Reciprocal Peer Recommendation for Learning Purposes

Boyd A. Potts
The University of Queensland
b.potts@uqconnect.edu.au

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

Carl Reidsema
The University of Queensland
c.reidsema@uq.edu.au

Aneesha Bakharia
The University of Queensland
a.bakharia1@uq.edu.au

Mark Belonogoff
The University of Queensland
mark.belonogoff@gmail.com

Melanie Fleming
The University of Queensland
melanie.fleming@uq.edu.au

ABSTRACT

Larger student intakes by universities and the rise of education through Massive Open Online Courses has led to less direct contact time with teaching staff for each student. One potential way of addressing this contact deficit is to invite learners to engage in peer learning and peer support; however, without technological support they may be unable to discover suitable peer connections that can enhance their learning experience. Two different research subfields with ties to recommender systems provide partial solutions to this problem. Reciprocal recommender systems provide sophisticated filtering techniques that enable users to connect with one another. To date, however, the main focus of reciprocal recommender systems has been on providing recommendation in online dating sites. Recommender systems for technology enhanced learning have employed and tailored exemplary recommenders towards use in education, with a focus on recommending learning content rather than other users. In this paper, we first discuss the importance of supporting peer learning and the role recommending reciprocal peers can play in educational settings. We then introduce our open-source course-level recommendation platform called RiPPLE that has the capacity to provide reciprocal peer recommendation. The proposed reciprocal peer recommender algorithm is evaluated against key criteria such as scalability, reciprocality, coverage, and quality and shows improvement over a baseline recommender. Primary results indicate that the system can help learners connect with peers based on their knowledge gaps and reciprocal preferences, with designed flexibility to address key limitations of existing algorithms identified in the literature.

CCS CONCEPTS

• **Information systems** → *Recommender systems*; • **Applied computing** → *Collaborative learning*;

KEYWORDS

Reciprocal Recommender, Peer Learning, Peer Support, RecSysTEL

LAK '18, March 7–9, 2018, Sydney, NSW, Australia

ACM ISBN 978-1-4503-6400-3/18/03...$15.00
https://doi.org/10.1145/3170358.3170400

ACM Reference Format:
Boyd A. Potts, Hassan Khosravi, Carl Reidsema, Aneesha Bakharia, Mark Belonogoff, and Melanie Fleming. 2018. Reciprocal Peer Recommendation for Learning Purposes. In *LAK '18: International Conference on Learning Analytics and Knowledge, March 7–9, 2018, Sydney, NSW, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3170358.3170400

1 INTRODUCTION

Increased enrolments in higher education and the global expansion of Massive Open Online Courses (MOOCs) place a greater emphasis on designing more innovative and flexible learning options [12]. The substantially larger student intakes accelerated by the 2008 Bradley Review in Australia [2] coincided with an increased need to counter downward trends in student engagement. As institutions increased their use of online learning to cope with scale as well as deliver higher quality blended courses, the risk of social isolation increased. Creating blended learning environments not only creates opportunities for increased engagement in learning but also new approaches to increase students' social/peer networks. Both engagement and social networks are important contributors to student success [54] and social isolation is a significant negative contributor to student success. This is particularly true of online learning [38], but also occurs in blended and face to face environments. One approach for mitigating potential negative consequences of technology enhanced learning is to adopt a community of inquiry pedagogy [32], in which peer-learning software systems focus on providing students with the ability to leverage the inherent power of such large cohorts to improve their learning and their university experience [55].

Supplementing the development of a learning community [37] with fit-for-purpose technology not only allows an increase in the frequency of interaction and communication between students, but creates an awareness of a community for them to draw upon or otherwise "belong" to [12]. Providing support tools and designing curricula that create opportunities for students to utilise peer networks does not alleviate all challenges associated with peer learning, however improved learning outcomes can follow from meaningful, purposive communications with peers. Learners who engage in peer learning and peer support not only contribute to the existing community, but can create entirely new communities of practice [10].

Falchikov [17] identified that such learning communities are also valuable for other reasons. For instance they lead to the development of "cognitive or intellectual skills or to an increase in knowledge and understanding" (p. 3), or to the development of communication and professional skills. There is much debate over the responsibility of educators to provide such 'soft skills' as part of curricula in light of finite time resources and competing demands from more traditional competencies. However, providing an environment that encourages the development and practice of soft skills is possible. Van Der Heijden and colleagues [51] found evidence for the participation in networks as an important predictor of employability, with interactions therein constituting a strong catalyst for learning as a highly social process. Empirical work on the construct of employability included this participation in a dimension called corporate sense, which can be considered to represent social competence [22].

