RiPLE: Recommendation in Peer-Learning Environments Based on Knowledge Gaps and Interests

Hassan Khosravi
University of Queensland
h.khosravi@uq.edu.au

Kirsty Kitto
University of Technology Sydney
kirsty.kitto@uts.edu.au

Kendra Cooper
Independent Scholar
kendra.m.cooper@gmail.com

Various forms of Peer-Learning Environments are increasingly being used in post-secondary education, often to help build repositories of student generated learning objects. However, large classes can result in an extensive repository, which can make it more challenging for students to search for suitable objects that both reflect their interests and address their knowledge gaps. Recommender Systems for Technology Enhanced Learning (RecSysTEL) offer a potential solution to this problem by providing sophisticated filtering techniques to help students to find the resources that they need in a timely manner. Here, a new RecSysTEL for Recommendation in Peer-Learning Environments (RiPLE) is presented. The approach uses a collaborative filtering algorithm based upon matrix factorization to create personalized recommendations for individual students that address their interests and their current knowledge gaps. The approach is validated using both synthetic and real data sets. The results are promising, indicating RiPLE is able to provide sensible personalized recommendations for both regular and cold-start users under reasonable assumptions about parameters and user behavior.

1. Introduction

The importance of peer-learning in post-secondary education is being increasingly recognized (Boud et al., 2014). This has led to the creation of a number of Peer-Learning Environments that are claimed to perform many different roles. They are often designed to: engage and satisfy students by instilling ownership; help build communities and recognize participation; and can provide rich, timely peer-generated feedback (Betts, 2013; Coetzee et al., 2015; Wright et al., 2015).

For example, PeerWise (Denny et al., 2008) is a free web-based system in which students can create multiple-choice questions as well as answer, rate, and discuss questions created by their peers. Empowering students with environments like these offers significant benefits, as they enhance student involvement in cognitively demanding tasks rather than the more passive answering of questions. Thus, students using PeerWise are required to identify missing knowledge, diagnose misconceptions, and provide feedback to their peers in their own words. Such tasks all ultimately enhance student learning (Chin and Brown, 2002; Rosenshine et al., 1996; Hardy et al., 2014). However, as the class size of a PeerWise instance grows, so does the number of available questions on the platform. This makes it more challenging for students to select questions that best suit their current learning needs. Will students select questions that fill their current knowledge gap? Or will they just select those easy questions that make them feel like they have mastered the material that they should be learning? To thrive in learning environments of this form, students need to be able to identify the characteristics of questions that will be both interesting and the most beneficial for their current knowledge needs. However, students often lack the requisite skills for making good decisions about what and how to study (Beswick et al., 1988; Biggs, 1999), which can leave them undirected and time wasted.

Recommender systems (RecSys; Ricci et al. 2011) offer a potential solution to problems of overwhelming choice by providing sophisticated filtering techniques to help people find the resources that they need. Specifically, as the field of recommender systems for technology enhanced learning (RecSysTEL; Manouselis et al. 2011) evolves it becomes possible to analyze the digital traces left by learners in these environments and use them to provide recommendations about resources that will most meet their learning needs and interests.

Here, a novel RecSysTEL solution is presented that helps students to navigate in complex Peer-Learning Environments, specifically those that deal with question answering in the form provided by PeerWise. An examination in Section 2. of related work in profiling student knowledge and RecSysTEL demonstrates that there is reason to believe that progress can be made by combining more sophisticated learner profiles with RecSysTEL solutions. In Section 3. the techniques and technologies used in the solution, RiPLE, are introduced in addition to the problem statement. The solution itself is presented in Section 4., and a simple example demonstrating how RiPLE works is in Section 5. Experimental validation and results are reported in Sections 6. and 7., followed by conclusions and a discussion of future work in Section 8.

2. Related Work

2.1. Recommendation Systems for Technology Enhanced Learning

RecSysTEL is an active and rapidly evolving research field. For example, Drachsler et al. (2015) perform an extensive classification of 82 different RecSysTEL environments, and Erdt et al. (2015) review the various evaluation strategies that have been applied in the field. Together these articles provide recent comprehensive surveys that consider more than 200 articles spanning over 15 years. Here, the focus is on collaborative filtering (CF), which identifies similar users and provides recommendations based upon their usage patterns. CF has been extensively employed in RecSysTEL, and an early LAK paper, Verbert et al. (2011) evaluated and compared the performance of different CF techniques on educational data sets, showing that the best choice of algorithm is data dependent. In a more recent study, Kopeinik et al. (2017) also concluded that the performance of the algorithms strongly depends on the properties and characteristics of the particular dataset. In combining educational data sets with social networks, Cechinel et al. (2013) used CF to predict the utility of items for users based on their interest and the interest of the network of the users around them. Similarly, Fazeli et al. (2014) proposed a graph-based approach that uses graph-walking for improving performance on educational data sets.

One important way in which RecSysTEL has been used in an educational setting is to recommend personalized learning objects. Thus, Lemire et al. (2005) used inference rules to provide context aware recommendation on learning objects, and Mangina and Kilbride (2008) recommend documents and resources within e-learning environments to expand or reinforce knowledge. Interestingly, Gomez-Albarran and Jimenez-Diaz (2009) combined content based filtering with collaborative filtering to make recommendations in a student authored repository. When recommending learning objects (e.g. questions) related to student knowledge gaps, Cazella et al. (2010) provided a semi-automated, hybrid solution based on CF (nearest neighbor) and rule based filtering, while Thai-Nghe et al. (2011) used students' performance predictions to recommend more appropriate exercises. CF techniques (basic, biased, and tensor matrix factorization) were used to address a number of different student behaviors and to model the temporal effect of students improving over time. Recently, Imran et al. (2016) provided an automated solution to personalize a learning management systems (LMS) using advanced learners' profiles to encapsulate their expertise level, prior knowledge, and performance in the course. The approach used association rule mining to create the learning object recommendations.

Matrix factorization (MF) is one of the most established techniques used in CF; however, despite its success in RecSys, MF has rarely been used in RecSysTEL. Out of the 124 papers referenced by Drachsler et al. (2015), only two papers directly use it (Salehi, 2013; Thai-Nghe et al., 2011). In Section 2.2. the discussion reveals that this is somewhat surprising; MF has been put to good use in EDM for generating latent profiles of student expertise and so ought to combine with RecSysTEL in a straightforward manner. Indeed, the intelligent tutoring systems that appear to be utilized in that body of work could be seen as closely related to RecSysTEL, although they tend to give students less autonomy to accept or reject the pathways chosen for them (Chen, 2008).

MF is particularly powerful in modeling students' performance and knowledge, because it implicitly incorporates guess and slip factors as latent factors (Thai-Nghe et al., 2011). In a question answering scenario slipping refers to the situation where a student has the required skill for answering a question but mistakenly provides the wrong answer; guessing refers to the situation where a student provides the right answer despite not having the required skill for solving the problem. This is a complex task that has received significant attention in the EDM community (Beck and Chang, 2007; Baker et al., 2008; Pardos and Heffernan, 2010).

2.2. Learner Profiling Using the Q-matrix

In contrast to the RecSysTEL literature an ongoing program of research, spanning more than 30 years, has sought to build models of student competencies and underlying knowledge, mapping them to educational tasks. An early EDM paper by Barnes (2005) discusses the *Q-matrix* approach, which maps test item results to latent or underlying knowledge structures. This mapping was originally performed using binary values, although these values can be straightforwardly mapped to probabilities if the binary values are replaced by a number ranging from zero to one. Thus, this approach constructs concept–question matrices that can be related to the performance of students using a variety of MF methods. Recent work by Desmarais and collaborators has made use of non-negative matrix factorization (NMF) to extract Q-matrices from different data sets (Desmarais et al., 2012; Desmarais, 2012; Desmarais and Naceur, 2013), claiming NMF is far more interpretable than many MF techniques due to its insistence upon non-negative values in the two new matrices, which enables a probabilistic interpretation of the resulting matrices.

The model of MF that is adopted very much affects the results that are obtained. In particular, the move from compensatory operations (which each added skill adds to the success of a topic) to more conjunctive operators (where missing skills will lead to a student failing a test item) has been recognized (Barnes, 2010; Desmarais et al., 2012; Desmarais, 2012), but there is no clear consensus as to which factorization method should be used. Indeed, it is possible to factorize matrices describing student performance in other ways, and in Section 3.1. one such alternative is presented.

While originally constructed by experts who defined the question to concept mappings, Q-matrices can be automatically constructed using simple hill-climbing algorithms which vary the number of concepts and the values in randomly seeded matrices, attempting to find a Q-matrix that best describes all student responses. In contrast to results obtained by traditional clustering methods, Q-matrices are more interpretable, which makes them interesting tools for communicating with both faculty and students about capabilities and weaknesses. The paper by Barnes (2005) demonstrated that, in at least some cases, students who were given a self-guided option in an experiment could choose questions that were highly correlated with a Q-matrix "least understood concept" constructed from a simple lesson based tutorial. Furthermore, Barnes was able to demonstrate that a small sample of self-guided students who chose differently from the Q-matrix prediction "could have benefited from reviewing a Q-matrix selected concept" before their final exam, stating correctly that a "student may not realize when he should review a particular topic." Sometimes the items in which a student is most interested are not those from which they could best benefit. This suggests that RecSysTEL can perhaps be used to improve outcomes based upon profiles of student knowledge, particularly in more complex scenarios where student confusion is likely to increase.

Q-matrices have been shown to compare favorably with Bayesian Knowledge Tracing (BKT) when it comes to predicting student success (Thai-Nghe et al., 2010), but remain very difficult to use in scenarios based more around modeling student knowledge of topics. They tend to perform better when concepts and topics are distinct from one another, as happens with e.g. French and mathematics, but less well on trivia (or questions for which there is more overlap) (Winters, 2006; Desmarais, 2012).

