questions (correct or incorrect). Additive Factors Analysis Model (AFM) [8] added the step duration (time between actions). The performance was improving as more factors are considered. However, the drawback of the methods based on the knowledge tracing is that they need a requirement for accurate concept labeling. Though recent studies showed that there is a possibility to make use of deep learning algorithm (i.e., Recurrent Neural Network) to predict students' performance on consecutive questions, the performance highly relied on the predefined question order (i.e., most of the students followed the same order to solve questions) [28].

For questions in question pools, they have no predefined order and no accurate concept labels [39], which hinders the way to easily adapt methods based on the knowledge tracing. Many studies have used traditional machine learning methods to predict the drop out rate or the course grade [7, 19, 21–23]. For example, Naive Bayes (NB), RF, GBDT, SVM, k-Nearest Neighbour (KNN) with Dynamic Time Warping (DTW) distance were used to predict college students' dropout [22]. Chen *et al.* [7] also used Logistic Regression and Nearest-neighbors to predict the dropout in MOOCs. Kabakchieva and Dorina [19] compared different machine learning models (Decision tree classifier, Bayesian classifiers, KNN classifier) for college students for grade prediction. Regardless of the chosen algorithm, much of the performance of a prediction model depended on the proper selection of feature vectors [22] and the transformation of feature vectors [17]. To achieve a higher accuracy of student performance prediction in the interactive online question pools, we need to extract more meaningful features that are closely related to students problem-solving capabilities.

2.2 Problem-Solving Feature Extraction

Many problem-solving features have been extracted and applied to the student performance prediction on questions. Among them, the submission record and the clickstream are two data types that are studied most. Xia *et al.* [39] used the submission records to model how a student solves a particular problem in online question pools, e.g., solving the problem with one submission or many submissions. Chen and Qu [6, 30] extracted features of the clcikstream data of watching MOOC videos and analyzed its correlation with final grades. Chounta and Carvalho [8] proposed the use of the response time (i.e., the time between seeing the questions and giving a response to tutor's question or task) to predict students' performance for unseen tasks in intelligent tutoring systems. The result showed that the quadratic response time parameter outperformed the linear response time parameter in the prediction tasks.

However, students' problem-solving behavior is a complex process involving different stages – problem decomposition, abstraction, and execution [35, 41]. It is critical to figure out different stages to better understand and predict their influence on individuals' performance [40] rather than treating it as a whole solving period. In the interactive question pool we study, more detailed mouse movement sequences (i.e., the trajectories with both position and timestamp information) are collected, which reflect the students' problem-solving ability in extensive details. Stahl *et al.* [35] introduced *"think time"*, which is a period of uninterrupted silence time given by the teacher in class and all students are asked to complete appropriate information processing tasks. Different students may have different abilities in processing the question information before they start to solve it [38]. Inspired by these previous work, we propose extracting "think time" from students' mouse movement trajectories to delineate the thinking process and problem-solving capabilities of different students.

2.3 Question Similarity Calculation

Question similarity is one of the key features to infer students' performance in previous researches. For example, When students repeatedly solve questions under the same topic, they may improve the mastery on this learning concept [3, 9, 26]. A wide range of work has been done to calculate question similarity when there is no accurate expert annotation of the problems. One branch of work calculated the semantic similarity of questions based on Natural Language Processing (NLP). For example, Song *et al.* [34] first identified keywords in questions, and then calculated semantic similarity between questions based on the keywords. They also extended the semantic similarity to statistic similarity, which was calculated using the cosine similarity between two question vectors. Each question vector was a string of binary bits with each bit indicating the existence of a certain word. Their results showed that the combination of semantic similarity and statistic similarity could achieve a better performance than each individual algorithm. Similarly, textual similarities [4] and question type similarity [1] were also proposed to further improve the performance prediction.

However, all these methods require accurate knowledge tags and abundant text information about the questions. The questions in the interactive math question pools usually use simple description to describe the problem background without showing the knowledge tested explicitly. Another branch of work tried to calculate the question similarity according to the interaction data between students and problems. For example, Xia *et al.* [39] calculated the question similarity based on the submission types (e.g., successful submission after one submission or many submissions). However, more detailed student interaction information (e.g., mouse movement trajectories), which intrinsically reflects students' problem-solving habits and capabilities, is not considered. HIN is defined as the information network with more than one type of objects or relationship between objects, which is used to describe the complex structure of information in the real world. Bibliographic information network and Twitter information network are examples of HIN [36]. In this paper, we use the HIN to incorporate different information (e.g., students' historical scores, mouse movement trajectories) into calculating the question similarity.

3 CONTEXT

Our study is built on the dataset collected from an interactive online math question pool. This section introduces the interactive math questions, the collected data and the overall student performance prediction framework of our study.