Tasks relying heavily on interpersonal and cross-departmental engagement emerge in most workplaces, even if not explicitly designed from a team perspective [14]. These behavioural qualities are widely valued by prospective employers [1, 3, 18, 30, 39, 46], and this demand is also reflected in graduates' perceptions, with many reporting both that such skills are of high value in their careers and were poorly addressed in their studies [13].

The social dimension in education is receiving attention as institutions seek to nurture social learning opportunities and bolster students' success. From a more immediate perspective, social support is essential to student engagement, wellbeing and retention in the first year of university [19, 42, 54]. As such, it is in the interests of education providers to support the development of social ties wherever possible, towards equipping graduates with successful social skills as well as domain knowledge.

With the benefits of peer and social learning having been established and widely accepted, methods for their effective facilitation present the current challenge. In large classes teachers face a daunting task in connecting learners with peers who can adequately help each other for collaborative assessments or in reaching their academic aspirations generally. Many students in their first year of university will have difficulty navigating their new academic and social environment and adequately exploiting the available resources [11].

Recommender systems [48] offer a potential solution by providing sophisticated filtering techniques to help people find the resources that they need. Specifically, the field of recommender systems for technology enhanced learning (RecSysTEL; [34]) has focused on providing recommendations in the educational setting, with much of the primary research on the recommendation of relevant content and learning objects [15, 35, 36, 52, 53].

In a different body of work on recommending other people, rather than objects, most of the focus has been on online dating and we argue that the general framework of these reciprocal recommender systems [44] can be adapted successfully to educational settings for social learning. We introduce an open-source course-level platform called RiPPLE (Recommendation in Personalised Peer Learning Environments) that has the capacity of providing reciprocal peer recommendation, enabling learners to provide learning support, seek learning support, or find study partners. A further technical contribution of the paper is a new reciprocal recommendation algorithm, designed to support peer learning opportunities.

In preparation for trialling RiPPLE in four large courses at a research-intensive university, 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 cohort, distributions of competencies, availabilities and willingness to collaborate. Synthetic data sets were created for this purpose. Primary results indicate that the system can help learners connect with peers based on their knowledge gaps and reciprocal preferences.

2 BACKGROUND

The area of reciprocal peer recommendation is closely related to peer learning and group formation and two subfields tied to recommender systems, namely, RecSysTEL and Reciprocal Recommender Systems. Section 2.1 provides a brief review of the past work closely related to peer learning and group formation. Section 2.2 presents brief reviews of the literature on RecSysTEL and Reciprocal Recommender Systems and considers how these fields relate to reciprocal peer recommendation.

2.1 Peer Learning and Group Formation

Early work in the area of peer recommending in a university course resulted in the Peer Help System, PHeLpS [20] which, among other functions, provided students with a means of finding a peer helper and facilitating direct communication between both parties. Comprehensive student models based on general information and evaluations of knowledge, competency and willingness to provide help contributed to a score for each student that could be used by the system to select appropriate helpers when required. Knowledge representation models were employed to make accurate inferences about student knowledge that was not explicitly assessed which greatly assisted in the system's ability to find the most suitable peers [20].

PHelpS and a subsequent system I-Help [6] facilitated effective use of existing peer networks as a learning resource with a focus on just-in-time requests for help in an online environment. As such, the goals did not necessarily involve the encouragement of soft skills, social networking or collaboration but rather the provision of a new resource for receiving assistance with specific problems in the peer community. More recently, tools such as MOOCchat which uses a structured, real-time (synchronous) chat session have been employed for students to engage in a formatively assessed discussion of concepts in the online component of a very large (1000+ student) on-campus, hands-on design projects course [47]. Asynchronous peer-learning support tools such as Piazza or Casper [23, 49] allow students to post queries for assistance on their prescribed course assessment or learning activities in their own time.

