Reciprocal Content Recommendation for Peer Learning Study Sessions

Boyd A. Potts, Hassan Khosravi, and Carl Reidsema

The University of Queensland, Brisbane, Australia

Abstract. Recognition of peer learning as a valuable supplement to formal education has lead to a rich literature formalising peer learning as an institutional resource. Facilitating peer learning support sessions alone however, without providing guidance or context, risks being ineffective in terms of any targeted, measurable effects on learning. Building on an existing open-source, student-facing platform called RiPPLE, which recommends peer study sessions based on the availability, competencies and compatibility of learners, this paper aims to supplement these study sessions by providing content from a repository of multiple-choice questions to facilitate topical discussion and aid productiveness. We exploit a knowledge tracing algorithm alongside a simple Gaussian scoring model to select questions that promote relevant learning and that reciprocally meet the expectations of both learners. Primary results using synthetic data indicate that the model works well at scale in terms of the number of sessions and number of items recommended, and capably recommends from a large repository the content that best approximates a proposed difficulty gradient.

Keywords: Reciprocal Recommender, Peer Learning, Peer Support, RecSysTEL

1 Introduction

The use of peers in teaching, mentoring or other learning support roles has long been recognised as a valuable addition to the learning environment and has developed from early beginnings in understaffed village schools [8] to the development of formalised, systematic guidelines [24]. The proposed academic and social benefit to students' engagement in formal or informal peer support forms the basis of a rich body of literature in education and psychology, grounded in a fundamental acknowledgement of the importance of peers in successful learning and positive university experience. Learners who engage in peer learning and peer support not only contribute to the existing community, but can create entirely new communities of practice [3]. These communities can be supplemented with fit-for-purpose technology to increase the frequency of interaction and communication between students and create an awareness of a community for them to "belong" to and draw upon as a resource over the course of their studies [6].

However, peer interactions are often topically disconnected from academic objectives and even with the intent to provide academic support [17], simply

encouraging connectedness in learning environments risks being ineffective in terms of any targeted, measurable effects on learning. As part a comprehensive evaluation of students' experiences with Peer Assisted Study Sessions (PASS), [16] surveyed participating students over their satisfaction and the program's effectiveness. In an open-ended question of students' suggestions for improving the PASS program, 36% of usable responses related to the focus of the sessions. More specifically, approximately half of this group wanted questions and examples reflecting those in exams and assignments. The other half were concerned specifically with a standard of preparation in PASS sessions, such as an organised outline for each session, commenting that some facilitators were poorly organised. Among the quoted examples of feedback in [16] was "More structure, follow some kind of plan would be helpful" $p11$. Other feedback included a desire for notes/handouts for the session and even some feedback that tutors were not able to answer specific questions about the content. These results demonstrate that it is not enough to simply arrange study sessions with peers without any guidance or support, which is the problem being addressed in this paper.

RiPPLE (Recommendation in Personalised Peer Learning Environments) [12] is an open-source, adaptive, crowdsourced, web-based, student-facing learning platform that provides personalised content and peer learning support tailored to the needs of each individual at scale. RiPPLE has two main features: (1) recommendation of formative exercises based on the knowledge state of each learner, as described in [13]; and recommendation of peer learning study sessions based on the availability, learning preferences and needs of each learner, as described in [23]. In this paper, we extend the functionality of RiPPLE to use the knowledge state of users to recommend content for the scheduled study sessions such that it reciprocally meets the expectations of both learners.

In preparation for trialling RiPPLE in four large face-to-face courses, an initial set of experiments were conducted. The evaluation concentrates on the feasibility of the recommender by evaluating the impacts of the size of the question repository and model parameter settings. Synthetic data sets were created for this purpose. Primary results indicate that the system can recommend content for study sessions that reciprocally meets the expectations of both learners.

2 Background

The area of reciprocal content recommendation is closely related to two research areas: (1) knowledge tracing, which is briefly covered in Section 2.1; and two subfields tied to recommender systems, namely, recommender systems in technology enhanced learning (RecSysTEL) and reciprocal recommender systems, which are briefly covered in Section 2.2.

2.1 Knowledge Tracing

Heavily studied within the intelligent tutoring community [4], knowledge tracing is the task of modeling the knowledge state of students so that their future

performance on learning activities can accurately be predicted [5]. The Bayesian Knowledge tracing (BKT) algorithm [5] is one of the most prominent methods used for knowledge tracing, using Hidden Markov Models to capture the student knowledge states as a set of binary variables representing whether or not a concept has been mastered.