2.3. Using Knowledge Gap Profiles in RecSysTEL

While many recommendation systems have been developed in TEL, they tend not to make use of MF for their profiling of students. Similarly, there appear to be few attempts to couple student profiles regarding knowledge with a scalable RecSys solution. Here, a full system is presented that: (i) Takes note of student performance in a real world and open ended question answering scenario, (ii) constructs a learner profile based upon performance using MF that maps out their current knowledge gaps with respect to the environment in which they are participating, and (iii) recommends questions that will help them to remove this knowledge gap, while preferentially selecting questions that are similar to those that they have previously rated as interesting.

3. RiPLE Techniques, Technologies, and Problem Definition

In this section, more explicit details are provided on the MF algorithm and the Peer-Learning Environment that are used in this study. These details enable the definition of a tighter set of design requirements and the further refinement of the research problem in Section 3.3.

3.1. Matrix Factorization

Assuming $H_{N\times K}$ represents the latent factors underlying user behavior giving h_u, a vector of latent factors representing user u. Similarly, $Q_{M\times K}$ is assumed to represent the latent factors of a question set, where q_i is a vector of latent factors representing question i. After the mapping of users and questions to the latent factors, the rating of a user u for a question i can be approximated as:

$$\hat{r}_{ui} = q_j^{\mathsf{T}} h_u = \sum_{k=1}^{K} q_{ik} h_{uk} \tag{1}$$

Matrix $\hat{R} = \{\hat{r}_{ui}\}$ is then used to capture all predicted ratings that users give a set of questions, with elements given by Equation 1. The goal of MF is to learn the matrices H and Q, which are used to compute values for $\hat{R}$, which approximate the unseen ratings that are actually given by users represented by R. To learn these factors, a MF system minimizes the following regularized squared error term on the set of known ratings:

$$\sum_{(u,i)\in R_{train}} (r_{ui} - q_i^{\mathsf{T}} h_u)^2 + \lambda(\|q_i\|^2 + \|h_u\|^2), \tag{2}$$

where $(u, i) \in R_{train}$ represents (u, i) pairs such that the rating of user u for question i is present in the training data set and λ is a parameter controlling the extent of the regularization.

The initial values of latent variables in H and Q are sampled from a standard normal distribution with zero mean and standard deviation of one. By performing stochastic gradient descent, in each iteration looping through the ratings in R, latent variables in H and Q are updated using Formulas (3) and (4) and tuned in order to locally minimize (2). The constant value γ represents the learning rate, which is often determined using a validation set.

$$h_u = p_u + \gamma.((r_{ui} - q_i^{\mathsf{T}} h_u).q_i - \lambda.h_u) \tag{3}$$

$$q_i = q_i + \gamma.((r_{ui} - q_i^{\mathsf{T}} h_u).h_u - \lambda.q_i) \tag{4}$$

Extended research has aimed to improve this method generally. Koren (2008) illustrated that addition of mean normalization and a bias parameter for each user and item (in this case a question) can capture the effects associated with each, allowing only the true interaction portion of the ratings to be modeled in H and Q. This method, referred to as **Biased Matrix Factorization (BMF)**, is employed in RiPLE.

3.2. The PeerWise Learning Environment

PeerWise (Denny et al., 2008) is a free web-based system in which students can both (i) create multiple-choice questions for sharing, and (ii) answer, rate, and discuss questions created by their peers. More than 1500 universities, schools and technical institutes from around the world have adopted Peerwise[1], and many papers have been published that discuss research completed using the platform (Hardy et al., 2014; Lumezanu et al., 2007; Bates et al., 2012; Purchase et al., 2010).

In PeerWise students are expected to direct their questions towards the learning goals of the course. Students receive immediate feedback on any answers that they record in the system.

[1] https://peerwise.cs.auckland.ac.nz/

They are also shown a sample solution and data about how other students have answered the same question. This helps them to assess how well they are performing compared to their peers. Questions can also be evaluated using peer-review, which encourages students to evaluate the quality and difficulty of any questions they answer, providing constructive open-ended comments in the process if appropriate. This feature enables asynchronous discussions over a period of time, where students can rate the quality of questions, providing feedback for one another as to how they might be improved.

The crowdsourcing process facilitated by this Peer-Learning Environment can lead to a repository of rich and high-quality multiple-choice questions that can be reused in future offerings of a course, as well as studied in their own right.

PeerWise also includes several "game-like" elements (such as badges, points and leaderboards) to inspire students to become more engaged with the platform. All activities remain anonymous to students; however, instructors can view the identity of question and comment authors, and to delete inappropriate questions. When students create a question, they can tag it with relevant topics, which can be student generated depending upon the settings chosen by an instructor. Instructors can also choose to predefine all tags to be used in the course if they feel that student generated tags will not work for their scenario.

PeerWise currently does not provide personalized recommendations to students. However, the main PeerWise page where the questions are presented supports basic sorting functionality. Questions can be sorted based on different characteristics such as popularity, difficulty, and date of creation. A student can then manually search through the displayed questions to find suitable candidate questions for answering. Additional information about the reputation of the author and the number of times the question has previously been answered is also provided.

3.3. The Research Problem

The open-ended structure of PeerWise leads to the specific research problem that this study aims to address: as a large and unstructured store of multiple choice questions, PeerWise can rapidly become un-navigable for students. This can lead to students focusing upon questions that reinforce existing knowledge, or satisfy their general interests, instead of those that are most likely to help them to satisfy study requirements. A RecSys could be used to discourage this behavior, but such a system must be able to both identify knowledge gaps in an individual learner's profile, and find questions that are most likely to satisfy that knowledge gap. Ideally such a RecSys would be able to perform this function while prioritizing questions in which a user is likely to have an interest, as this will help to maintain their engagement with the system.

The aim of this work is to enhance a RecSys using BMF with the concept of a learning profile, producing a RecSySTEL designed for the PeerWise learning environment. As a set of further requirements this tool should: enable a proof of concept scenario where users can choose different foci for the recommendations that they receive, support cold start users, scale appropriately while exhibiting robust behavior, and allow users to understand the reason for the recommendations presented.

4. Introducing RiPLE

At a high level, RiPLE[2] applies a suite of established approaches to harness data available in Peer-Learning Environments and provide personalized recommendations tailored towards each users' interests and knowledge gaps. RiPLE is organized into five main modules: Input Data, Data Integration, Learning Profile, Recommendation Engine, and Modes of Operation. Figure 1 provides an overview of the system. Boxes in the Input Data module represent data gathered from PeerWise. The top part of the double boxes in the Data Integration, Learning Profile, and Recommendation Engine modules represent computations; the bottom part of the double boxes represent the results. The boxes in the Modes of Operation module represent the final selection and presentation of tailored recommendations to the users based on the operational mode selected. A summary of the notation used in describing RiPLE is presented in Table 1.

In what follows, Section 4.1. provides more information about the input data, and Section 4.2. discusses how the data are used to infer knowledge gaps. Section 4.3. introduces the learning profile, Section 4.4. describes how learning profile enhanced recommendations are made, and Section 4.5. summarizes the different operational modes of the system.

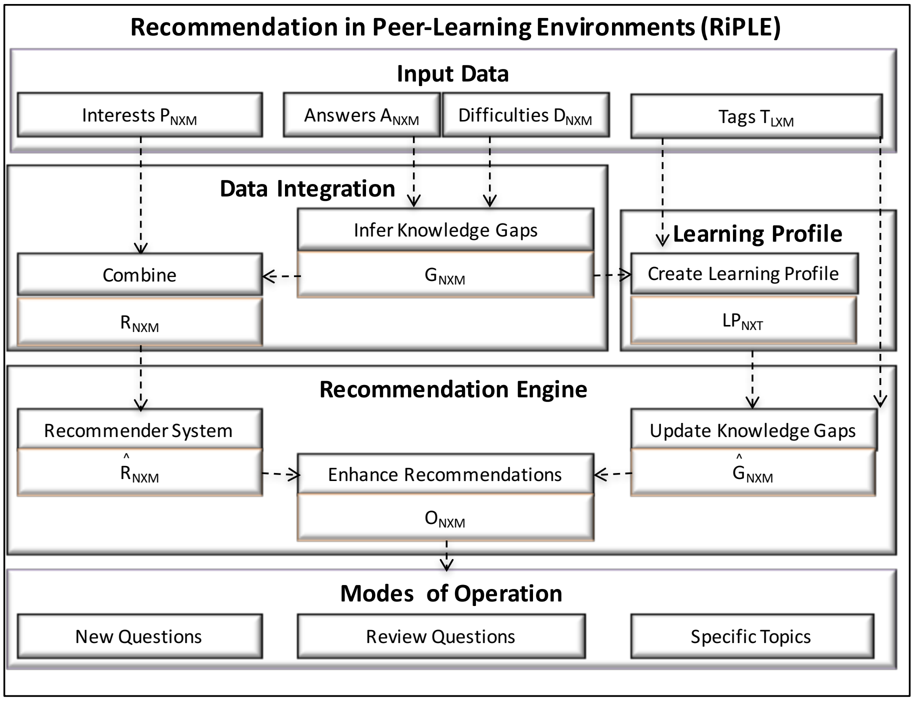


Figure 1: RiPLE: a framework for Recommendation in Peer-Learning Environments.