3.1 Interactive Math Question

The platform we cooperate with is an interactive question pool used by more than 40,000 students from 30+ primary and junior high schools in Hong Kong. There are 1,720 interactive math questions on the platform and each question has labels from *math dimension*, *difficulty*, and *grade*. The *math dimension* indicates the general knowledge domain of the question. *Difficulty* is a five-star rating ranging from 1-5 (easy to hard), which is predefined by several education experts and the question maker. *Grade* represents the year of study this question is designed for. Figure 1 shows an example of the interactive math question in the question pool, students need to use their mouse to drag the red dot to fulfill the requirement and get a score ranging from 0 to 100. Since the scores are discrete and possible scores of each question are not the same, we manage to map the original score ranging from 0 to 100 to 4 score classes (0-3).

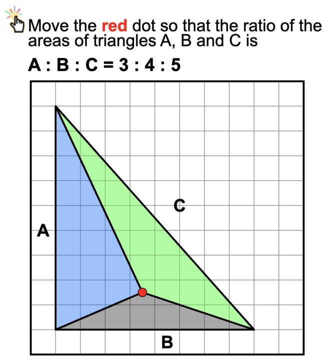
Move the red dot so that the ratio of the areas of triangles A, B and C is
A : B : C = 3 : 4 : 5
A
B
C

Figure 1: An example of the interactive math question.

Table 1: The statistics of two datasets: ADD (area dimension dataset) and DGDD (deductive-geometry dimension dataset). Both consist of students' problem-solving records from April 12, 2019 to June 27, 2019.

Dataset	#Questions	#Trajectories	#Students
ADD	61	1764	724
DGDD	64	2610	562

3.2 Data Collection

The data we collected are historical score records and mouse movement trajectories. In total, we collected 858,115 historical score records from September 13, 2017 to June 27, 2019 in total. We developed a tool to collect students' mouse movement trajectories, which included the mouse events (mouse move, mouse down, mouse drag and mouse up), the timestamps, and the positions of the mouse during the whole problem-solving process. We selected two question sets that contains rich mouse interactions from two math dimensions (*Area* and *Deductive geometry*). Table 1 shows the statistical information of these two question sets. There are 61 and 64 questions in these two categories respectively. 724 students produced 1764 mouse movement trajectories in the Area and 562 students made 2610 mouse movement trajectories in Deductive geometry. Note that the system allows students to submit up to two times for each question, our research goal in this paper is to predict students' score on the first submission of the question.

3.3 Overall Prediction Framework

In order to conduct a state-of-the-art machine learning prediction experiment, we survey existing studies for the features used in student performance prediction [22, 24], mouse trajectory feature definition [33, 42], and possible methods in question similarity calculation [1, 4, 34, 36, 37]. We summarize this knowledge with our dataset and task, then we propose our prediction framework in Figure 2. It mainly contains three modules: data collection and preprocessing, feature extraction, and prediction and evaluation.

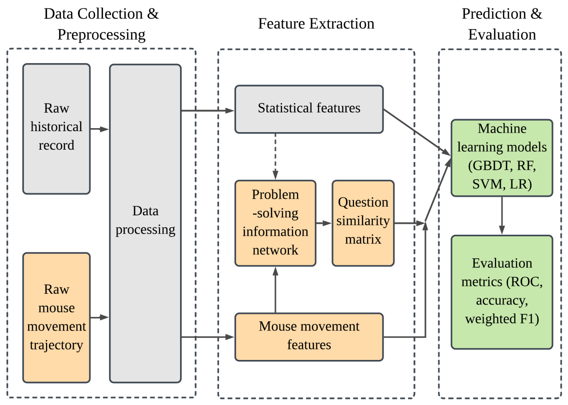
Data Collection & Preprocessing
Feature Extraction
Prediction & Evaluation
Raw historical record
Raw mouse movement trajectory
Data processing
Statistical features
Problem -solving information network
Question similarity matrix
Mouse movement features
Machine learning models (GBDT, RF, SVM, LR)
Evaluation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

Figure 2: The prediction framework contains three modules: data collection & preprocessing module, feature extraction module, and prediction & evaluation module. The blocks highlighted in yellow and green correspond to the major contributions of this paper.

In the feature extraction module, as suggested by previous studies [13, 15], students' recent performance may have a great impact on prediction. We then 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) as well as the number of total submissions and second submissions. Further more, mouse movement trajectories are processed 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 the problem-solving information network, a 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 features. Similar to previous research [14, 22], we compare prediction accuracy, weighted F1 and ROC curves and feature importance score in GBDT to evaluate and discuss our method.

4 FEATURE EXTRACTION

In this section, we introduce the feature extraction module in detail. Specifically, we first explain how to detect change points in the mouse movement trajectories, based on which we illustrate how we extract mouse movement features (i.e., think time, first drag-and-drop, and first attempt). We then introduce other statistical features like students' historical performance features. Lastly, we describe how to use a HIN to incorporate both statistical features and mouse movement features to calculate the similarities between questions. Assuming that students often have similar performance on similar questions, we integrate features of similar questions to further enhance the performance prediction on a new question.