BKT has received significant attention and improvement since it was first proposed. [1] introduced parameters for slipping (where a student has the required skill but mistakenly provides the wrong answer) and guessing (where a student provides the right answer despite not having the required skill for solving the problem). Later, [18] effectively extended BKT to capture item difficulty which led to improved prediction accuracy. More recently, [29] introducing a new set of parameters capturing prior knowledge of individual learners further improved BKT.

Other algorithms with comparable or superior predictive power to BKT have also been proposed for predicting student performance, such as the Learning Factors Analysis Framework [2] and the Performance Factor Analysis framework [19]. [11] incorporated Item Response theory (IRT) into knowledge tracing, and more recently, [20] and [25] used recurrent neural networks for deep knowledge tracing.

Knowledge tracing algorithms are often used as a part of a intelligent tutoring systems [4] or in adaptive learning environments [5] to provide content for an individual learner. In the work presented in this paper, we use the algorithms developed by [13] to provide reciprocal content meeting the expectations of two learners attending the study session.

2.2 RecSysTEL and Reciprocal Recommender Systems

Recommender systems for technology enhanced learning is an active and rapidly evolving research field. In the educational setting, much of the primary research on recommender systems is directed at the recommendation of relevant content and resources for learning, such as context-aware recommendations on learning objects[14], resources within e-learning environments to expand or reinforce knowledge [15], and appropriate exercises based on students' performance predictions [26]. More broadly, researchers with a focus on intelligent tutoring systems and adaptive learning environments have been using knowledge tracing algorithms for recommendation of content to learners.

Reciprocal recommender systems have widely applied to online dating [21], job matching [10], and social networking [9]. Indeed they comprise the popular approach to any problem involving a match between two or more entities, tailored by the best consideration of the specific needs and constraints of the immediate environment. Fundamentally they seek solutions such as a list of recommendations that fulfill users' preferences and expectations to a higher degree than competing items. In reciprocal recommendation, the preferences of two or more users must be satisfied concurrently. That is, a recommendation is made on the extent to which the preferences of one user are compatible with those of

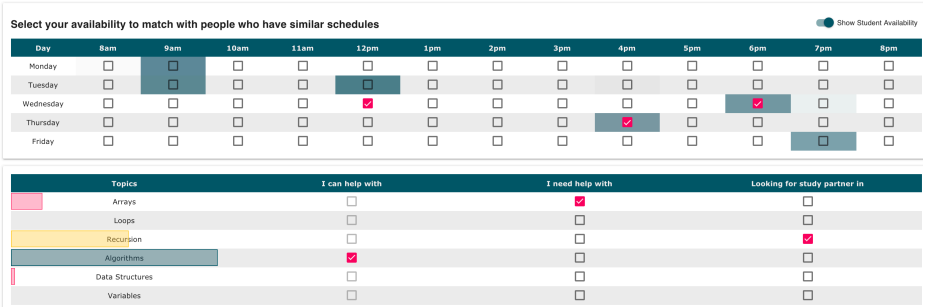


Fig. 1: User preferences in RiPPLE.

another, inducing a higher level of complexity than other recommender systems [22].

Previous work by [23] developed a reciprocal recommender in RiPPLE that allows students to find peers and establish study sessions for the purposes of providing or receiving topical peer support and find study partners. Peers were matched on the basis of coursework competencies and compatibility of preferences, such as weekly availability, and received a short list of recommendations with invitations to connect with recommended peers through the platform (Figure 1). The current paper paper aims to supplement these study sessions by providing content in the form of multiple choice questions (MCQ) from the RiPPLE repository to facilitate communication between peers and productivity in the study sessions.

3 Formal Notation and Problem Formulation

Let $U_N = \{u_1 \dots u_N\}$ denote the set of users that are enrolled in a course in RiPPLE, where u_n and $u_{n'}$ refer to arbitrary users. Let $Q_M = \{q_1 \dots q_M\}$ denote the repository of the questions that are available to users in a course in RiPPLE, where q_m and $q_{m'}$ refer to arbitrary questions. A two-dimensional array $A_{N \times M}$ keeps track of question answers, where $a_{nm} = 1$ indicates that user u_n has answered question q_m correctly, and $a_{nm} = 0$ indicates that user u_n has answered question q_m incorrectly.