4.1. Input Data

As discussed in Section 3.2., the input data consists of multiple-choice questions that are tagged with distinct topics and user ratings for quality and difficulty. Table 1 summarizes all input data, characterizing it with N to denote the number of users that are registered on the Peer-Learning Environment, M for the number of multiple-choice questions that have been contributed to the

[2] Source code for RiPLE is available at https://github.com/hkhosrav/RiPLE

Input Data	
N	Number of users
M	Number of questions
L	Number of topics
$A_{N \times M}$	Matrix, where a_{ui} is 1 if user u answers question i correctly and 0 if answered incorrectly
$P_{N \times M}$	Matrix, where p_{ui} is the interest level user u has expressed for question i
$D_{N \times M}$	Matrix, where d_{ui} is the difficulty level user u has expressed for question i
$T_{M \times L}$	Matrix, where t_{ij} is $\frac{1}{g}$ if i is tagged with g topics, including j and 0 otherwise
Data Integration	
$G_{N \times M}$	Matrix, where g_{ui} is the knowledge gap of user u based on question i
$\bar{d}_M$	Vector, where $\bar{d}_i$ is the average difficulty expressed for question i across all of the users
kgw	Constant to weight the relative impact of knowledge gaps and interests
$R_{N \times M}$	Matrix, where r_{ui} is the benefit that user u would receive from doing question i
Learning Profile	
$S_{N \times M}$	Matrix, where s_{ui} is 1 if user u has attempted question i and 0 otherwise
$C_{N \times L}$	Matrix, where c_{uj} is the weighted sum of questions tagged with topic j attempted by user u
$LP_{N \times L}$	Matrix, where lp_{uj} is the approximated knowledge gap of user u on topic j
$\bar{lp}_L$	Vector, where $\bar{lp}_j$ is the average knowledge gap for topic j across all of the users
β	constant parameter controlling the impact of the learning profile
Recommendation Engine	
$O_{N \times M}$	Matrix, where o_{ui} is the predicted personalized score of question i for user u
$\hat{R}_{N \times M}$	Matrix, where $\hat{r}_{ui}$ is the predicted benefit that user u would receive from doing question i
$\bar{r}_M$	Vector, where $\bar{r}_i$ is the average benefit of question i across all of the users
$\hat{G}_{N \times M}$	Matrix, where $\hat{g_{ui}}$ is the inferred knowledge gap of user u on question i based on lp_u

Table 1: A summary of the notation used in describing RiPLE

environment, and L as the number of distinct topics that have been used to tag the questions. The expressed user ratings and correctness values in a PeerWise data set are used to populate the matrices. The interests (P) are originally stored in PeerWise as integers in the range of [0, 5] and the difficulties (D) as integers in the range of [0, 2]. Data input to RiPLE from PeerWise are organized in four matrices:

Interests, P**:** Each user can rate the quality of the questions that they have answered. This information is represented in a matrix $P_{N \times M}$, where p_{ui} captures the interest level user u has expressed for question i. These ratings are stored as a value between between 0 and 1 when expressed and as $Null$ otherwise.

Difficulties, D**:** Each user can rate the difficulty of the questions that they have answered. This information is represented in a matrix $D_{N \times M}$, in which d_{ui} captures the difficulty level user u has expressed for question i. These ratings are stored as a value between 0 and 1 when expressed and as $Null$ otherwise.

Tags, T**:** Each question can have 0 to L topics assigned (i.e. tagged) to it. The information on topics assigned to each question is represented in a matrix $T_{M \times L}$, in which $t_{ij} = 0$ indicates that question i is not tagged with topic j and $t_{ij} = \frac{1}{g}$ indicates that question i is tagged with $1 \leq g \leq L$ associated topics, including j.

Answers, A**:** The correctness of the answers provided by the users is represented in a matrix $A_{N \times M}$. If a user u answers a question i correctly then the matrix entry is set at $a_{ui} = 1$,

$a_{ui} = 0$ indicates an incorrect answer, and $a_{ui} = Null$ indicates that question i has not been attempted by user u.

4.2. Data Integration

This module uses the input data A, D, and P to produce an overall rating matrix $R_{N\times M}$, where r_{ui} captures the extent to which a user u would benefit from answering a question i. This matrix represents how much the knowledge gaps of individual users can be reduced while keeping engagement at a maximum. $R_{N\times M}$ is constructed in two steps.

Inferring Knowledge Gaps First, information from D and A is combined to create a scoring function that maps user performance to knowledge gaps. The function determines user u's lack of knowledge about question i, independent of their performance on other questions. Matrix D is used for computing a vector $\bar{d}$, where $\bar{d}_i = \frac{\sum_{u=1}^{N} d_{ui}}{N}$ represents the average rating for question i across all users. The scoring function produces a matrix $G_{N\times M}$, where g_{ui} infers user u's lack of knowledge about question i.

$$g_{ui} = (1 - a_{ui})(\frac{0.5 - a_{ui}}{1+ \bar{d}_i}) + a_{ui}(\frac{0.5 - a_{ui}}{2- \bar{d}_i}) \tag{5}$$

A higher value for (5) indicates a larger knowledge gap. This function captures two intuitions about user responses:

1. An incorrect answer indicates a knowledge gap for topics related to that question. The significance of the gap may be approximated by the difficulty level of the question; answering an easy question incorrectly suggests a large gap and answering a hard question incorrectly suggests a smaller one.

2. Answering questions correctly provides evidence that no knowledge gap exists, suggesting user competency on the related topics. The significance of the competency may be approximated by the difficulty level of the question; answering an easy question correctly illustrates low level of competency and answering a hard question correctly illustrates a higher level of competency.

Equation (5) uses summation to combine these intuitions. The first part of the equation is positive, indicating a knowledge gap for an incorrectly answered question i weighted by $\hat{d}_i$. The second part contributes to the score with a negative value, indicating competencies, when the question is answered correctly. For example, answering an easy question i with $\bar{d}_i = 0.1$ incorrectly results in the scoring function returning 0.45, and answering a hard question i' with $\bar{d}_{i'} = 0.7$ correctly results in the scoring function returning -0.38. Given that the difficulties are stored as values between 0 and 1, values in G remain in the range of [-0.5, 0.5].

Combining Knowledge Gaps and User Interests The knowledge gaps inferred from the previous calculation are then combined with students' interests to produce a matrix R, which captures the extent to which users would benefit from answering different questions:

$$R = kgwG + (1 - kgw)P. \tag{6}$$

Here, a weight term ($0 \leq kgw \leq 1$) is used to represent the impact of the knowledge gaps upon users. This may be be set as default for an entire cohort, or for individual students. Both students and instructors could set kgw at the individual level, or additional machine learning techniques could also be used in future work. The value of kgw allows the weight of the knowledge gaps and the interests of the students to be adjusted along a spectrum between what the users of RiPLE need (to master course content) and what they prefer (to improve engagement).

4.3. Learning Profile

This module uses the input data T and the knowledge gap matrix G provided by Data Integration module to produce a student-topic learning profile $LP_{N \times L}$, in which each vector (lp_u) approximates a user's knowledge gaps across all topics associated with the course. A negative value in the vector, i.e. $lp_{uj} < 0$ indicates that the user u has demonstrated some knowledge on topic j, a positive value indicates a knowledge gap on that topic, and 0 represents a neutral state, where the positive and negative scores have balanced each other out for that topic. The learning profile is computed in two steps:

1. Matrix $G_{N \times M}$ stores information about the lack of knowledge exhibited by all users for each question, and matrix $T_{M \times L}$ stores information about the tags associated with each question. Multiplying the two (GT) allows for an understanding to be gained about topic-level knowledge gaps in the system *per se*. The value stored in cell $[u, j]$ of the resulting matrix depends on the number and weight of questions tagged with topic j that have been attempted by each user u. This means that the values in this matrix require normalization.
2. Normalization is achieved using a user-topic count matrix $C_{N \times L}$, in which c_{uj} represents the weighted sum of questions attempted by user u that have been tagged with topic j. This matrix can be computed using $C = ST$, in which s_{ui} is 1 if question i is attempted by user u and 0 otherwise.

Putting the two steps together, the learning profile is computed using the following formula:

$$LP = \frac{GT}{C} \tag{7}$$

This learning profile may be shared with students to inform them of their knowledge gaps and competencies at a topic-level, enabling them to understand the reason for RiPLE's recommendations. As mentioned in Section 3.3., this is one of the core requirements of the system. It would also allow them to compare their performance with the cohort. Such learner centered learning analytics could help to stimulate self-reflection among students, as well as providing an early alert for those that are performing below their targeted performance goal. Future work will seek to explore this intriguing possibility.

Cold-start users A user that has answered zero or very few questions is referred to as a cold-start user. The knowledge gaps of a cold-start user c that has answered zero questions are represented with a vector of zeros. In this scenario, the system is unable to reliably infer user c's knowledge gaps, and therefore, cannot make meaningful recommendations. To address this issue, the knowledge gaps for cold-start users are estimated using the average knowledge gaps for the cohort, a vector $\bar{lp}$, where $\bar{lp}_j = \frac{\sum_{u=1}^{N} lp_{uj}}{N}$ represents the average knowledge gap for topic j across all of the users in LP. This solution to the cold start problem assumes that most new users will have similar knowledge gaps to the average user, an assumption that could be questioned, but which should result in better initial recommendations for most users in the system (by definition).

4.4. Recommendation Engine

This module uses the benefit matrix R (produced by the Data Integration module) and learning profile matrix LP (produced by the Learning Profile module) to produce a matrix O which contains vectors o_u predicting the extent to which user u would benefit from each of the questions in the PeerWise system. The process of making these recommendations is accomplished in three steps. Again, cold start users require unique processing for this module (see below).

Exemplary Recommendation First, matrix factorization as described in Section 3.1. is employed to characterize users and questions using vectors of latent factors that form $\hat{R}_{N \times M}$. This predicts the extent to which users might benefit from completing unseen questions.

Updating the student-question knowledge gap This step uses the matrices LP (produced by the Learning Profile module) and the input tag matrix T to produce an updated student-question matrix $\hat{G}_{N \times M}$, in which $\hat{g}_{ui}$ approximates user u's knowledge gap of question i based on lp_u and the tags associated with i. This is accomplished using the following equation:

$$\hat{G} = LPT^{\mathsf{T}} \tag{8}$$

Multiplying $LP_{N \times L}$ and $T^{\mathsf{T}}_{L \times M}$ propagates the lack of knowledge from course topics over to the associated questions.