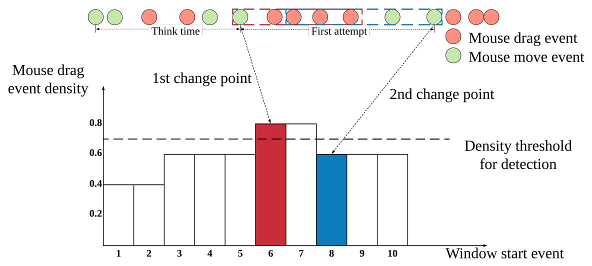

Figure 3: A sample mouse movement trajectory and the scheme diagram of change point detection algorithm. The 1st change point is both the think time end point and the first attempt start point. The 2nd change point is the first attempt end point.

4.1 Change Point Detection

To infer students' problem-solving capabilities from the mouse movement trajectories, we need to identify different problem-solving stages [40, 41]. 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. As shown in Figure 3, we split the mouse movement trajectory into three subparts 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. Think time is a stage that starts from when a student opens the question and ends at the students use the mouse to solve the question with actual interactions [35]. The first attempt stage is a series of mouse drag events trying to solve the problem after the think stage and ends when the frequency of mouse drag events becomes low. The first change point can differentiate the think time stage from the first attempt stage and the second change point is the boundary of the first attempt stage and the following actions stage. A straightforward way of change point detection is to use the first movement of the mouse as a signal that the student starts to solve the question. However, in the real world, it is not always true that the first mouse movement represents the starting point of solving a problem. After analyzing the mouse movement trajectories, we find that there may exist a short drag-and-drop or click by mistake, which makes it seem like the student starts to solve questions. To detect the most probable start and end time points of each stage, we propose the change point detection algorithm.

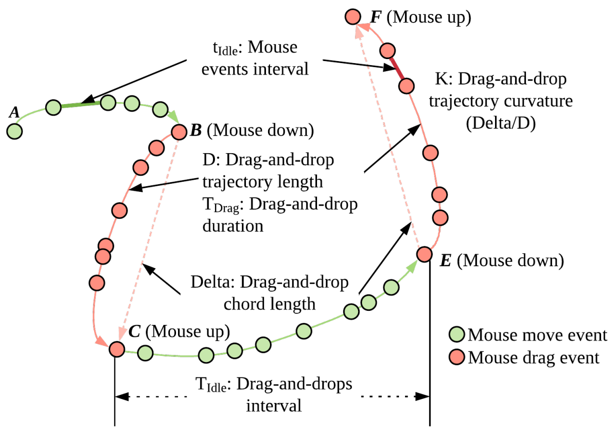
F (Mouse up)
t_{Idle}: Mouse events interval
K: Drag-and-drop trajectory curvature (Delta/D)
A
B (Mouse down)
D: Drag-and-drop trajectory length
T_{Drag}: Drag-and-drop duration
Delta: Drag-and-drop chord length
E (Mouse down)
C (Mouse up)
Mouse move event
Mouse drag event
T_{Idle}: Drag-and-drops interval

Figure 4: An example of mouse movement trajectories in problem-solving process.

We use the sliding window to detect the change points, which is a common method for abrupt change detection in the time series data [2]. It has two key parameters: window size and detection threshold. The detection threshold can be observed through the test data or sometimes random selection [2]. For a mouse movement trajectory, we define the total mouse events as C_t, the total mouse drag events as C_d. The event density in trajectory is defined as:

$$\rho = C_d / C_t \tag{1}$$

We count the mouse drag events and calculate the mouse drag event density in sliding window and compare the value of density with the threshold. The scheme of change detection is as follows:

- Move the window from the beginning of the mouse events sequence, when the first time ρ is higher than the threshold, the first point of the sliding window is defined as the first change point.
- Continue sliding window until ρ is lower than the threshold, the last point of the sliding window is defined as the second change point.

Table 2: 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.

4.2 Mouse Movement Features

Student interaction data, especially mouse movement data, contains massive information [16, 33]. We extract two sets of features: TFF (think time, first attempt, and first drag-and-drop) and MDSM (mouse drag statistical measurements) based on previous studies [33, 35, 42]. We use TFF and MDSM to model students' initial and overall behaviors in a problem-solving process, respectively.

As for the TFF, the think time and the first attempt have been introduced in Section 4.1. Drag-and-drops is a series of consecutive mouse 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 event with a mouse down and a mouse up. Table 2 shows the detailed attributes of the features in TFF.

As for MDSM, we define the following features to represent mouse drag statistical measurements as Figure 4 shows.

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

However, there may be more than one drag-and-drop in a trajectory. To extract useful information in drag-and-drops, we use three statistical methods median, mean and interquartile range (IQR), to measure it (Table 3). Besides the features above, we also build other

Table 3: 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.

features to measure the whole trajectory, including the number of drag-and-drops, total time, total mouse events and mean, median, IQR of mouse events in every second.

4.3 Other Statistical Features

Information such as score records and score distribution on questions can also reflect a student's ability and the question difficulty for all the students. Features extracted from such records are widely applied in prediction of students' performance, for example, Yu *et al.* [45] used students' recently solved questions to construct temporal features and Manrique *et al.* introduced average grades of students to predict their dropout rates [22]. To make full use of the information, we take the cross feature approach. A cross feature is a synthetic feature formed by multiplying (crossing) two or more features. Crossing combinations of features can provide predictive abilities beyond what those features can provide individually [10]. Thus, we apply the cross feature method to questions' and students' basic statistics. As Table 4 shows, we have three parts of statistical features: question statistics, student statistics, and the recent statistics of a student.