Many studies in the domain of peer-recommending have focused on solving the problem of group formation. A peer-finding service called DEPTHS [25] used participants' competencies such as social connections and knowledge level to suggest potential collaborators in a core collaborative project assessment. DEPTHS used users' search history of online resources to identify peers who were interacting with similar content [25]. In this regard, peer recommending in DEPTHS was based on emergent properties of users' interactions with the system (implicit preferences) rather than explicit preferences.

Another system, DIANA [40], addressed the formation of small heterogeneous groups for the purposes of collaborative learning. Using arrays of student characteristics (estimated subject knowledge, communicative skills and leadership skills), DIANA employed a genetic algorithm to form collaborative groups in a university programming course. An evaluation using course grades found that groups formed using the algorithm performed significantly better than groups that were either randomly assigned or self-organised [40].

TeamAnneal [26] used a simulated annealing algorithm to create student groups based on demographic information, academic performance, class sign-on information or other constraints to ensure groups have the desired characteristics for particular learning activities.

Olakanmi and Vassileva [41] reviewed several other grouping algorithms and identified four limitations of the current approaches: constraints on the number of user preferences/attributes resulting in inflexibility where a user's requirements do not align perfectly with those built into the system, "orphaned learners" who are not grouped successfully, self-report evaluation, and a focus on enhancing the learning activities rather than benefits to the learners' skills. We designed our algorithm with these criticisms in mind.

2.2 RecSysTEL and Reciprocal Recommender Systems

Recommender systems for technology enhanced learning is an active and rapidly evolving research field. For example, Drachsler et al. [15] performed an extensive classification of 82 different systems, and Erdt et al. [16] reviewed the various evaluation strategies that have been applied in the field. Together these articles provide recent comprehensive surveys that considered more than 200 articles spanning over 15 years. In the educational setting, much of the primary research on recommender systems is directed at the recommendation of relevant content and resources for learning. For example Lamire et al. [31] used inference rules to provide context-aware recommendations on learning objects, Bousbahi and Chorfi [5] recommended courses to learners, and Mangina and Kilbride [33] recommended documents and resources within e-learning environments to expand or reinforce knowledge. Interestingly, Khosravi et al. [28] combined content based filtering with collaborative filtering (CF) to make recommendations in a student-authored repository. When recommending learning objects (e.g. questions) related to student knowledge gaps, Cazella et al. [8] provided a semi-automated, hybrid solution based on CF (nearest neighbour) and rule-based filtering, while Thai-Nghe et al. [50] used students' performance predictions to recommend more appropriate exercises.

Reciprocal recommender systems has a literature rich with applications in online dating [43], job matching [24], and social networking [21], each highly tailored by the best consideration of the specific needs and constraints of the immediate environment. Fundamentally they entail operationalised user preferences and seek solutions such as a list of recommendations that match those preferences to a higher degree than competing items. In reciprocal recommendation, items are usually other users whose preferences must also be fulfilled and so entail a higher level of complexity than other recommender systems [44]. Much of the research in social recommending has been developed and evaluated in existing social networks and particularly online dating sites where the site and/or the individual users may have a substantial data history which can be used to train machine learning algorithms [7, 9, 29]. In peer recommending the environment may be a newly formed cohort of learners (such as a first-year university course) who have no data history but can volunteer preferences on a narrow selection of relevant variables for the purposes of matching.

By their nature, recommender systems across domains will exhibit similarities to a high degree. All systems incontrovertibly share the same fundamental goal - to provide recommendations that are well-received by users according to their preferences, explicit and implicit, in an otherwise overwhelming information environment where the likelihood of users successfully finding preferred items without technological assistance is very low. However the nature of domain-specific information and definition of a successful recommendation is so heavily context- and goal-dependent that little more than the general way of thinking can be adapted or generalised from existing systems to new domains. This is particularly true of the formulation of user preference models upon which recommendations are to be based, making them necessarily bespoke.

As such, while reciprocal peer recommendation has similarities to traditional recommendations in education and reciprocal recommendations in other domains, it does have a distinct nature. Although some primary research has been done on utilising peer learning and support for improving learning and enhancing the learning experience of students, the area remains fertile for many research and development opportunities. In this paper we introduce a platform that can enable adoption of peer support systems for both large on-campus and online courses with competency-based user preference models.