Each course consists of a set of knowledge units $\Delta_L = \{\delta_1 \dots \delta_L\}$ referred to as a knowledge space, where δ_l refers to an arbitrary knowledge unit. Questions can be tagged with one or more knowledge units; $\Omega_{M \times L}$ is a two-dimensional array, where ω_{ml} is 1 if question q_m is tagged with δ_l and 0 otherwise. A two-dimensional array $\Lambda_{N \times L}$ is used for representing students' knowledge states approximated by the system, where λ_{nl} represents the knowledge state of u_n on δ_l. This information is used to produce a two-dimensional array $\Phi_{N \times M}$, where ϕ_{nm} shows the probability of user u_n answering question q_m correctly. Values in Φ are computed using algorithms introduced by Khosravi et al in [13].

Let's further assume that students can hold different roles for participation in peer learning sessions, where e_1 is used to present providing peer learning support, e_2 for seeking peer learning support. Using algorithms introduced by Potts et al. in [23], a list of study sessions is generated. Each study session is in the form $[u_n, u_{n'}, \delta]$ indicating that user u_n has received recommendation to connect with user $u_{n'}$ on topics δ.

The aim of this paper is to use information from $\Phi_{N \times M}$ to extend the notion of a study session to be of the form $[u_n, u_{n'}, \delta, Q'_k]$ indicating that a list of k questions $Q'_k \in Q_M$ are to be covered during the session.

4 Reciprocal Content Recommendation

This section focuses on the problem of how best to algorithmically select a suitable set of multiple choice questions for a study session between two peers, that match their role preferences (providing support or receiving support) and provide learning opportunities in terms of subjective difficulty, from a large repository of items. We proceeded within the framework of the following propositions:

1. A user receiving support, u_n, should work through a set of items of increasing difficulty so as to initially orient the session and increasingly provide opportunities for new learning; and
2. The user providing support, $u_{n'}$, should be able to answer these same questions with a high probability of being correct. This should be true of all items such that $u_{n'}$ can adequately assist in learning that u_n would be less likely to accomplish without assistance.

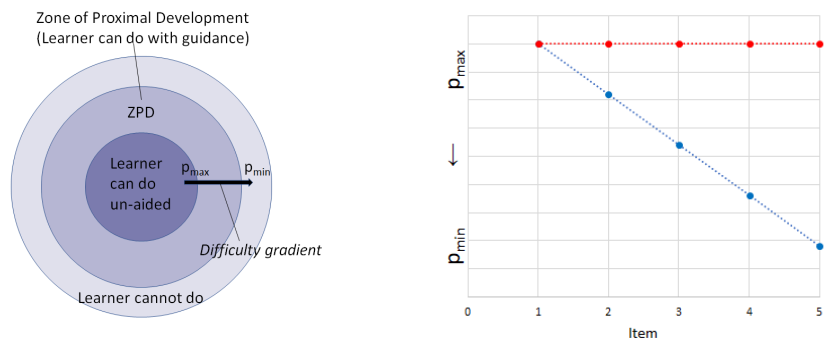
With respect to proposition 1, a difficulty gradient d was computed based on the nominated maximum probability that a user answers an item correctly (p_{max}), the nominated minimum probability that a user answers an item correctly (p_{min}) and the number of items to be recommended for a session (k), as input parameters for the characteristic linear slope function

$$d(p_{min}, p_{max}, k) = \frac{p_{min} - p_{max}}{k - 1} \tag{1}$$

where d represents the increase in subjective item difficulty (with respect to ϕ_{nm}; the probability of user u_n answering question q_m correctly) for each sequential item in the recommended set of k items.

Figure 2 illustrates the concept of the difficulty gradient in terms of the zone of proximal development (ZPD) [28], a popular concept in learning and education that refers to the range of tasks that a learner is capable of completing if provided with guidance. The ZPD exists between those tasks that can be done autonomously and those that are beyond the person's current capability. Proposition 1 aims to traverse outward with respect to the ZPD of the user receiving support as shown in Figure 2a

It is important to note that the parameter values of p_{min} and p_{max} can be system-set, applying to all users in seeking-support role. Recall that values in Φ



(a) Zone of proximal development

(b) Difficulty gradients

Fig. 2: Difficulty gradients and relationship to the zone of proximal development

are inherently personalised to each user, reflecting the probability that a given user answers the item correctly as computed by the algorithm from [13]. As such, personalised item recommendations will result from system-set values of p_{min} and p_{max} for all users without the need for users to specify these variably as explicit preferences.

The slope function in equation 1 was used to create a one-dimensional array, $D_k = \{d_1, d_2, ..., d_k\}$, where elements d_i represent the difficulty of item $i \in \{1, 2, ..., k\}$ across the set of k recommendations. Setting $p_{min} < p_{max}$ ensures items increase in subjective difficulty.