We use the expression A × B to represent the cross feature of A and B, expression #C in [E] to represent the numbers of C for each category or dimension in E and expression %C in [E] to represent the proportion of C for each category or dimension in E. For example, in students statistics, %Submission in [math dimension × grade × difficulty] represents the proportion of a student's submission number in each math dimension with a specific grade and difficulty level.

4.4 Problem-solving Information Network and Similarity Calculation

To predict the performance of a student s_i on a question q_x using mouse movement features, an intrinsic requirement is that we should have the mouse movement trajectories. However, they are not available before a student actually finishes the question. Prior studies [32, 43] have shown that a student's performances on similar questions are often similar. Therefore, we propose finding a question q_y similar to the question q_x and using its mouse movement features extracted from the trajectories as part of the feature vectors to predict a student's performance on q_x.

The similarity between questions can be evaluated from different perspectives (e.g., difficulty level, question content, student mouse movement interactions, etc.). For example, for two questions that examine the same knowledge, their difficulty levels and the required problem-solving skills can be significantly different for different students. To more accurately delineate the similarity between any two questions, we use students' interactions (e.g., mouse movement trajectories) as the bridge from one question to another question. Specifically, we build a network consisting of both interactions between students and questions and the intrinsic attributes of questions to calculate similarity for each pair of questions. Such a network is called *problem-solving information network* in this paper.

Table 4: Other statistical features: questions statistics, student statistics and student recent statistics. Symbol: #: Number of records, %: Proportion of records. Expression A × 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 × grade × difficulty]	Student's proportion of submissions in each specific math dimension, grade and difficulty.
1stAvgScore in [math dimension × grade × 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 × 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 × 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 × difficulty]	Score standard deviation of each grade and difficulty in past N days.

4.4.1 *Problem-solving Information Network Structure.*

The problem-solving information network, a typical bipartite HIN, is established between two kinds of objects, students (S) and questions (Q). For each question $q \in Q$, it has links to several students and the link between a question q and a student s is defined as solving (s solves q) or solved by (q is solved by s). The network schema of the problem-solving network is shown in Figure 5 (a) and the symmetric meta-path in our network is defined as Question-Student-Question (QSQ), which denotes the relationship between two questions that have been solved by the same student.

4.4.2 *Similarity Calculation.*

Sun *et al.* [37] proposed a meta-path based similarity framework and the framework on meta path $\mathcal{P}$ is

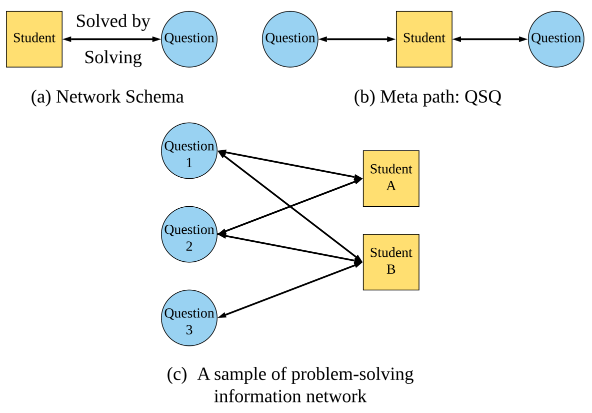

Figure 5: The schematic diagrams of problem-solving information network. (a) Network schema, (b) Meta path: *QSQ*, and (c) A sample of problem-solving information network.

defined as:

$$s(a, b) = \sum_{p \in \mathcal{P}} f(p) \tag{2}$$

where $f(p)$ denotes a measure defined on a path instance $p \in \mathcal{P}$ from an object a to another object b.

Following this guideline, we propose applying such a meta-path based similarity framework to our application scenario and further measure the similarity of a question q_x and another question q_y under a meta path QSQ in the problem-solving network (Figure 5). Each path instance $q_x s_i q_y \in QSQ$ passing a student s_i is measured using the cosine similarity:

$$f(q_x s_i q_y) = \frac{\mathbf{Feature_{ix}} \cdot \mathbf{Feature_{iy}}}{\|\mathbf{Feature_{ix}}\| \|\mathbf{Feature_{iy}}\|} \tag{3}$$

where $\mathbf{Feature_{ix}}$ is the feature vector, consisting of both mouse movement features generated based on the mouse trajectory of the submission on q_x by a student s_i (as introduced in Section 4.2) and the score of his/her submission.