Enhancing Recommendations The updated benefit matrix $\hat{R}$ and the updated student-question matrix $\hat{G}$ that were extracted in the previous two steps are used to create the recommendation output matrix O, in which o_{ui} represents the personalized rating of question i for user u tailored towards their knowledge gaps and interests. Values in O are computed using the following formula:

$$O = \hat{R} + \beta \hat{G} \tag{9}$$

where β is a parameter controlling the impact of the learning profile, which may be determined using a validation set.

Cold-start Users The regularized squared error used in matrix factorization sets the latent factors of a user u based on two terms: minimizing the first term tunes the latent factors of u for predicting the ratings in the training set and minimizing the second term helps keep the latent factors small to avoid over-fitting (see Equation (2)). Since cold-start users do not have any ratings in the training set, the first term does not affect the outcome, so the learning algorithm is encouraged to reduce the error rate of the cost function by setting the latent factors all to zero without paying a penalty on the first term. Since multiplying a vector of zeros by the latent factors of any question returns zero, the system is unable to make any meaningful recommendation for cold-start users.

One possible solution for overcoming this problem is to use mean normalization. Let $\bar{r}$ be a vector storing the average rating for each question, so $\bar{r}_i$ represents the average rating for question i across all users in R. During the learning phase, values in R are normalized with $\bar{r}$ using the following formula:

$$r_{ui} = r_{ui} - \bar{r}_i. \tag{10}$$

With this update, after the learning phase, values in R have the following interpretation: $r_{ui} > 0$ indicates that u would rate i higher than average, $r_{ui} < 0$ indicates that u would rate i lower than average, and $r_{ui} \simeq 0$ indicates that u would rate i close to the average. Using mean normalization has the benefit that the system's ratings for a cold-start user c, which is $r_c = \{0\}$, has now the interpretation that c's rating of each of the questions is the global average for that question.

After the learning phase, values are de-normalized and stored back in $\hat{R}$ using the following formula, in which $\bar{r}_i$ is added back to the ratings for question i

$$\hat{r}_{ui} = r_{ui} + \bar{r}_i. \tag{11}$$

4.5. Modes of Operation

The system operates in three different modes, each having its own advantages and use case. The modes select questions to present to the user u from their vector in the output matrix O_u (produced by the Recommendation Engine).

Exploring new questions: In this mode, the system is designed to present users with questions that they have not seen before, preferentially choosing the unseen questions with the highest recommendation values for the user u, in the vector O_u. This mode is ideal for general practice, allowing users to explore new questions that are tailored towards their interests and reducing their knowledge gaps.

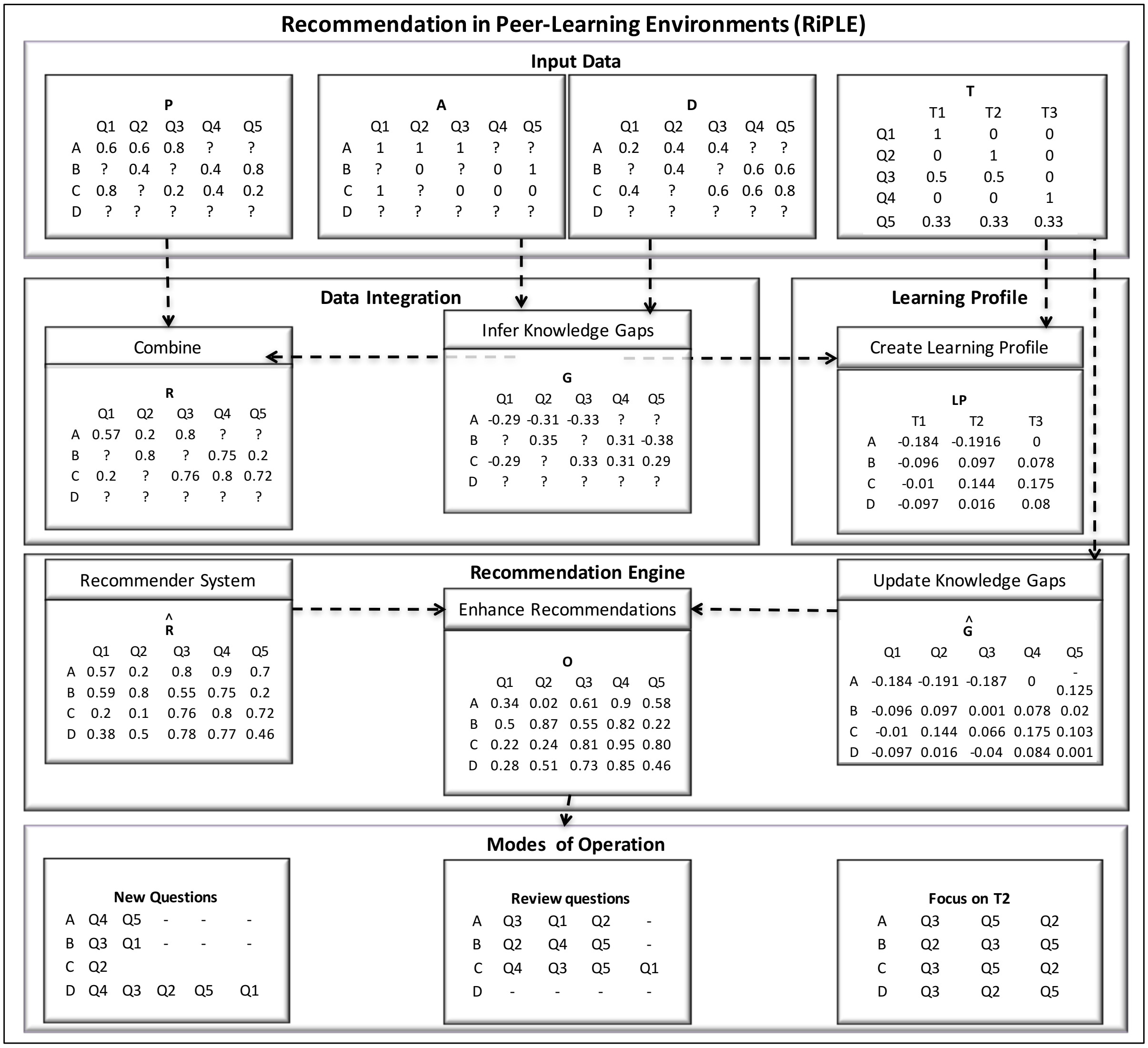
Reviewing answered questions: In this mode, the system is designed to present users with questions that they have seen before, preferentially choosing the seen (answered) questions with the highest recommendation values for the user u, in the vector O_u. This mode is ideal for preparation for exams; the system prioritizes questions that cover topics where the user lacks knowledge and topics in which they are interested.

Focusing on specific topics: In this mode, the system is designed to present the user with questions from selected topic(s), regardless of whether they were previously attempted or not, choosing those which have the highest recommendation values for the user u, in the vector O_u. This mode is ideal for practice on specific topics, in which the system prioritizes questions that the user finds most interesting and helpful in reducing their knowledge gaps.

5. Simple Example

To ground the above discussion of RiPLE, a simplified example with four students, five questions and three topics is presented. Figure 2 shows an overview of the example based on the framework provided in Figure 1. In this example, Alice (A), Bob (B), and Catherine (C) are all defined as existing active users and Dean (D) is a new cold-start user of RiPLE. For this simple example the two parameters are set to $\beta = 1$ and $kgw = 0.8$.

In the current scenario, Alice has correctly answered the first three questions, and she has not found them to be very challenging. Because of her correct answers on topics $T1$ and $T2$, her learning profile vector [-0.18, -0.19, 0] indicates a significant lack of knowledge gap on those topics, and a neutral state on $T3$ since she has not attempted any questions on that topic. Forming $\hat{G}$ by propagating the information from the knowledge gaps over to questions based on



P

	Q1	Q2	Q3	Q4	Q5
A	0.6	0.6	0.8	?	?
B	?	0.4	?	0.4	0.8
C	0.8	?	0.2	0.4	0.2
D	?	?	?	?	?

A

	Q1	Q2	Q3	Q4	Q5
A	1	1	1	?	?
B	?	0	?	0	1
C	1	?	0	0	0
D	?	?	?	?	?

D

	Q1	Q2	Q3	Q4	Q5
A	0.2	0.4	0.4	?	?
B	?	0.4	?	0.6	0.6
C	0.4	?	0.6	0.6	0.8
D	?	?	?	?	?

T

	T1	T2	T3
Q1	1	0	0
Q2	0	1	0
Q3	0.5	0.5	0
Q4	0	0	1
Q5	0.33	0.33	0.33

R

	Q1	Q2	Q3	Q4	Q5
A	0.57	0.2	0.8	?	?
B	?	0.8	?	0.75	0.2
C	0.2	?	0.76	0.8	0.72
D	?	?	?	?	?

G

	Q1	Q2	Q3	Q4	Q5
A	-0.29	-0.31	-0.33	?	?
B	?	0.35	?	0.31	-0.38
C	-0.29	?	0.33	0.31	0.29
D	?	?	?	?	?