Under proposition 2, users in a providing support role should have $p_{min} = p_{max}$ and should be a high value. That is, for an item to be a suitable inclusion for a study session in which a user is providing support, that user should be able to answer the item correctly with a high probability. This should be true of all k items. This feature aims to address the feedback in [27] that peer facilitators were unable to answer some questions relating to the session content. Study material for each session should take into account the known or inferred competencies of each participant as a basis for selecting the study items to be included in the session plan.

Then for a given difficulty gradient, a scoring function for multiple choice items in the repository that preferences those that most closely align with the gradient is required. That is, a function with maxima at each point d_k in the difficulty gradient with the fitness of any given item assessed by the function height.

Tagged items in the repository were assessed for fitness for each point in D, using a Gaussian function g,

$$g(\phi_{nm}, d_i, \sigma, \delta_l) = \begin{cases} 0 & \delta_l = 0 \\ e^{\frac{-(\phi_{nm} - d_i)^2}{2\sigma^2}} & \delta_l = 1 \end{cases} \tag{2}$$

such that items with subjective difficulties (ϕ_{nm}; the probability of the user answering the item correctly) that more closely approximated the desired values in D received higher scores than those that did not. Items that are not tagged as belonging to the relevant topic ($\delta_l = 0$) receive a score of zero.

The probability of an item being allocated to a less-than-optimal position (i.e. relatively too hard or easy compared to d_i) in D is controlled by the value of σ. Using known properties of the standard normal distribution (z-scores) we calculated the value of σ using the function $\sigma(d, z) = \frac{(d-\frac{d}{2})}{z}$. For example, to ensure that the probability of an item being sub-optimally allocated to a position along D was less than .001, a value of z=3.1 would be required.[1] For users providing support, setting $p_{min} = p_{max}$ results in a difficulty gradient of zero. Consequently $\sigma = 0$ incurs an undefined result of g. The function g in equation 2 therefore uses the same σ derived for the user receiving support. The effects of σ on recommendations are detailed in the evaluation section.

Figure 2b illustrates the values in the array D for two users, where the user providing support has $p_{min} = p_{max} = 1$, and the user receiving support has p_{max}=1 and p_{min}=.1, resulting in d=-.225, giving $D = \{1, .775, .55, .325, .1\}$ and the set of Gaussian functions, centered on d_i, against which potential items were scored for their suitability for recommendation as the ith item in the set.

The first panel in Figure 3 shows the function g for five items (k=5) increasing in difficulty from p_{max}=1 to p_{min}=.1 Topically-tagged items in the MCQ repository are evaluated for their suitability by the height of the relevant function centered on the desired difficulty d_i. For the user providing support, $p_{min} = p_{max} = 1$ resulting in all items being evaluated under the same g function with center $p_{min} = p_{max}$.

A three-dimensional array $G_{U \times Q \times k}$ is created in which g_{nmi} shows the height of the Gaussian function centered on d_i and evaluated at ϕ_{nm} for user u_n and question m, given by equation 2. The degree of fit for each item can be computed by the harmonic mean of the corresponding scores in G for u_n and $u_{n'}$ for each item. Those items with the highest fit were selected for recommendation.
This is shown in the second panel of Figure 3. Note that the harmonic mean gives preference to the score of the user receiving support. This was a design decision that recommendations should be made with the user receiving learning support as the priority.

A simple example is shown in Table 1 for k=3, p_{min}=.1, p_{max}=1, z=3.1 and N=6. The function has selected the set of items [1, 5, 2] as suitable for the study session (shown in bold). The user providing support has $p_{min} = p_{max} = 1$ for all items, and the user receiving support has items increasing in difficulty along gradient D = [1, .55, .1]. It is important to note again that the g scores reflect the extent to which the item's subjective difficulty (ϕ) for the individual user matches the values along the gradient. With respect to making recommendations, g scores (and their harmonic means) have ordinal, rather than cardinal, value. The actual ability of a user is reflected only in ϕ.

[1] Evaluated as the probability that a random score under a standard normal distribution is greater than the midpoint between two sequential items, ($\frac{d}{2}$).