We normalize the similarity score of each path instance $q_x s_i q_y$ by using the sum of the similarity scores of all the students who have finished both questions q_x, q_y, guaranteeing that every score in our problem-solving network is between 0 and 1. Thus, the final similarity between q_x and q_y on meta path QSQ is defined as:

$$s(q_x, q_y) = \frac{\sum_{q_x s_i q_y \in QSQ} \frac{\mathbf{Feature_{ix}} \cdot \mathbf{Feature_{iy}}}{\|\mathbf{Feature_{ix}}\| \|\mathbf{Feature_{iy}}\|}}{\left|\{q_x s_i q_y : q_x s_i q_y \in QSQ\}\right|} \tag{4}$$

5 EXPERIMENTS

5.1 Experiment Setup

Our experiment was conducted on the two datasets introduced in Table 1. 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. First, we built question statistics and student statistics (Table 4) with all records before April 12, 2019. We then calculated the student recent statistics for each record after April 12, 2019, using the data in the past 14 days of that record. For those submissions without recent records, we assigned -1 to all recent performance features. The features listed in Table 4 served as features in baseline method.

Based on the proposed mouse movement features, similarity matrix M_{sim} of each dataset was extracted according to the problem-solving network and related similarity calculation algorithms mentioned in Section 4.4. 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 (with a similarity threshold of 0.8 in the experiment on ADD, 0.7 in the experiment on DGDD) and is 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 4, q_y's mouse movement features including features in Table 2 and Table 3, q_y's score class and the similarity score between q_x and q_y.

Since we discarded submissions with no similar questions, 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 [5]. As for hyperparameter tuning in the four algorithms, grid-search was conducted in model training on our datasets [22]. 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 6.

5.2 Performance Comparison

The results for our dataset ADD and DGDD are in Table 6. 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. In addition, the performances of our method and the baseline are similar in both datasets.

Table 5: 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**

Table 6: 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**

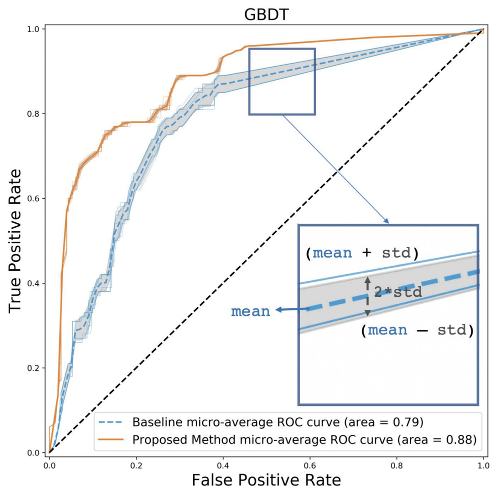

Figure 6: ROC-AUC curve of two methods on ADD (Area dimension question dataset).Gray borders represent mean+std or mean-std. Area between the two curves is ABROCA.

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 $(1, 1)$ [11, 14], 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 [14]. 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 6). As Table 5 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 7) for each algorithm. 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 it is also called Gini importance[27]. 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 helps 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.

6 DISCUSSION

The above experiment results demonstrate that our approach can achieve higher accuracy for predicting student performance prediction in interactive online question pools than using the baseline features. However, there are still some issues that need further discussions.

Parameter Configurations Some parameters that need to be set in the proposed approach and some of these parameters are empirically chosen after considering different factors. For example, the sliding window is used to detect the change points (Section 4.1), where the value of the window size and threshold need to be determined first. Too large a window size will make the mouse drag event histogram too smooth, while too small a window size may make the mouse drag event frequency histogram too steep. Both have a negative effect on change point detection. The corresponding threshold is also important. We empirically set it as the average mouse drag density to guarantee that the detected change points are exactly the actual ones. With similar considerations, we choose only the question with the highest similarity score and require that the similarity score should be at least 0.7, when we try to incorporate the information of similar questions. Our experiment results provide support for the effectiveness of the current parameter settings.

Prediction Models This paper focuses on proposing novel methods to extract features for student performance prediction. These features are further combined with existing well-established traditional machine-learning based prediction models to predict student future performance. Compared with other methods based on deep neural networks (e.g., deep knowledge tracing [28, 44]), we argue that predictions based on feature extraction have better explainability, as the features are often intuitive and meaningful.

Data Issues Since there are no other dataset of interactive online question pools publicly available, the whole study is conducted on the data collected from only one interactive online math question pool. But our proposed approach can be easily extended to other datasets of interactive online question pools, which mainly