LP

	T1	T2	T3
A	-0.184	-0.1916	0
B	-0.096	0.097	0.078
C	-0.01	0.144	0.175
D	-0.097	0.016	0.08

$\hat{R}$

	Q1	Q2	Q3	Q4	Q5
A	0.57	0.2	0.8	0.9	0.7
B	0.59	0.8	0.55	0.75	0.2
C	0.2	0.1	0.76	0.8	0.72
D	0.38	0.5	0.78	0.77	0.46

O

	Q1	Q2	Q3	Q4	Q5
A	0.34	0.02	0.61	0.9	0.58
B	0.5	0.87	0.55	0.82	0.22
C	0.22	0.24	0.81	0.95	0.80
D	0.28	0.51	0.73	0.85	0.46

$\hat{G}$

	Q1	Q2	Q3	Q4	Q5
A	-0.184	-0.191	-0.187	0	-0.125
B	-0.096	0.097	0.001	0.078	0.02
C	-0.01	0.144	0.066	0.175	0.103
D	-0.097	0.016	-0.04	0.084	0.001

New Questions

A	Q4	Q5	-	-	-
B	Q3	Q1	-	-	-
C	Q2				
D	Q4	Q3	Q2	Q5	Q1

Review questions

A	Q3	Q1	Q2	-
B	Q2	Q4	Q5	-
C	Q4	Q3	Q5	Q1
D	-	-	-	-

Focus on T2

A	Q3	Q5	Q2
B	Q2	Q3	Q5
C	Q3	Q5	Q2
D	Q3	Q2	Q5

Figure 2: An example of RiPLE with four students, five questions, and three topics that shows how the system operates.

the associated tags leads to the indication that answering $Q4$ followed by $Q5$, which both focus on $T3$ benefits her the most in terms of reducing her knowledge gaps. Considering the output vector from matrix O for Alice, in the explore mode RiPLE recommends answering $Q4$ over $Q5$ since it would help her the most in overcoming the existing knowledge gap on $T3$. In the review mode, the system recommends answering $Q3$ over the other questions since they all contribute roughly the same in overcoming her knowledge gaps, but she has expressed a higher interest towards $Q3$. In the focus mode assuming $T2$ is selected, the system recommends $Q3$, with a slight edge, over Q5 because of her high interest on that question. Assigning a larger value to β (e.g. 1.5 instead of 1) would have resulted in $Q5$ being recommended over $Q2$.

Bob has answered three questions altogether, two of which are answered incorrectly. Matrix D also shows that he has rated the questions as more challenging than Alice. Because of his incorrect answers on topics $T2$ and $T3$, his learning profile vector [-0.096, 0.097, 0.078] indicates a knowledge gap on those topics. The predicted gap for topic $T2$ is greater since the question answered incorrectly on $T2$ had a lower level average difficulty compared to the question that was answered on $T3$. Bob answered an easy question correctly on $T1$, so the vector shows a slight competency on that topic. Forming $\hat{G}$ leads to the prediction that answering $Q2$, focused on $T2$, followed by $Q4$, focused on $T3$, would benefit Bob the most in terms of reducing his knowledge gaps. Considering the output vector from matrix O for Bob, in the explore mode the system redirects recommendations from $\hat{R}$ to recommend answering $Q3$ over $Q1$, allowing him to receive further practice on $T2$. In both the review mode and the focus mode, assuming $T2$ is selected, the system recommends answering $Q2$ since this would help overcome knowledge gaps on $T2$ and Bob has expressed relatively high interest in that question.

Catherine (C) has answered four questions, but only one of them correctly. Based on her two incorrect answers on $T3$, one incorrect answer on $T2$, and one correct and one incorrect answer on $T1$ her learning profile vector is computed as [-0.01, 0.144, 0.175]. The slight competency on $T1$ arises because the $T1$ weight of the question she answered correctly ($Q1$) was greater than the one she answered incorrectly ($Q5$). Consistent with these findings, $\hat{G}$ indicates that she would benefit the most from answering $Q4$, which solely focuses on $T3$. Considering the output vector from matrix O for Catherine, in the explore mode the system recommends answering $Q2$ since it is the only unanswered question. In the review mode, the system recommends answering $Q4$, helping her overcome knowledge gaps on $T3$. Despite her high interest for $Q1$, it receives a very low overall rating as it does not help her overcome her major knowledge gaps. Assigning a smaller value to kgw (0.1 instead of 0.8) would have resulted in $Q1$ being recommended over $Q4$. In the focus mode, assuming $T2$ is selected, the system recommends $Q3$ with a slight edge over $Q5$ because of the higher rating in $\hat{R}$.

Dean has not answered any questions, so he is a cold-start user. Since mean normalization is used, Dean's latent factors in $\hat{R}$ are filled with average ratings from the training data set (e.g., $\hat{r}_{DQ1}$ is equal to 0.38, which is the average of r_{AQ1} and r_{CQ1}). By using mean normalization, the system approximates Dean's knowledge gaps based on the knowledge gaps of the cohort, $\bar{lp}$; therefore, he is expected to benefit the most from questions on $T3$ and the least from questions on $T1$. Considering the output vector from matrix O for Dean, in the explore mode the system recommends answering $Q4$, which is also recommended to many regular users. In the review mode, the system cannot make any recommendations since he hasn't answered any questions before. In the focus mode, the questions from each topic that the cohort would mostly benefit from would also be recommended to Dean.

6. Validation: Using Synthetic Data Sets

The goal of RiPLE is to provide accurate recommendations that help users to overcome knowledge gaps while keeping their engagement to a maximum by prioritizing questions that are of interest to them. In this section, the behavior of the system is validated and examined under different circumstances using synthetic data sets, in which the underlying knowledge gaps of the students are pre-defined. Each experiment is repeated five times. Reported values are the average results across the five runs. In these experiments, users that have answered less than three questions are considered cold-start users.

The following metrics are used for evaluating the output.

Metric for Question-level recommendations As used commonly in recommender systems evaluations, Root Mean Squared Error (RMSE) is used for measuring the error in the recommendation:

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in ds}(r_{ui} - \hat{r}_{rui})^2}{|ds|}} \tag{12}$$

where ds is the set of all pairs of (u, i) in the data set for which RMSE is being reported.

Metric for topic-level recommendations Accuracy of the model in terms of recommending questions that match students' most significant knowledge gap:

$$Accuracy = \frac{match}{|ds|} \tag{13}$$

where $match$ is the number of instances $\in ds$ where the topic of the recommendation matches student's most significant knowledge gap.

6.1. Template for Generating Synthetic Data Sets

The experiments discussed in this section make use of synthetic data sets generated using the following sequence of steps. First, a set of users with pre-defined knowledge gaps over a set of topics are created. Second, a set of questions with a pre-defined topic, level of difficulty and discrimination is generated. Knowledge gaps must sum to one, and are constructed by sampling from a Dirichlet distribution, where α defines the sparsity of the distribution; a smaller value of α creates a sparser distribution over knowledge gaps, producing synthetic users with a large gap over one topic. The topics associated with a question are sampled from a discrete uniform distribution; their level of difficulty and discrimination are both sampled from a normal distribution. The probability of a user u answering a question i correctly is computed using a 2-parameter logistic Latent Trait Model from classical Item Response Theory (Drasgow and Hulin, 1990), as recommended by (Desmarais and Pelczer, 2010):

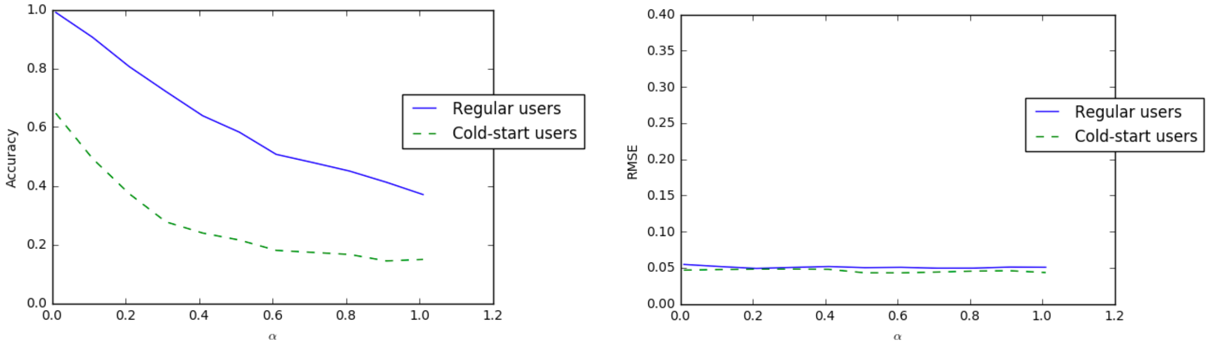
$$\frac{1}{1 + e^{-a_i(\theta_s - b_i)}} \tag{14}$$

where θ_s represents user's average lack of knowledge gaps (competencies) in the topic(s) associated with question i , b_i is the difficulty level and a_i is the discrimination level of question i. The difficulty level user u has expressed towards question i is sampled from a normal distribution based on the difficulty level of i. The interest level that user u has expressed towards question i is sampled from a uniform distribution.

In all generated data sets 400 users, 1100 questions, and 22000 answers are sampled, which roughly matches the numbers from the historical data set that is used for exploration in Section 7. If not otherwise stated, the hyper-parameters are set using the following default values: $\alpha = 0.1$, $L = 10$, $\beta = 0.1$, $kgw = 0.8$, $\gamma = 0.1$, $k = 5$. Results are evaluated using 5-fold cross-validation.

6.2. Impact of Varying Parameters in Synthetic Data Set Generation

Impact of the Sparsity of the Pre-defined Knowledge Gaps (α): Figure 3 illustrates the effect of α, which defines the sparsity of user knowledge gaps among the topics, upon the accuracy and RMSE of RiPLE. For regular users, RiPLE can provide recommendations that target users most significant knowledge gaps when α is small. Increasing α, leads



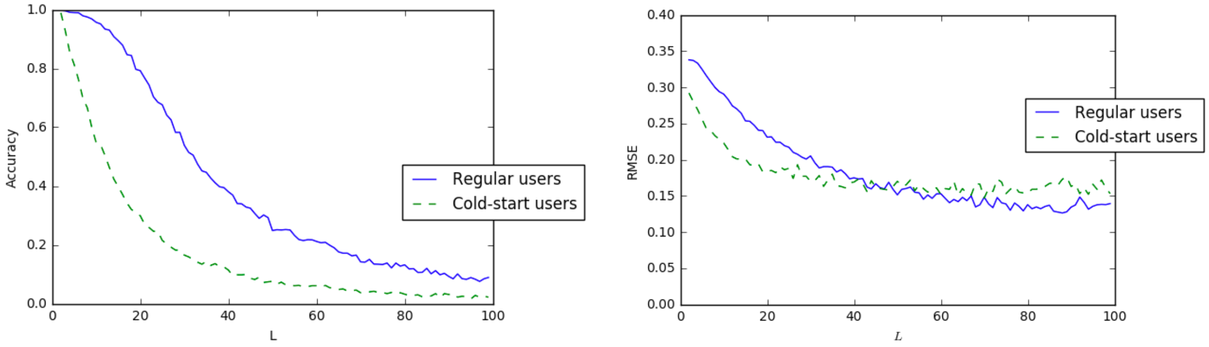
(a) Accuracy as α is increased

(b) RMSE as α is increased

Figure 3: Changes in accuracy and RMSE as the sparsity of the pre-defined knowledge gaps is decreased.

to the simulation of users with less extreme pre-defined knowledge gaps, making it more challenging for the system to accurately identify their most significant gap. For cold-start users, the system's accuracy is lower, as expected. We note that RiPLE is still able to provide reasonable recommendations, considering the limited data available on those users, if α does not grow too large. Since users' knowledge gaps are defined as a vector that sums to one, changes in α do not have a significant impact on the overall probability of a user answering questions correctly, but only moves the knowledge gaps among topics, therefore the RMSE remains quite stable as α is increased.