3 PROVIDING RECIPROCAL PEER RECOMMENDATION IN RIPPLE

This section introduces both the proposed platform and reciprocal recommender system. Section 3.1 provides an overview of the developed platform called RiPPLE. Section 3.2 presents a formal description of the problem under investigation using mathematical notation. Section 3.3 defines a non-reciprocal compatibility function, which is used in the reciprocal peer recommendation algorithm introduced in Section 3.4. Finally, Section 3.5 introduces a small example to illustrate how the algorithm works. Table 1 provides a summary of the notation used in this section.

3.1 Platform Description

RiPPLE is an adaptive, web-based, student-facing, open-source platform that aims to provide a personalised and flexible learning experience that better suits the needs and expectations of the digitally minded learners of the 21st century. RiPPLE (1) empowers learners to contribute to co-creation of learning content; (2) recommends learning content tailored to the needs of each individual; and (3) recommends peer learning sessions based on the learning preferences and needs of individuals that are interested in providing learning support, seeking learning support or finding study partners [27].

Source code [1] and a prototype [2] of the platform are available through a GitHub account.

[1] https://github.com/hkhosrav/RiPPLE-Core
[2] https://hkhosrav.github.io/RiPPLE-Core/

The prototype currently only supports the use of the platform on desktops and laptops and does not extend to use on other smart devices.

Individuals nominate their availability in hourly blocks between 08:00 and 20:00, Monday to Friday, and their preferences for providing or seeking peer learning support and finding study partners across the range of course-relevant topics. An example of the interface is provided in Figure 1. An indicator of the individual's competency is provided in the form of a coloured bar chart superimposed over the list of topics. The option to provide peer support is only available for those topics in which the student meets a required competency threshold, denoted in the bar chart by the colour blue. Competency levels are derived initially from student's responses to multiple choice questions (MCQs) which are also contained in the RiPPLE platform. Competencies can be updated using the individual's cumulative performance on assessment items and quizzes progressively during the teaching period using algorithms described by Khosravi et al. in [28].

Figure 2 provides an example of a peer support recommendation in the student interface, identifying the potential peer learning supporter and the two topics for which support is available. Students then have the option to ignore or request a meeting with the recommended peer at the nominated time and date.

3.2 Problem Formulation

Assume that a platform that enables students to provide peer learning support such as RiPPLE is being used within a course. Let U denote the number of learners that are using the platform, and u, u_1 and u_2 refer to arbitrary learners. Let L denote the number of distinct topics that are covered in the course, and l to denote an arbitrary topic. Let T denote the number of weekly available time slots for scheduling a study session, and t to denote an arbitrary time slot. Finally, let Q present the number of different roles a learner can have during a study session and q present an arbitrary role; in the current setting, Q is set to three: q=1 is used to present providing peer learning support, q=2 is used to present seeking peer learning support, and q=3 is used to present searching for study partners.

Let's assume that the following information can be collected through the platform:

- **Requests,** $R_{U \times L \times Q}$: A three dimensional array where $R_{ulq} =$ 1 indicates that user u has indicated interest in participating in a study session on topic l with role q.
- **Competencies,** $C_{U \times L}$: A two-dimensional array in which C_{ul} shows the competency of user u in topic l. Values in C are in the range of 0 to 100.
- **Availability,** $A_{U \times T}$: A two-dimensional array in which $A_{ut} =$ 1 shows that user u is available at time t, and $A_{ut} = 0$ shows that user u is not available at time t.
- **Preferences,** $P_{U \times Q}$: A two dimensional array in which P_{uq} shows the competency preference of user u in role q.

The aim of the platform is to provide a list of up to k recommendations for each user, where a recommendation is of the form [u_1, u_2, [l], [q], t, s] indicating that user u_1 receives recommendation to connect with user u_2 on a list of topic [l] on a list of roles [q] at time t with a reciprocal score of s. For each given recommendation, the platform attempts to satisfy the preferences of both users as much as possible. The output of the recommender system is a list of N **Recommendations,** $Recoms_N$ that include up to k recommendations for each learner. The platform can use *Recoms* to display a set of recommendations to each learner.