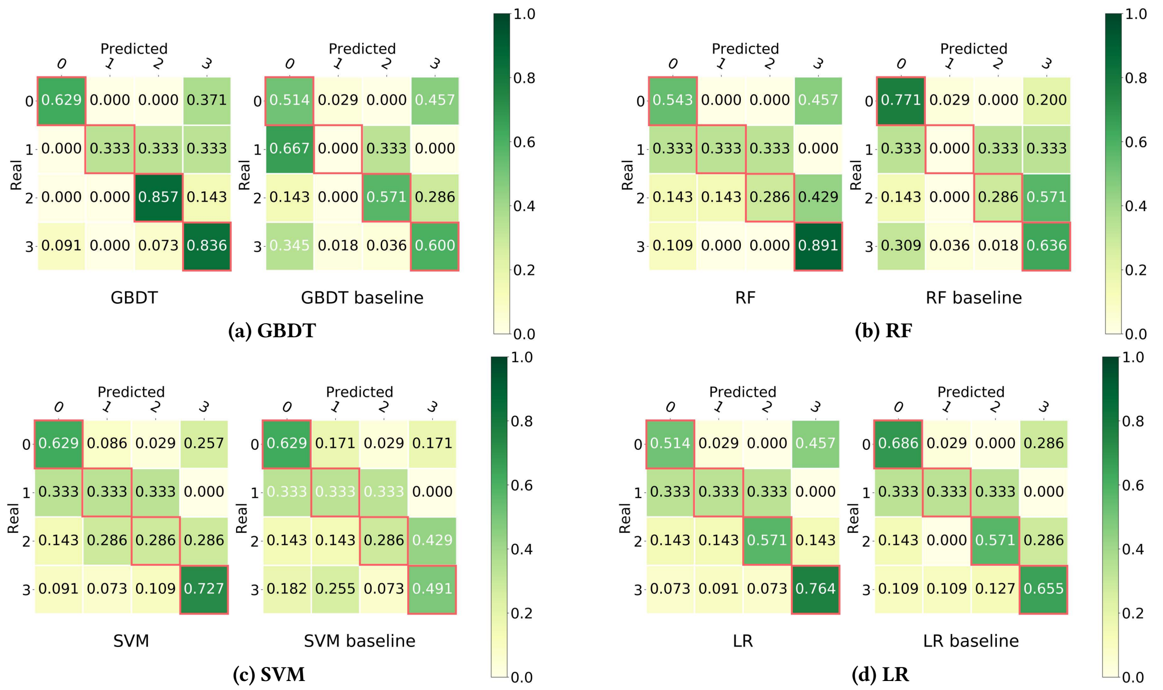

Figure 7: Predicted distribution heatmap of each score class with four algorithms on dataset ADD. The proportion of True Positive samples in each score class is in red box. In (a)-(d), the heatmap on the left shows the result of our proposed method and the right one shows the baseline method. ADD: area dimension question dataset.

involve drag-and-drop interactions. Moreover, the interaction features based on student mouse movement trajectories rely on the size of the interaction records. When there are too few student interaction records for a specific question, it will be difficult for us to accurately compare the similarity between a certain question and others using student interaction features. With more student interaction data being collected, the reliability of the proposed approach can be further improved. In addition, our method can be easily extended to other devices such as tablets with touch screens since the collected data has the same format (i.e., timestamp, event, and position).

7 CONCLUSION

Different from the extensively-studied student performance prediction in MOOCs platforms, student performance prediction in interactive online question pools can be more challenging due to the lack of knowledge tags and predefined question order or course curriculum. We proposed a novel method to boost student performance prediction in interactive online question pools by incorporating student interaction features and similarity between questions. We extracted new features based on student mouse movement trajectories to delineate problem-solving details of students. We also applied HIN to further consider students' historical problem-solving information on similar questions, as students' recent performance on similar questions can also be a good indicator of student future performance on a certain question. We conducted extensive experiments with the proposed method on the dataset collected from a real-world interactive online math question pool. Compared with using only the traditional statistic features (e.g., average scores), the proposed method achieved a much higher prediction accuracy across different models in different question classes. The results further confirm the effectiveness of our method.

In future work, we would like to combine the proposed method with adaptive question recommendation in interactive online question pools, providing different students with personalized online learning and question practice in online question pools. Also, it would be interesting to summarize different problem-solving patterns using mouse trajectories to better model and understand students' learning behaviors.

ACKNOWLEDGMENTS

The work is funded by HK ITF under grant ITS/388/17FP. Y. Wang is the corresponding author. Thank Trumptech (Hong Kong) Limited for the data support and Min Xu for proofreading.

REFERENCES

[1] Palakorn Achananuparp, Xiaohua Hu, Xiaohua Zhou, and Xiaodan Zhang. 2008. Utilizing Sentence Similarity and Question Type Similarity to Response to Similar Questions in Knowledge-sharing Community. In *Proceedings of QAWeb 2008 Workshop, Beijing, China.*

[2] Michèle Basseville, Igor V Nikiforov, et al. 1993. *Detection of Abrupt Changes: Theory and Application.* Vol. 104. Prentice Hall Englewood Cliffs.

[3] Hao Cen, Kenneth Koedinger, and Brian Junker. 2006. Learning Factors Analysis–A General Method for Cognitive Model Evaluation and Improvement. In *Proceedings of the International Conference on Intelligent Tutoring Systems.* Springer, 164–175.

[4] Delphine Charlet and Géraldine Damnati. 2017. SimBow at SemEval-2017 Task 3: Soft-Cosine Semantic Similarity between Questions for Community Question Answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017).* Association for Computational Linguistics, Vancouver, Canada, 315–319. https://doi.org/10.18653/v1/S17-2051

[5] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Int. Res.* 16, 1 (June 2002), 321–357. http://dl.acm.org/citation.cfm?id=1622407.1622416

[6] Qing Chen, Yuanzhe Chen, Dongyu Liu, Conglei Shi, Yingcai Wu, and Huamin Qu. 2015. Peakvizor: Visual analytics of peaks in video clickstreams from massive open online courses. *IEEE transactions on visualization and computer graphics* 22, 10 (2015), 2315–2330.