IMPACT OF NUMBER OF TOPICS (L): Figure 4 illustrates the impact of increasing L, which shows the distinct number of topics that have been used for tagging the questions.



(a) Accuracy as L is increased

(b) RMSE as L is increased

Figure 4: Changes in accuracy and RMSE as the number of topics is increased.

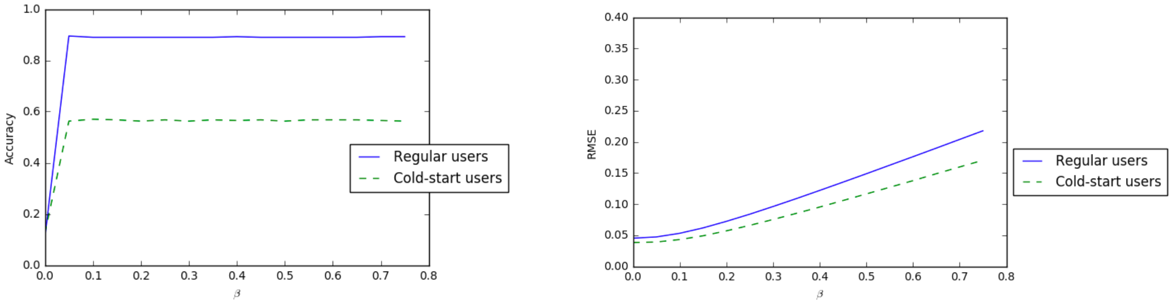
For regular users with $L < 10$, RiPLE can provide recommendations targeting their most significant knowledge gap over 90% of the time. For $10 \leq L \leq 20$, which would be the case for most of the commonly taught courses, the system remains relatively reliable being able to identify the most significant gap over 80% of the time. For more extreme cases where $20 \leq L \leq 100$, the system does computationally scale; however, the task becomes much more challenging, and the accuracy drops significantly. For cold-start users, the system finds it more challenging to determine the most significant knowledge gap of a user (as expected) since they are unlikely to have encountered a question on that topic.

As described in (14), the probability of a user answering a question correctly relies significantly on the pre-defined knowledge gaps of the user. When dealing with users with sparse knowledge gaps, the sampled values determining whether a user correctly answers a question with an unknown topic has the highest standard deviation, unpredictability, in the case of $L = 2$ since there is approximately a 50-50 split between the question being answered correctly and incorrectly; therefore, the highest RMSE is observed when $L = 2$. For $L > 2$, the standard deviation of sampled values is reduced since most of the questions would have a higher probability of being answered correctly.

SUMMARY The results presented in this section have demonstrated that both α and L have a significant impact on the performance of RiPLE. For small values of α and L, which contribute to the creation of a simple environment with strong correlations among values in the data set, RiPLE can provide recommendations that strongly match users' knowledge gaps, validating the theoretical foundations of the system. Larger values of α and L can be used to create more realistic data sets, in which RiPLE is still able to perform relatively well. Extreme values of α and L (that lead to the creation of data sets more complicated than expected in real data sets on Peer-Learning Environments) demonstrate the scalability of the system, showing that it exhibits robust behavior under more extreme circumstances.

6.3. IMPACT OF VARYING RIPLE MODEL PARAMETERS

IMPACT OF THE LEARNING PROFILE (β): Figure 5 demonstrates how the output of the system changes as we increase β, which controls the impact of the learning profile.



(a) Accuracy as β is increased

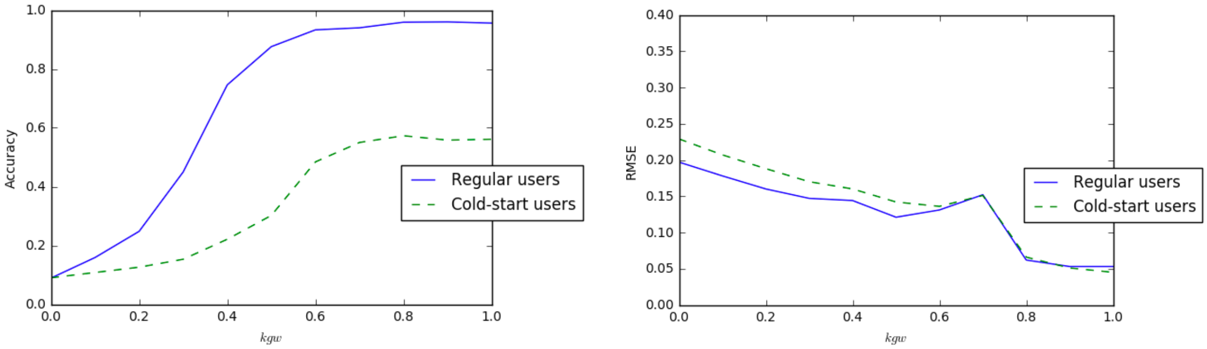
(b) RMSE as β is increased

Figure 5: Changes in accuracy and RMSE as the impact of the learning profile is increased.

In this figure, $\beta = 0$ shows the output of RiPLE where recommendations are made without considering the learning profile (i.e., the system has no knowledge of what topics a question can be classified with). In this setting, the probability of students receiving questions that target their most significant knowledge gap is close to 10%, which with ten topics is the expected result. As β is increased to 0.05, the probability of students receiving questions that target their knowledge gaps is increased without any significant changes in the RMSE. For regular users, the accuracy is 96% and for cold-start users, the accuracy is close to 57%. For both sets of users when $\beta > 0.05$, the RMSE starts to increase without significant changes to the accuracy. This suggests that keeping the other parameter settings constant, $\beta = 0.05$ produces the best results. This general trend also occurs for other values of α and L; however, the accuracy drops as α and L are increased, and the best value for β varies in different experiments.

The results of this experiment demonstrate that the value of β has a significant impact on the performance of the system. The goal is to set β, such that users will be exposed to questions targeting their knowledge gaps without making significant sacrifices on the RMSE, which partially represents users' interests.

IMPACT OF KNOWLEDGE GAPS IN DEFINING USER BENEFITS (kgw): Figure 6 shows how the output of RiPLE changes as we increase kgw, which determines how much the system should emphasize knowledge gaps compared to the interests of users.



(a) Accuracy as kgw is increased

(b) RMSE as kgw is increased

Figure 6: Changes in accuracy and RMSE as the impact of knowledge gaps in defining user benefits is increased.

Increasing kgw improves RiPLE's accuracy in providing users with recommendations that target their most significant gap while reducing the overall RMSE. This is expected since the increase in kgw adds more synergy between users' defined benefits and their knowledge gaps. The sacrifice, which is not captured by either the accuracy or the RMSE, is that increasing kgw tailors recommendations away from users' interests. With small values of kgw, where the emphasis is mostly on the interests, RiPLE is unable to provide recommendations that target user's most significant knowledge gaps. When kgw is set to 0.8, the system has almost 100% chance of providing regular users with recommendations that target their most significant gap. A similar trend, but with a lower overall accuracy, is observed for cold-start users. Though the system provides the flexibility for the user or a meta-user (e.g., instructors) to set the value of kgw, it appears that increasing it beyond a point, which in this experiment is 0.8, may only disregard users' interests without providing any additional benefits in determining gaps.

SUMMARY The results presented in this section demonstrate that both kgw and β can affect the impact of knowledge gaps in the final recommendations made by RiPLE. The main difference is that kgw sets the impact at a question-level in defining how student benefits are determined, which in turn impacts how values in R are stored. In contrast, β establishes the impact at a topic-level using the learning profile, ensuring that the system's recommendations are aligned with the user's interests (based on kgw).

7. Exploration: Using a Historical Data Set

The behavior of RiPLE is explored in this section using a historical data set from a first-year programming course at The University of British Columbia. The data set is captured in the course by using PeerWise. Two main questions are considered here:

Question 1. How does the accuracy of RiPLE respond as one of its core components, the RecSys, is varied across a collection of standard techniques? The conjecture is that the RecSys is an independent component: the accuracy of RiPLE reflects the accuracy of the RecSys technique used. In other words, RiPLE does not have unintended interactions with the RecSys used, which would allow its replacement in the future as new techniques, with improved accuracy, become available.

Question 2. How well do the recommendations made by RiPLE reflect a student's knowledge gaps and interests? The quality of the recommendations is considered using the following refinement: How do the identified knowledge gaps for users relate to their final examination achievements on topics? The conjecture is that the identified knowledge gaps in RiPLE reflect the actual knowledge gaps of the users as indicated in their final examination achievements. In other words the knowledge gaps identified for users by RiPLE are accurate.

How do the recommended questions for users relate to their identified knowledge gaps? The conjecture is that the recommended questions in RiPLE would match the identified knowledge gaps of the users. In other words, how accurate is RiPLE when recommending questions?

Are the recommended questions for users personalized? The conjecture is that if the recommended questions in RiPLE match the identified knowledge gaps of the users and there are a large number of users, then a large number of distinct questions that span the topics would be recommended by RiPLE. This leads to a personalization of the questions recommended by RiPLE are personalized for individual students.