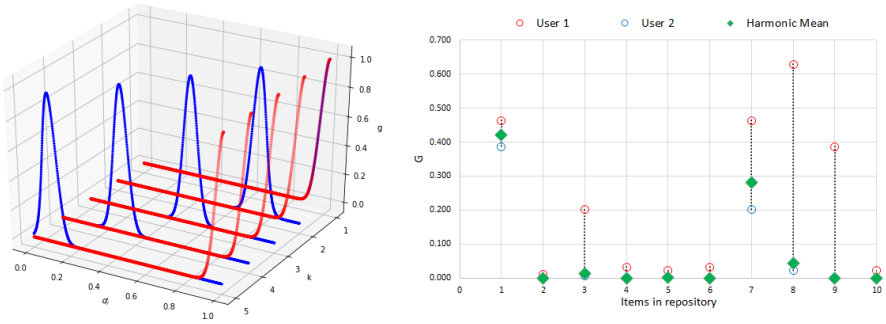


Fig. 3: Panel 1: Gaussian scoring functions along difficulty gradients with k=5, p_{min}=.1, p_{max}=1, z=3.1 for the user receiving support (blue), and $p_{min} = p_{max} = 1$ for the user providing support (red). Items with difficulties more closely approximating the desired gradient receive higher scores than those that do not. Panel 2: Example scoring of ten candidate items for $k = 1$. Item 1 is recommended, having the highest harmonic mean of both users' function scores (cf. Table 1).

Table 1: Item scoring example

	u_n Providing Support				$u_{n'}$ Receiving Support				Harmonic Mean		
		g_{nm}				$g_{n'm}$			*harmonic mean*		
Item	ϕ	$d_1 = 1$	$d_2 = 1$	$d_3 = 1$	ϕ	$d_1 = 1$	$d_2 = .55$	$d_3 = .1$	fit_1	fit_2	fit_3
1	0.91	0.464	0.464	0.464	0.90	0.387	$< 10^{-3}$	$< 10^{-3}$	**0.422**	$< 10^{-3}$	$< 10^{-3}$
2	0.78	0.010	0.010	0.010	0.35	$< 10^{-3}$	0.022	0.003	$< 10^{-3}$	0.014	**0.004**
3	0.87	0.201	0.201	0.201	0.77	0.007	0.010	$< 10^{-3}$	0.013	0.019	$< 10^{-3}$
4	0.93	0.628	0.628	0.628	0.80	0.022	0.003	$< 10^{-3}$	0.043	0.005	$< 10^{-3}$
5	0.9	0.387	0.387	0.387	0.62	$< 10^{-3}$	0.628	$< 10^{-3}$	$< 10^{-3}$	**0.479**	$< 10^{-3}$
6	0.8	0.022	0.022	0.022	0.43	$< 10^{-3}$	0.255	$< 10^{-3}$	$< 10^{-3}$	0.041	$< 10^{-3}$

5 Algorithm

Pseudocode for the recommender is shown in Algorithm 1. It takes inputs of the session (S; an agreed study partnership for two users) and for the function parameters described in equations 1 and 2 generates a list (Q') of k items from the repository (Q) which are appended to the overall session recommendation (S) in RiPPLE. For brevity, we have referred to the users as mentor and mentee in the pseudocode to represent providing and receiving support, respectively, but acknowledge these titles have different implications in the wider literature.

The function to find the highest reciprocal (harmonic mean) scores in line 16 searches the array for the maximum value and assigns the corresponding item to its corresponding gradient position in the set. Both the selected item (array row)

and the position along the gradient (array column) are subsequently assigned as empty sets, and the array is searched for the next maximum, and so on in a loop until all positions in the set have been assigned a recommendation.

For example, in Table 1 the first item to be recommended in the loop was item 5, having the highest score in the array, and it was allocated to its position (second) in the set. Both row 5 and column 2 were then emptied, and the new array consisting of only the remaining items and positions was searched, with item 1 being recommended to its position (first) in the set. This ensures that the recommendations are always the best available, without repeating items.

Algorithm 1: Recommending items

```
Input : S, Φ_{N×M}, k, p_min, p_max, σ
G_{U×M×k} = {};
dMentor = findOptimalDifLevels(p_max, p_max, k);
dMentee = findOptimalDifLevels(p_min, p_max, k);
foreach studySession ∈ S do
    u_n = retrieveMentor(studySession);
    u'_n = retrieveMentee(studySession);
    δ = retrieveTopics(studySession);
    reciprocalScore_{M×k} = {};
    for m ← 0 to M do
        for i ← 0 to k do
            G_nmi = g(φ_nm, dMentor_i, σ, δ);
            G_n'mi = g(φ_n'm, dMentee_i, σ, δ);
            reciprocalScore_mi = harmonicMean(G_nmi, G_n'mi);
        end
    end
    Q' ← findBestquestions(reciprocalScore);
    S'.append([u_n, u_n', δ, Q']);
end
return S';
```

6 Evaluation

6.1 Simulated data

User data were simulated for 20,000 users, with competencies (α) drawn from a random normal distribution with mean μ = .65 and standard deviation sd = .25. Users with competency greater or equal to .9 (μ + 1sd) were assigned to providing support roles ($N = 1,565$), and users with competency less than or equal to .65 (μ) assigned to seeking support roles ($N = 10,017$). Item data were then simulated for 5000 questions (Q), with difficulties (β) between 0 and 1 drawn from a random uniform distribution.