[7] Yuanzhe Chen, Qing Chen, Mingqian Zhao, Sebastien Boyer, Kalyan Veeramachaneni, and Huamin Qu. 2016. DropoutSeer: Visualizing Learning Patterns in Massive Open Online Courses for Dropout Reasoning and Prediction. In *Proceedings of the 2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*. IEEE, 111–120.

[8] Irene-Angelica Chounta and Paulo F. Carvalho. 2019. Square It Up!: How to Model Step Duration when Predicting Student Performance. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge (LAK19)*. ACM, New York, NY, USA, 330–334. https://doi.org/10.1145/3303772.3303827

[9] Albert T Corbett and John R Anderson. 1994. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-adapted Interaction* 4, 4 (1994), 253–278.

[10] Machine Learning: Feature Crosses. 2019. https://developers.google.com/machine-learning/crash-course/feature-crosses/video-lecture Accessed: 2019-9-20.

[11] Classification: ROC Curve and AUC. 2018. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc Accessed: 2019-9-20.

[12] Mucong Ding, Kai Yang, Dit-Yan Yeung, and Ting-Chuen Pong. 2019. Effective Feature Learning with Unsupervised Learning for Improving the Predictive Models in Massive Open Online Courses. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge (LAK19)*. ACM, New York, NY, USA, 135–144. https://doi.org/10.1145/3303772.3303795

[13] April Galyardt and Ilya Goldin. 2014. Recent-performance Factors Analysis. In *Proceedings of the 7th International Conference on Educational Data Mining*. 411–412.

[14] Josh Gardner, Christopher Brooks, and Ryan Baker. 2019. Evaluating the Fairness of Predictive Student Models Through Slicing Analysis. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge (LAK19)*. ACM, New York, NY, USA, 225–234. https://doi.org/10.1145/3303772.3303791

[15] Ilya M Goldin and April Galyardt. 2015. Convergent Validity of a Student Model: Recent-Performance Factors Analysis.. In *Proceedings of the 8th International Conference on Educational Data Mining*. 548–551.

[16] Stuart Hagler, Holly Brugge Jimison, and Misha Pavel. 2014. Assessing Executive Function Using A Computer Game: Computational Modeling of Cognitive Processes. *IEEE Journal of Biomedical and Health Informatics* 18, 4 (2014), 1442–1452.

[17] Jeff Heaton. 2016. An Empirical Analysis of Feature Engineering for Predictive Modeling. In *Proceedings of the SoutheastCon 2016*. IEEE, 1–6.

[18] Cheng Irene, Goebel Randy, and Basu Anup. 2010. *Multimedia in Education: Adaptive Learning and Testing*. World Scientific.

[19] Dorina Kabakchieva. 2013. Predicting Student Performance by Using Data Mining Methods for Classification. *Cybernetics and Information Technologies* 13, 1 (2013), 61–72.

[20] Tanja Käser, Nicole R. Hallinen, and Daniel L. Schwartz. 2017. Modeling Exploration Strategies to Predict Student Performance Within a Learning Environment and Beyond. In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference (LAK '17)*. ACM, New York, NY, USA, 31–40. https://doi.org/10.1145/3027385.3027422

[21] Jorge Maldonado-Mahauad, Mar Pérez-Sanagustín, Pedro Manuel Moreno-Marcos, Carlos Alario-Hoyos, Pedro J Muñoz-Merino, and Carlos Delgado-Kloos. 2018. Predicting Learners' Success in a Self-paced MOOC through Sequence Patterns of Self-regulated Learning. In *Proceedings of the 13th European Conference on Technology Enhanced Learning*. Springer, 355–369.

[22] Rubén Manrique, Bernardo Pereira Nunes, Olga Marino, Marco Antonio Casanova, and Terhi Nurmikko-Fuller. 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)*. ACM, New York, NY, USA, 401–410. https://doi.org/10.1145/3303772.3303800

[23] Pedro Manuel Moreno-Marcos, Carlos Alario-Hoyos, Pedro J Muñoz-Merino, and Carlos Delgado Kloos. 2018. Prediction in MOOCs: A Review and Future Research Directions. *IEEE Transactions on Learning Technologies* (2018).

[24] Kyle A O'Connell, Elijah Wostl, Matt Crosslin, T Lisa Berry, and James P Grover. 2018. Student Ability Best Predicts Final Grade in a College Algebra Course. *Journal of Learning Analytics* 5, 3 (2018), 167–181.