Course Parameters	
U	Number of users
L	Number of topics
T	Number of available time slots per week
Q	Number of roles, which is three (q=1, provide learning support), (q=2, seek learning support), (q=3, search for study partner)
Input Data	
$R_{U \times L \times Q}$	A three dimensional array where $R_{ulq} = 1$ indicates that user u has shown interest in providing role q in topic l.
$C_{U \times L}$	A two-dimensional array in which C_{ul} shows the competency of user u in topic l.
$A_{U \times T}$	A two-dimensional array in which A_{ut} shows the availability of user u at time t.
$P_{U \times Q}$	A two-dimensional array in which P_{uq} shows the competency preference of user u for roles q.
k	Maximum number of recommendations to be provided for each user.
Competency preference model	
$J_{u_1 u_2 l}$	The joint competency of $u1$ and $u2$ in a topic l.
τ	The desired minimum threshold for any $J_{u_1 u_2 l}$.
α	The leniency parameter for τ.
σ	The standard deviation for the distribution of competency compatibility scores.
ϵ	A very small constant number used for setting the score of users that are incompatible.
Output	
$Recoms_N$	A list of N recommendations where each recommendation is a list in the form of [u_1, u_2, [l], [q], t, s] indicating that user u_1 has received recommendation to connect with user u_2 on topic list [l] holding roles [q] at time t with a reciprocal score of s.

Table 1: A summary of the notation used in Section 3.

3.3 Defining a Competency Preference Model and Compatibility Function

In the context of reciprocal recommenders, compatibility between two users can be defined generally as the extent to which their goals and preferences align or are otherwise complementary in accordance with the users' objectives. In the current system, the primary challenge is in defining compatibility as a function of learners' requests (R), competencies (C), availability (A), and preferences (P). Compatibility in terms of R and A are considered as hard constraints. As such, the compatibility score $s_{u_1 u_2}$ of two users u_1 and u_2 that have incompatible requests or availabilities is set to a small constant value ϵ. This is done via Algorithm 1, which is defined in Section 3.4. This section outlines our approach for computing $s_{u_1 u_2}$ only for users u_1 and u_2 that are compatible in terms of R and A.

The score of $s_{u_1 u_2}$ is computed as the product of two factors: (1) the effectiveness of a study session based on the learners' competencies and (2) the preferences of u_1. In computing $s_{u_1 u_2}$, only the

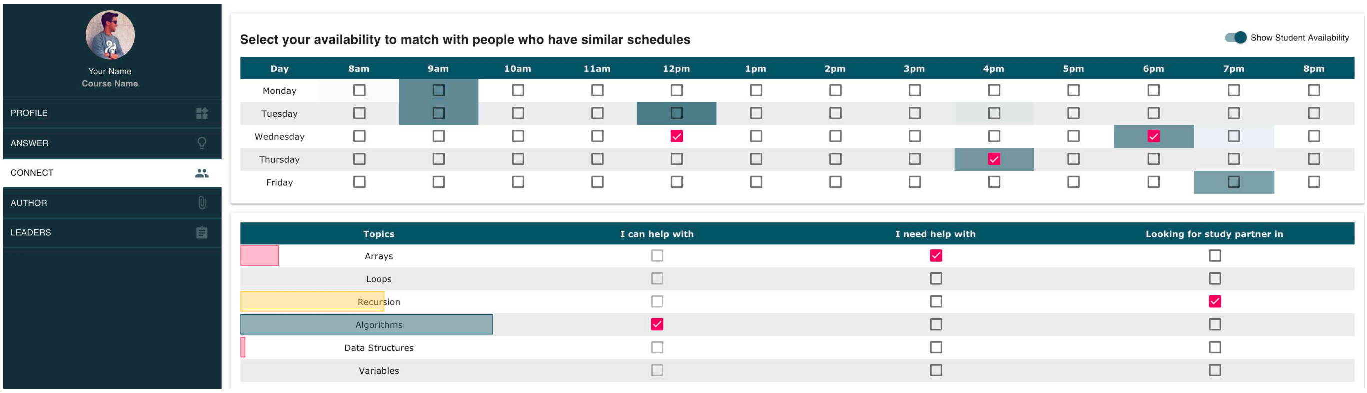


Figure 1: Example student interface for preferences and competencies.

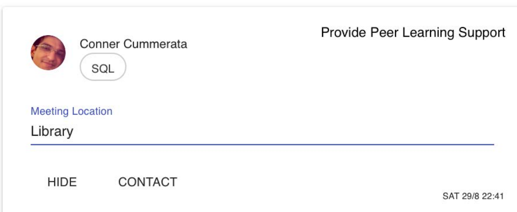


Figure 2: Example recommendation for a learner to provide peer learning support.

preferences of u_1 are considered. Section 3.4 later discusses how $s_{u_1u_2}$ and $s_{u_2u_1}$ may be combined towards computing a reciprocal score.