Following a description of the historical data set in subsection 7.1., Question 1 is explored in subsection 7.2., and Question 2 is considered in 7.3.

7.1. Data Set Description

A historical PeerWise data set has been used in this analysis that was created in a required, introductory course in C programming for engineering students offered at The University of British Columbia in 2014. To encourage participation, students received grades for their use of the PeerWise environment: (i) They were required to author at least 3 questions and to correctly answer at least 45 questions (worth 1.5% of final mark) and (ii) a grade was calculated from the "Answer Score" (AS) and "Reputation Score" (RS), which were computed by the PeerWise system, using the following formula: $\frac{min(AS,RS)\times 1.5}{500}$, (worth 1.5% of final mark). In total 377 students authored 1111 questions, assigned 1700 tags that cover 10 topics, and answered 21432 questions.

Recalling the discussion of Section 4.1. it is necessary to scale some of the data stored in the matrices P and D to real values between [0, 1] for this study. The answers from matrix A are binary and do not require scaling. The results reported here are generated using 60% of data for training the model with different hyper-parameter settings, 20% of data used for setting the hyper-parameters, and 20% for assessing the accuracy of the model.

7.2. Exploring the Behavior of RiPLE Using Alternative RecSys Techniques

As Figure 1 summarizes, RiPLE uses a RecSys for predicting the benefits users might receive from answering unseen questions. In this study the behavior of RiPLE is explored as the RecSys is varied across a collection of standard techniques. MyMediaLite (Gantner et al., 2011), an open source RecSys library that provides implementations of a collection of standard RecSys algorithms, is used to compare the accuracy of RiPLE the following standard RecSys techniques:

User-based Average (U-AVG) which computes the average ratings across all users to approximate how u might rate unseen items.

Item-based Average (I-AVG) which computes the average ratings across all items to approximate how i might be rated by users.

User-based KNN (U-KNN) computes similarities between users using the Pearson correlation coefficient to find the K most similar users to a user u. The past ratings from the K nearest neighbors are then used to approximate how u might rate unseen items.

Item-based KNN (I-KNN) computes similarities between items using the Pearson correlation coefficient to find the K most similar items to an item i. The past ratings that the K nearest neighbors have received are then used to approximate how i might be rated by users.

Matrix Factorization (MF) in its simplest form as described in Section 3.1.

Biased Matrix Factorization(BMF) extends conventional matrix factorization with the addition of a bias parameter for each user and item as described by (Koren, 2008).

Many additional, alternative algorithms are available, which could also have been employed by RiPLE. In this work the discussion is limited to solutions available in established, open-source libraries to ensure that developing a prototype system is feasible.

Figure 7 visualizes the RMSE results for the cases where the six techniques are used in RiPLE; error bars are calculated using standard deviation. MF and BMF, which are both based on matrix factorization, outperform the standard user-based or item-based approaches and are within standard error from one another in this data set. Their superior performance can be explained by their ability to implicitly incorporate latent features that may tie to characteristics such as the “slip” and “guess” factors (Thai-Nghe et al., 2011).

In response to Question 1, the results indicate the RecSys behaves much like an independent component: the accuracy of RiPLE consistently reflects the accuracy of the RecSys technique used. The RecSys could reasonably be replaced in future studies, for example, as further improvements become available in the community.

7.3. Exploring the Recommendation Quality by RiPLE

In this study, the behavior of RiPLE is explored with respect to the quality of the recommendations produced. The analysis is performed in a single run on the data set, which would correspond to providing recommendations to users in time to prepare for their final examination near the end of the term. Given that the analysis is performed at the end of the term and users receive

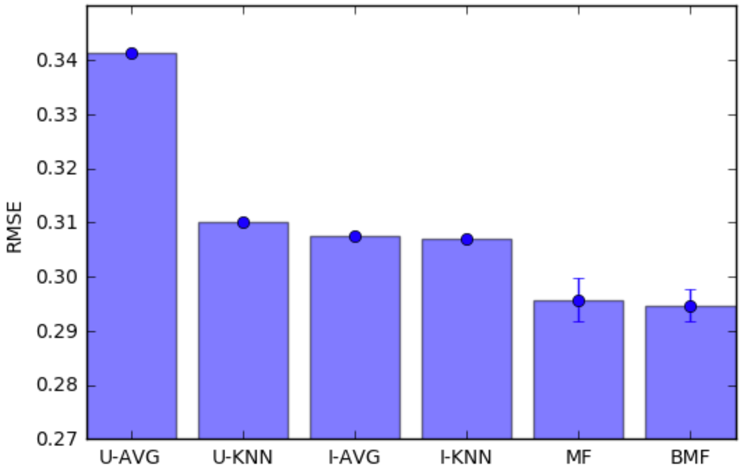


Figure 7: Comparison of the predictive accuracy of RiPLE with standard RecSys techniques implemented in MyMediaLite using RMSE

grades for participation, the data set has only a few cold-start users. Consequently, the study considers the entire cohort without splitting them into regular and cold-start users.

The study uses RiPLE with the most accurate RecSys technique identified (Section 7.2.): BMF. The following hyper-parameter settings are identified: $\gamma = 0.002$, $K = 2$, $itr_r = 300$, $kgw = 0.8$, and $\beta = 0.51$. In all cases other than kgw, the hyper-parameter values are derived using the validation set; under this setting, the RMSE is computed as 0.2947.

A summary of the data characteristics, recommendations, and the final examination score achieved is illustrated in Table 2 (topic name, number of questions, class-level gap, total number of users that received recommendations on each topic, and the class-level exam grade). The topics are listed in the order in which they are covered in class, which may explain why some of the earlier topics receive more questions than others.

In response to Question 2, the results indicate the quality of the recommendations by RiPLE is promising. With respect to the quality of the identified knowledge gaps by RiPLE, the topic of the three most significant class-level gaps (average gap on topics over the cohort) are programming comprehension, File IO, and functions. These match the topics that receive the lowest average grade on the final exam, indicating the knowledge gaps identified by RiPLE reflect the actual knowledge gaps of the students.

With respect to the relationship of the recommended questions for users to their identified knowledge gaps, **89%** of the users receive recommendations that match their primary (most significant) gap identified from their learning profile vector. In other cases, the users receive recommendations on alternative questions that match knowledge gaps in their learning profile vector. This can occur, for example, when the interests of a user are determined to be a superior match for questions that address a secondary or tertiary ranked knowledge gap.

With respect to the quality of personalized recommendations by RiPLE, the questions recommended at a topic-level broadly span all available topics. In addition, a relatively large number, 99, of distinct questions are presented as the first recommendation to individual students. These results suggest that RiPLE differentiates users and provides personalized recommendations.

Topic	**# Questions**	**Class-level Gap ($\bar{lp}$)**	**# Topic-level Recommended**	**Class-level Exam Grade**
Introduction	175	-0.202	23	80%
Fundamentals	554	0.012	49	91%
Conditionals	81	-0.023	30	93%
Loops	246	-0.033	34	68%
File I/O	34	0.160	31	66%
Functions	218	0.084	41	55%
Arrays	276	-0.020	38	65%
DAQ Systems	75	-0.090	6	80%
Comprehension	15	0.210	30	30%
Syntax	41	0.070	9	77%

Table 2: Information on the name, number of questions, class-level gap, total number of users that received a recommendation on each topic, and the class-level exam grade.

8. Conclusions and Future Work

Explicitly addressing knowledge gaps in RecSysTEL is a challenging, open research topic. A novel RecSysTEL, RiPLE, was introduced as a way of providing accurate, personalized recommendations to students who are using Peer-Learning Environments. The system consists of five modules defining the Input Data, Data Integration, Learning Profile, Recommendation Engine, and Modes of Operation. The Recommendation Engine uses an established collaborative filtering algorithm (matrix factorization), which is enhanced with a learning profile. This enables the recommendation of specific multiple-choice questions to users that reflect both (i) user interests (the kinds of questions they rank highly) and (ii) their knowledge gaps (the questions they need work on to accomplish learning objectives related to a course). Multiple operational modes allow students to explore new questions, review previously answered questions, or focus on questions from specific topics according to their current learning requirements.

Experimental validation of RiPLE used both synthetic data sets and a historical data set. The synthetic data sets were used to assess the behavior of RiPLE under diverse circumstances, by varying parameters in both the data generation template and the RiPLE model. This demonstrated that the behavior of RiPLE is consistent with expectations over a range of parameter settings, and therefore that there is reason to believe the solution will be robust in a real-world setting. The historical data set was used to explore the accuracy of alternative RecSys techniques, and to demonstrate that RiPLE is likely to both provide recommendations that are both useful and personalized. As RiPLE was able to extract class level knowledge gaps ($\hat{lp}$) which correlated with the final class level exam grade, we have reason to believe that RiPLE is performing appropriately in its profiling of students. Indeed, the result that 89% of students would have received a recommendation that matches with their most significant knowledge gap suggests that the current parameter settings for RiPLE are appropriate and that it has the potential to improve learning outcomes for students. Furthermore, the wide range in recommended topics (with 99 distinct questions provided as a recommendation in this test) suggests that RiPLE is not fixating upon a class wide weakness, but providing well personalized advice as to what questions might help a student best. These results are promising, suggesting that students may significantly benefit from exposure to a tool such as RiPLE.

There are several limitations in the current work which restrict the generalizability of the results. The most significant limitation is that the validation is not a controlled experiment that provides compelling evidence of RiPLE's capacity to make recommendations that lead to better learning. To address this limitation an A/B experiment is planned, in which the control group would receive random recommendations and the experimental group would receive recommendations from RiPLE, to determine whether RiPLE's recommendations lead to measurable learning gains. The design of the controlled experiment is envisioned to include the use of a PeerWise environment extended with RiPLE. Discussions are underway with the founder of PeerWise for integrating RiPLE into their platform. Furthermore, the historical data set is from one class of students (first-year undergraduate engineering course, C programming, Computer Science Department, and so on), which means that current parameter settings are unlikely to generalize across all educational settings. Further investigation is needed to explore the behavior and application of RiPLE in alternative educational domains (e.g., Medicine, Humanities) and settings (e.g., MOOCS). These may, for example, drive the need for very high levels of scalability regarding the number of users, questions, and course topics.