[25] Zachary A. Pardos and Neil T. Heffernan. 2011. KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model. In *Proceedings of the 19th International Conference on User Modeling, Adaption, and Personalization (UMAP'11)*. Springer-Verlag, Berlin, Heidelberg, 243–254. http://dl.acm.org/citation.cfm?id=2021855.2021877

[26] Philip I. Pavlik, Hao Cen, and Kenneth R. Koedinger. 2009. Performance Factors Analysis –A New Alternative to Knowledge Tracing. In *Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 531–538. http://dl.acm.org/citation.cfm?id=1659450.1659529

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

[28] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas Guibas, and Jascha Sohl-Dickstein. 2015. Deep Knowledge Tracing. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'15)*. MIT Press, Cambridge, MA, USA, 505–513. http://dl.acm.org/citation.cfm?id=2969239.2969296

[29] Chris Piech, Mehran Sahami, Jonathan Huang, and Leonidas Guibas. 2015. Autonomously Generating Hints by Inferring Problem Solving Policies. In *Proceedings of the Second (2015) ACM Conference on Learning @ Scale (L@S '15)*. ACM, New York, NY, USA, 195–204. https://doi.org/10.1145/2724660.2724668

[30] Huamin Qu and Qing Chen. 2015. Visual analytics for MOOC data. *IEEE computer graphics and applications* 35, 6 (2015), 69–75.

[31] Mary Budd Row. 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 (1974), 81–94.

[32] Antonio A Sánchez-Ruiz, Guillermo Jimenez-Diaz, Pedro P Gómez-Martín, and Marco A Gómez-Martín. 2017. Case-Based Recommendation for Online Judges Using Learning Itineraries. In *Proceedings of the 25th International Conference on Case-Based Reasoning*. Springer, 315–329.

[33] Adriana Seelye, Stuart Hagler, Nora Mattek, Diane B Howieson, Katherine Wild, Hiroko H Dodge, and Jeffrey A Kaye. 2015. Computer Mouse Movement Patterns: A Potential Marker of Mild Cognitive impairment. *Alzheimer's & Dementia: Diagnosis, Assessment & Disease Monitoring* 1, 4 (2015), 472–480.

[34] Wanpeng Song, Min Feng, Naijie Gu, and Liu Wenyin. 2007. Question Similarity Calculation for FAQ Answering. In *Proceedings of the Third International Conference on Semantics, Knowledge and Grid (SKG 2007)*. IEEE, 298–301.

[35] Robert J Stahl. 1994. *Using "Think-time" and "Wait-time" Skillfully in the Classroom.* ERIC Clearinghouse.

[36] Yizhou Sun and Jiawei Han. 2013. Mining Heterogeneous Information Networks: A Structural Analysis Approach. *SIGKDD Explor. Newsl.* 14, 2 (April 2013), 20–28. https://doi.org/10.1145/2481244.2481248

[37] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based Top-K Similarity Search in Heterogeneous Information Networks. *Proceedings of the VLDB Endowment* 4, 11 (2011), 992–1003.

[38] Khushboo Thaker, Paulo Carvalho, and Kenneth Koedinger. 2019. Comprehension Factor Analysis: Modeling Student's Reading Behaviour: Accounting for Reading Practice in Predicting Students' Learning in MOOCs. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge (LAK19)*. ACM, New York, NY, USA, 111–115. https://doi.org/10.1145/3303772.3303817

[39] Meng Xia, Mingfei Sun, Huan Wei, Qing Chen, Yong Wang, Lei Shi, Huamin Qu, and Xiaojuan Ma. 2019. PeerLens: Peer-inspired Interactive Learning Path Planning in Online Question Pool. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 634, 12 pages. https://doi.org/10.1145/3290605.3300864

[40] Xiaolu Xiong, Zachary A Pardos, et al. 2011. An Analysis of Response Time Data for Improving Student Performance Prediction. (2011).

[41] Aman Yadav, Hai Hong, and Chris Stephenson. 2016. Computational Thinking for All: Pedagogical Approaches to Embedding 21st Century Problem Solving in K-12 Classrooms. *TechTrends* 60, 6 (2016), 565–568.

[42] Takashi Yamauchi. 2013. Mouse Trajectories and State Anxiety: Feature Selection with Random Forest. In *Proceedings of the 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction (ACII '13)*. IEEE Computer Society, Washington, DC, USA, 399–404. https://doi.org/10.1109/ACII.2013.72

[43] Raciel Yera Toledo, Yailé Caballero Mota, and Luis Martínez. 2018. A Recommender System for Programming Online Judges Using Fuzzy Information Modeling. *Informatics* 5(2), 17 (2018).

[44] Chun-Kit Yeung and Dit-Yan Yeung. 2018. Addressing Two Problems in Deep Knowledge Tracing via Prediction-consistent Regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale (L@S '18)*. ACM, New York, NY, USA, Article 5, 10 pages. https://doi.org/10.1145/3231644.3231647

[45] Hsiang-Fu Yu, Hung-Yi Lo, Hsun-Ping Hsieh, Jing-Kai Lou, Todd G McKenzie, Jung-Wei Chou, Po-Han Chung, Chia-Hua Ho, Chun-Fu Chang, Yin-Hsuan Wei, et al. 2010. Feature Engineering and Classifier Ensemble for KDD Cup 2010. In *KDD Cup 2010*.