The first factor, inspired by the findings of Blumenfeld [4] on successful peer learning, proposes that for a peer learning session to contribute to effective learning the peers joint competency should be above a certain threshold. A parameter τ is introduced that reflects the minimum desired threshold, which may be set by one of the course instructors or using a validation set. In the absence of an empirical basis, we define the joint competency of a partnership between u_1 and u_2 on a topic l as the magnitude of the vector of their competencies on topic l in a two-dimensional Cartesian space given by:

$$J_{u_1u_2l} = \sqrt{C^2_{u_1l} + C^2_{u_2l}} \tag{1}$$

The defined joint competency $J_{u_1u_2l}$ is used in a logistic function H to compute the extent to which the partnership of u_1 and u_2 on topic l meets the expected desirable threshold determined by τ.

$$H(u1, u2, l, \tau, \alpha) = \frac{1}{1 + e^{-\frac{J_{u_1u_2l}-\tau}{\alpha}}} \tag{2}$$

The score of the session to be held between u_1 and u_2 on topic l is calculated as the height of the logistic function, H. A scaling parameter α is used to determine the leniency of this measure for a pairing of u_1 and u_2 on a topic l such that $J_{u_1u_2l} < \tau$. This leniency may be useful in reducing orphaned users during implementation in sparse cohorts. It is interesting to note that H is symmetric; therefore, $\forall u_1, u_2\ H(u_1, u_2, l, \tau, \alpha) = H(u_2, u_1, l, \tau, \alpha)$.

The second factor is based on the preferences of u_1, which are shown by a vector P_{u_1}. In this vector, P_{u_1q} shows the competency preference of user u_1 for role q. For example for, p_{u_11} = -10 means that u_1 prefers providing support to peers with a competency of around 10 less than C_{u_l1} and p_{u_12} = 30 means that u_1 prefers seeking support from peers that have a competency of around 30 more than C_{u_1l}. To be able to provide meaningful recommendations, we constrain eligibility by role such that users (1) provide support to less competent learners, (2) seek support from more competent learners and (3) find study partners with relatively similar competency to that of their own.

The contribution to the compatibility score $s_{u_1u_2}$ in a topic l with a role q is calculated as the height of the Gaussian function G at value C_{u_2l} with centre $C_{u_1l} + P_{u_1q}$ with a standard deviation of σ on a 100 point scale.

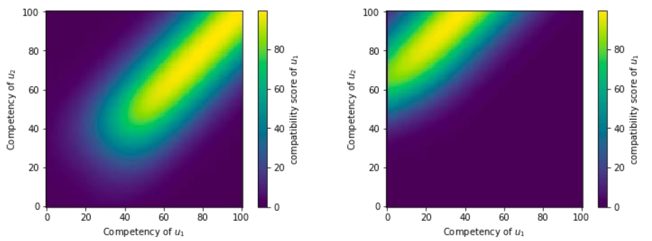
$$G(u_1, u_2, q, l, P_{u_1q}, \sigma) = 100e^{\frac{-(C_{u_2l}-(C_{u_1l}+P_{u_1q}))^2}{2\sigma^2}} \tag{3}$$

The parameter σ models the leniency of users in terms of being matched with peers that do not exactly fit their specified preferences. The function G is asymmetric, only cooperating with the preferences of u_1.

Taken together, $s_{u_1u_2}$ is calculated as the product between the two values from the Gaussian and logistic functions, summed over all matched topics and u_1's associated role in each topic as shown below.

$$s_{u_1u_2} = \sum_{(l_i,q_i)\in([l],[q])} H(u_1, u_2, l_i, \tau, \alpha) \times G(u_1, u_2, q_i, l_i, P_{u_1l_iq_i}, \sigma) \tag{4}$$