Simulated user data for Φ were generated using the three parameter logistic model from classical Item Response Theory outlined in [7] as $\phi_{nm} = c + \frac{1-c}{1+e^{\beta-\alpha}}$, where β represents the difficulty level of question m, α represents the competency level of user u_n and c representing the probability of randomly guessing correctly (e.g. c=.25 for a four-choice question). We used a low value for random guess probability $c = .05$ to better reflect the current context, which is to encourage discussion and knowledge-building rather than simply selecting the correct response, and to better simulate the users' states of knowledge about the question were they to be given no opportunity to guess from a list of answers known to contain the correct response.

Simulated values of Φ typically ranged between $\approx$ .50 - .75 and $\approx$ .24-.67 for providing support and seeking support users, respectively. Recommender parameters for all simulations were set to $p_{min} = .1$ and $p_{max} = 1$. Despite these values being outside the range of simulated values in Φ, the recommender functions were designed to capture the best fit from the available data, and setting $p_{max} = 1$ ensures that where required (such as in item 1), the item most likely to be correctly answered (highest ϕ value) is selected.

6.2 Scalability and Quality

Firstly, to assess the recommender's ability to return the optimal items under a range of values for σ, we used the simulated Φ data but planted ideal values for the relevant difficulty gradient at known items. If the recommender worked correctly, these items would be returned in the list of recommendations without interference from other items. Using a random sample of 300 study sessions (pairs of users) from the simulated data, and a high value of $k = 50$, it was found that all planted values were reliably selected by the recommender without error. This process was used to justify the theoretical correctness of the algorithm.

The quality of recommendations was conceptualised as the extent to which the subjective difficulties (ϕ_{nm}) of recommended items reflects the optimal difficulties calculated by the difficulty gradient D. This was approached using recommendations for $k = 5$, 10, 20 and 50 using independent random samples (with replacement) of 300 sessions from the simulated data set.

The **Root-Mean-Square Error** (RMSE) represents the standard deviation of the differences between model data and observed data and is widely used as a measure of accuracy. In each sampled partnership, the RMSE was calculated between the values in D (model) and the actual ϕ values of the items that were recommended (observed).

Spearman's rank correlation coefficient (Spearman's rho) is a measure of the dependence between item rankings in two sets. It is the nonparametric equivalent of Pearson's correlation (r), ranging from 0 to 1 and is appropriate for discrete ordinal variables. It was used in this evaluation, in conjunction with the RMSE, to assess the extent to which the difficulty of recommended items aligned with the model values in D. Specifically, for users receiving support Spearman's rho was used to check that recommended items were in ranked order of difficulty, as would be expected given the difficulty gradient D.

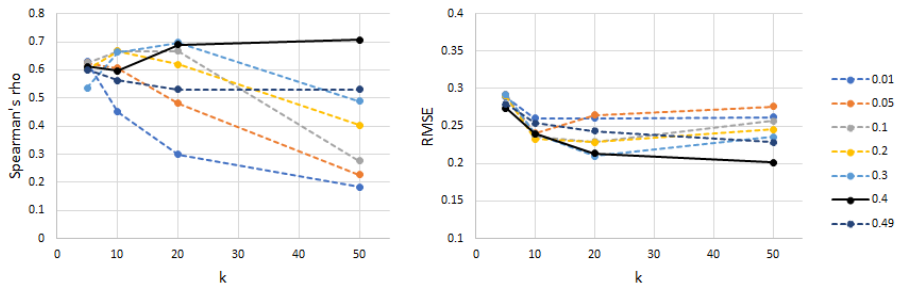


Fig. 4: Mean Spearman's correlation and Root-Mean-Square Error for n=300 samples with varying k and overlap between Gaussian functions.