There are several interesting directions to pursue in future work. The formulation of the updated student-question knowledge gap matrix $\hat{G}_{N\times M}$ could be explored as a similarity function potentially describing a network among the questions. It would be interesting to compare the results of the different formulations (i.e., learning profile approach vs. a network approach). It would also be important to refine RiPLE to consider the learning effect using alternative factorization techniques, as students improve their understanding of topics over time. The PeerWise data set provides timestamp information, which may be included in the recommendation model to create more sophisticated models of individual users which evolve in time. Also, the interpretability of the recommendations made by RiPLE is a worthy topic of further investigation. Matrix factorization can result in models that are easier to understand, which suggests that a study could be designed to explore how students make sense of the recommendations, and how this impacts upon their metacognition, and ability to reflect upon and improve their participation in Peer-Learning environments.

In conclusion, the results are promising and demonstrate that it is possible to combine MF based learning profiles with a RecSys designed to help students identify knowledge gaps and then work to remove them. The presented system, while designed for the PeerWise environment, is general enough that it could be applied to other environments which store similar information. This means that the system presented here can be used to explore many issues related to student profiling and personalization in a wide variety of question answering scenarios.

References

Baker, R., Corbett, A. T., and Aleven, V. 2008. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *International Conference on Intelligent Tutoring Systems*. Springer, 406–415.

Barnes, T. 2005. The q-matrix method: Mining student response data for knowledge. In *American Association for Artificial Intelligence 2005 Educational Data Mining Workshop*. 1–8.

Barnes, T. 2010. Novel derivation and application of skill matrices: The q-matrix method. *Handbook on educational data mining*, 159–172.

BATES, S. P., GALLOWAY, R. K., AND MCBRIDE, K. L. 2012. Student-generated content: Using peerwise to enhance engagement and outcomes in introductory physics courses. In *AIP Conference Proceedings*. Vol. 1413. AIP, 123–126.

BECK, J. AND CHANG, K.-M. 2007. Identifiability: A fundamental problem of student modeling. *User Modeling 2007*, 137–146.

BESWICK, G., ROTHBLUM, E. D., AND MANN, L. 1988. Psychological antecedents of student procrastination. *Australian psychologist 23,* 2, 207–217.

BETTS, B. 2013. Towards a method of improving participation in online collaborative learning: Curatr. *Teaching and Learning Online: New Models of Learning for a Connected World 2*, 260–273.

BIGGS, J. 1999. What the student does: teaching for enhanced learning. *Higher education research & development 18,* 1, 57–75.

BOUD, D., COHEN, R., AND SAMPSON, J. 2014. *Peer learning in higher education: Learning from and with each other*. Routledge.

CAZELLA, S., REATEGUI, E., AND BEHAR, P. 2010. Recommendation of learning objects applying collaborative filtering and competencies. In *Key Competencies in the Knowledge Society*. Springer, 35–43.

CECHINEL, C., SICILIA, M.-Á., SÁNCHEZ-ALONSO, S., AND GARCÍA-BARRIOCANAL, E. 2013. Evaluating collaborative filtering recommendations inside large learning object repositories. *Information Processing & Management 49,* 1, 34–50.

CHEN, C.-M. 2008. Intelligent web-based learning system with personalized learning path guidance. *Computers & Education 51,* 2, 787–814.

CHIN, C. AND BROWN, D. E. 2002. Student-generated questions: A meaningful aspect of learning in science. *International Journal of Science Education 24,* 5, 521–549.

COETZEE, D., LIM, S., FOX, A., HARTMANN, B., AND HEARST, M. A. 2015. Structuring interactions for large-scale synchronous peer learning. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. ACM, 1139–1152.

DENNY, P., HAMER, J., LUXTON-REILLY, A., AND PURCHASE, H. 2008. Peerwise: students sharing their multiple choice questions. In *Proceedings of the fourth international workshop on computing education research*. ACM, 51–58.

DESMARAIS, M. C. 2012. Mapping question items to skills with non-negative matrix factorization. *ACM SIGKDD Explorations Newsletter 13,* 2, 30–36.

DESMARAIS, M. C., BEHESHTI, B., AND NACEUR, R. 2012. *Item to Skills Mapping: Deriving a Conjunctive Q-matrix from Data*. Springer Berlin Heidelberg, Berlin, Heidelberg, 454–463.

DESMARAIS, M. C. AND NACEUR, R. 2013. *A Matrix Factorization Method for Mapping Items to Skills and for Enhancing Expert-Based Q-Matrices*. Springer Berlin Heidelberg, Berlin, Heidelberg, 441–450.

DESMARAIS, M. C. AND PELCZER, I. 2010. On the faithfulness of simulated student performance data. In *Educational Data Mining (EDM 2010), Pittsburgh, PA, USA, June 11-13*. 21–30.

DRACHSLER, H., VERBERT, K., SANTOS, O. C., AND MANOUSELIS, N. 2015. Panorama of recommender systems to support learning. In *Recommender systems handbook*. Springer, 421–451.

DRASGOW, F. AND HULIN, C. L. 1990. Item response theory.

ERDT, M., FERNÁNDEZ, A., AND RENSING, C. 2015. Evaluating recommender systems for technology enhanced learning: A quantitative survey. *IEEE Transactions on Learning Technologies 8,* 4, 326–344.

FAZELI, S., LONI, B., DRACHSLER, H., AND SLOEP, P. 2014. Which recommender system can best fit social learning platforms? In *European Conference on Technology Enhanced Learning*. Springer, 84–97.

GANTNER, Z., RENDLE, S., FREUDENTHALER, C., AND SCHMIDT-THIEME, L. 2011. Mymedialite: a free recommender system library. In *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 305–308.

GOMEZ-ALBARRAN, M. AND JIMENEZ-DIAZ, G. 2009. Recommendation and students authoring in repositories of learning objects: A case-based reasoning approach. *International Journal of Emerging Technologies in Learning (iJET) 4,* 2009, 35–40.

HARDY, J., BATES, S. P., CASEY, M. M., GALLOWAY, K. W., GALLOWAY, R. K., KAY, A. E., KIRSOP, P., AND MCQUEEN, H. A. 2014. Student-generated content: Enhancing learning through sharing multiple-choice questions. *International Journal of Science Education 36,* 13, 2180–2194.

IMRAN, H., BELGHIS-ZADEH, M., CHANG, T.-W., GRAF, S., ET AL. 2016. Plors: a personalized learning object recommender system. *Vietnam Journal of Computer Science 3,* 1, 3–13.

KOPEINIK, S., LEX, E., SEITLINGER, P., ALBERT, D., AND LEY, T. 2017. Supporting collaborative learning with tag recommendations: a real-world study in an inquiry-based classroom project. In *Proceedings of the Seventh International Learning Analytics & Knowledge Conference*. ACM, 409–418.

KOREN, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.

LEMIRE, D., BOLEY, H., MCGRATH, S., AND BALL, M. 2005. Collaborative filtering and inference rules for context-aware learning object recommendation. *Interactive Technology and Smart Education 2,* 3, 179–188.

LUMEZANU, C., LEVIN, D., AND SPRING, N. 2007. Peerwise discovery and negotiation of faster paths. In *HotNets*.

MANGINA, E. AND KILBRIDE, J. 2008. Evaluation of keyphrase extraction algorithm and tiling process for a document/resource recommender within e-learning environments. *Computers & Education 50,* 3, 807–820.

MANOUSELIS, N., DRACHSLER, H., VUORIKARI, R., HUMMEL, H., AND KOPER, R. 2011. Recommender systems in technology enhanced learning. In *Recommender systems handbook*. Springer, 387–415.

PARDOS, Z. A. AND HEFFERNAN, N. T. 2010. Using HMMs and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. *Journal of Machine Learning Research W & CP*.

PURCHASE, H., HAMER, J., DENNY, P., AND LUXTON-REILLY, A. 2010. The quality of a peerwise mcq repository. In *Proceedings of the Twelfth Australasian Conference on Computing Education-Volume 103*. Australian Computer Society, Inc., 137–146.

RICCI, F., ROKACH, L., AND SHAPIRA, B. 2011. *Introduction to Recommender Systems Handbook*. Springer US, Boston, MA, 1–35.

ROSENSHINE, B., MEISTER, C., AND CHAPMAN, S. 1996. Teaching students to generate questions: A review of the intervention studies. *Review of educational research 66,* 2, 181–221.

SALEHI, M. 2013. Application of implicit and explicit attribute based collaborative filtering and BIDE for learning resource recommendation. *Data & Knowledge Engineering 87*, 130–145.

THAI-NGHE, N., DRUMOND, L., HORVÁTH, T., KROHN-GRIMBERGHE, A., NANOPOULOS, A., AND SCHMIDT-THIEME, L. 2011. Factorization techniques for predicting student performance. *Educational Recommender Systems and Technologies: Practices and Challenges*, 129–153.

THAI-NGHE, N., HORVÁTH, T., AND SCHMIDT-THIEME, L. 2010. Factorization models for forecasting student performance. In *Educational Data Mining 2011*.

VERBERT, K., DRACHSLER, H., MANOUSELIS, N., WOLPERS, M., VUORIKARI, R., AND DUVAL, E. 2011. Dataset-driven research for improving recommender systems for learning. In *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*. ACM, 44–53.

WINTERS, T. E. 2006. Educational data mining: collection and analysis of score matrices for outcomes-based assessment. Ph.D. thesis, University of California, Riverside.

WRIGHT, J. R., THORNTON, C., AND LEYTON-BROWN, K. 2015. Mechanical ta: Partially automated high-stakes peer grading. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*. ACM, 96–101.