The average correlation was moderate-to-high and varied with the setting for z (controlling the overlap between sequential Gaussian functions). Recommendations using large values of z (incurring less overlap) degraded with higher values for k. Figure 4 shows the dependency between the amount of overlap and the quality measures. Interestingly this was independent of the number of items (Q) in the simulated data. The RMSE was both independent of the size of Q and k, consistently $\approx .27$. Simulations demonstrated that allowing an overlap value of $\approx .4$ (shown in bold lines) yielded the best recommendations, especially for higher values of k where restricting the overlap severely degraded the ability of recommendations to fit the difficulty gradient.

Running time was examined over a range of 5 to 50 items recommended (k) and a range of 100 to 500 study partnerships, for 3000 repository items. The algorithm runs in linear time for both dimensions as shown in Figure 5. It is also linear in the number of repository items. In practice the number of recommended items k should be small, approximately five per hour of study, to encourage discussion and communication rather than having students complete as many multiple choice items as quickly as possible.

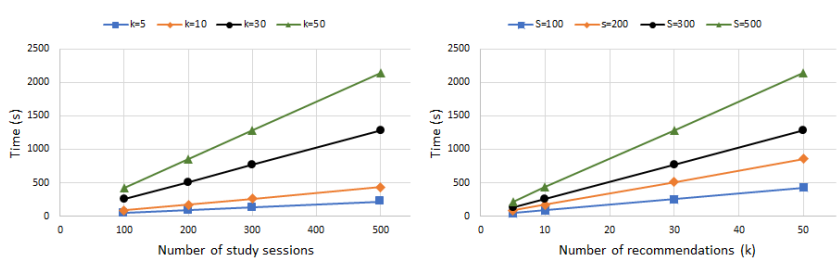


Fig. 5: Run time for variable k and number of study sessions with repository of 3000 items

7 Conclusion and Future Work

The aim of this paper was to establish a method that can suitably recommend study content in the form of multiple-choice questions from a large repository to support peer learning study sessions. Items were recommended on the basis of their subjective difficulty for each user, as derived through existing knowledge tracing algorithms, such that a set of recommended items conformed concurrently to two propositions - that items should increase in difficulty for the user who is receiving peer support, and that all items should be easily answered by the user providing peer support. A simple Gaussian scoring model was applied and using synthetic data containing known ideal values, it was demonstrated that the recommender reliably finds the optimal items at scale for large numbers of users, recommendations and repositories of items. For simulated data (without ideal values), the root-mean-square error and Spearman's rank correlation of the recommended item difficulties and the proposed ideal difficulties were calculated to evaluate the extent to which recommendations reflected the desired difficulty gradient and correct ordering, respectively.

These statistics are naturally dependent on the availability of values in the data set that closely approximate the model gradient, and it is a reasonable limitation to expect in future testing with real students. Nonetheless, the moderate-to-high positive values of Spearman's coefficient demonstrates that the recommended ordering of items in increasing difficulty is achievable, and that the Gaussian model ensures the best items are chosen from those available. The recommender performs more consistently for smaller values of k, with the quality dependent on the overlap settings for higher values of k. The quality is otherwise unaffected by the size of the item repository or cohort of students.

The success of the recommender (as defined by the two propositions) in simulated examples paves the way for the incorporation of feedback mechanisms from real students who engage in peer learning through RiPPLE's recommendations. Students can complete the recommended MCQs online, allowing the platform to collect session data which can then be used to monitor students' progress. This would, however, be a poor measurement of primary aim of stimulating discussion, interaction and learning. Indeed this measurement problem is a common limitation of many learning analytic tools that often collect only proxy indicators at best. Collecting qualitative feedback from students for each recommended session in the form of a well-designed and very brief survey component will be valuable, both to the refinement of the recommender systems and to understanding students' engagement with, and effectiveness of, the course content. Properly designed, recommender systems such as those used by RiPPLE and proposed in this paper can be deployed as both learning resources and measurement tools for further research.

References

1. Baker, R., Corbett, A.T., Aleven, V.: More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In: International Conference on Intelligent Tutoring Systems. pp. 406–415. Springer (2008)
2. Cen, H., Koedinger, K., Junker, B.: Learning factors analysis-a general method for cognitive model evaluation and improvement. In: Intelligent tutoring systems. vol. 4053, pp. 164–175. Springer (2006)
3. Conole, G.: Review of pedagogical models and their use in e-learning. Milton Keynes: Open University (2010)
4. Corbett, A.: Cognitive computer tutors: Solving the two-sigma problem. User Modeling 2001 pp. 137–147 (2001)
5. Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction 4(4), 253–278 (1994)
6. Dawson, S.: A study of the relationship between student communication interaction and sense of community. The Internet and Higher Education 9(3), 153–162 (2006)
7. Drasgow, F., Hulin, C.L.: Item response theory. (1990)
8. Goldschmid, B., Goldschmid, M.L.: Peer teaching in higher education: a review. Higher Education 5(1), 9–33 (1976)
9. Guy, I.: Social recommender systems. In: Recommender Systems Handbook, pp. 511–543. Springer (2015)
10. Hong, W., Zheng, S., Wang, H., Shi, J.: A job recommender system based on user clustering. Journal of Computers 8(8), 1960–1967 (2013)
11. Khajah, M.M., Huang, Y., González-Brenes, J.P., Mozer, M.C., Brusilovsky, P.: Integrating knowledge tracing and item response theory: A tale of two frameworks. In: Proceedings of Workshop on Personalization Approaches in Learning Environments (PALE 2014) at the 22th International Conference on User Modeling, Adaptation, and Personalization. pp. 7–12. University of Pittsburgh (2014)
12. Khosravi, H.: Recommendation in personalised peer-learning environments. arXiv preprint arXiv:1712.03077 (2017)
13. Khosravi, H., Cooper, K., Kitto, K.: Riple: Recommendation in peer-learning environments based on knowledge gaps and interests. JEDM-Journal of Educational Data Mining 9(1), 42–67 (2017)
14. Lemire, D., Boley, H., McGrath, S., Ball, M.: Collaborative filtering and inference rules for context-aware learning object recommendation. Interactive Technology and Smart Education 2(3), 179–188 (2005)
15. Mangina, E., Kilbride, J.: Evaluation of keyphrase extraction algorithm and tiling process for a document/resource recommender within e-learning environments. Computers & Education 50(3), 807–820 (2008)
16. van der Meer, J., Scott, C.: Students experiences and perceptions of peer assisted study sessions: towards ongoing improvement. Journal of Peer Learning 2(1), 3–22 (2009)
17. Newcomb, T.: A conversation with theodore newcomb. Psychology Today, Sept pp. 73–80 (1974)
18. Pardos, Z., Heffernan, N.: Kt-idem: introducing item difficulty to the knowledge tracing model. User Modeling, Adaption and Personalization pp. 243–254 (2011)
19. Pavlik Jr, P.I., Cen, H., Koedinger, K.R.: Performance factors analysis–a new alternative to knowledge tracing. Online Submission (2009)

20. Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L.J., Sohl-Dickstein, J.: Deep knowledge tracing. In: Advances in Neural Information Processing Systems. pp. 505–513 (2015)
21. Pizzato, L., Rej, T., Akehurst, J., Koprinska, I., Yacef, K., Kay, J.: Recommending people to people: the nature of reciprocal recommenders with a case study in online dating. User Modeling and User-Adapted Interaction 23(5), 447–488 (2013)
22. Pizzato, L., Rej, T., Chung, T., Yacef, K., Koprinska, I., Kay, J.: Reciprocal recommenders. In: 8th Workshop on Intelligent Techniques for Web Personalization and Recommender Systems, UMAP (2010)
23. Potts, B., Khosravi, H., Reidsema, C., Bakharia, A., Belonogoff, M., Fleming, M.: Reciprocal peer recommendation for learning purposes. In: Proceedings of the 8th International Conference on Learning Analytics and Knowledge (2018)
24. Ross, M.T., Cameron, H.S.: Peer assisted learning: a planning and implementation framework: Amee guide no. 30. Medical teacher 29(6), 527–545 (2007)
25. Sha, L., Hong, P.: Neural knowledge tracing. In: International Conference on Brain Function Assessment in Learning. pp. 108–117. Springer (2017)
26. Thai-Nghe, N., Drumond, L., Horváth, T., Krohn-Grimberghe, A., Nanopoulos, A., Schmidt-Thieme, L.: Factorization techniques for predicting student performance. Educational Recommender Systems and Technologies: Practices and Challenges pp. 129–153 (2011)
27. Van Der Heijden, B., Boon, J., Van der Klink, M., Meijs, E.: Employability enhancement through formal and informal learning: an empirical study among dutch non-academic university staff members. International journal of training and development 13(1), 19–37 (2009)
28. Vygotsky, L.S.: Mind in society: The development of higher psychological processes. Harvard university press (1980)
29. Yudelson, M.V., Koedinger, K.R., Gordon, G.J.: Individualized bayesian knowledge tracing models. In: International Conference on Artificial Intelligence in Education. pp. 171–180. Springer (2013)