Figure 3 displays the resulting distributions of compatibility scores of two users u_1 and u_2 from the perspective of u_1 on a topic l for arbitrary parameters $\alpha = 10$, $\sigma = 20$, $\tau = 40$ and P_{u_1} = [-30, 60, 0]. Figure 3a shows the compatibility of u_1 with other users as a function of competency, where u_1 is seeking study partners. Compatibility scores are normally distributed around $C_{u_1l} = C_{u_2l}$, with τ minimising compatibility between two users below the threshold competency. Figure 3b shows the compatibility of u_1 with other users as a function of competency, where u_1 is seeking peer learning support. In this example, u_1 prefers peer learning supports with 60 points higher in competency for successful tutoring. Therefore, compatibility scores are normally distributed around $C_{u_1 l} + 60 = C_{u_2 l}$. In this example, the constraint set by τ does not have an impact on the score as pairing u_1 with peers with competency $C_{u_1 l}+60$ guarantees a joint competency higher than the desired minimum threshold.



(a) Seeking partners compatibility **(b) Seeking support compatibility**

Figure 3: Compatibility scoring for u_1

3.4 Reciprocal Peer Recommendation Algorithm

Algorithm 1 presents a reciprocal peer recommendation algorithm. This algorithm takes R, C, A, P and k as input and generates a list of recommendations for the entire cohort that includes up to k recommendations per user. This algorithm could be executed by RiPPLE at the end of each week to provide recommendation on study sessions for the coming week.

At a high level, the algorithm generates a list of recommendations for each user that are combined into a single list, which is ultimately the output of the algorithm. The outer for loop of the algorithm selects a user u_1. For each user other than u_1, referred to as u_2, the algorithm uses A to find a mutually convenient time slot for a session to be held between u_1 and u_2. The algorithm then uses R to find a set of matching topics and a set of matching roles associated with these topics to be covered during the session. The reciprocal score between user u_1 and u_2 is captured in $Score[u_2]$. This score is set to ϵ for users that are not compatible with u_1. For users that satisfy the hard constraints, *Score* is calculated as the harmonic mean of the compatibilities from user u_1 to u_2 and vice-versa, which are computed using the competency preference model in section 3.3. Compared to the arithmetic mean, the harmonic mean guarantees to provide a much smaller reciprocal score for users whose compatibility towards each other differ considerably. Therefore, the algorithm would prioritise recommendations that benefit both users [43].

Figure 4 shows the distribution of competency preferences and the reciprocal scores for u_1 and u_2 on topic l, where $P_{u_1 1} = -10$ (u_1 is providing support and prefers users who have competency 10 points lower than $C_{u_1 l}$) and $P_{u_2 2} = 80$ (u_2 is seeking peer support with a preference for those who have competency 80 points higher than $C_{u_2 l}$). The extent to which both users are recommended to each other is defined by the harmonic mean distribution shown in the third frame of Figure 4.

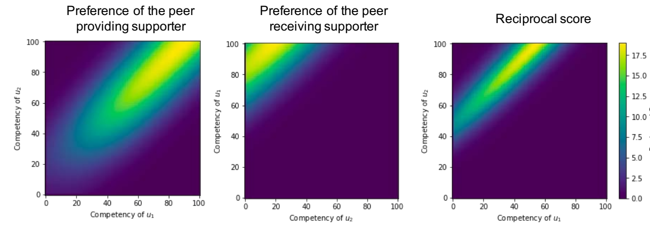


Figure 4: Distribution of competency preferences and the reciprocal scores for u_1 and u_2.

Once all of the reciprocal scores in $Score[U]$ have been computed, the k users with the highest score are selected. For any of these users that are compatible with u_1 (i.e., their reciprocal score is bigger than ϵ), a recommendation is appended to $u_1Recoms$. The set of the recommendations for each user are then added to *Recoms*, which store the recommendations for the entire cohort.

Algorithm 1: Generating a list of reciprocal recommendations for the entire cohort

Input : $R, C, A, P, k, \tau, \alpha, \sigma$
Output : *Recoms*

$U \leftarrow getusers(R)$;
$Recoms \leftarrow []$;
foreach $u_1 \in U$ **do**
  $u_1Recoms \leftarrow []$;
  $score[U] \leftarrow \{0\}$; $topics[U] \leftarrow \{\}$; $roles[U] \leftarrow \{\}$; $time[U] \leftarrow \{\}$;
  foreach $u2 \in \{U - u1\}$ **do**
    $time[u_2] \leftarrow findMatchingTime(u_1, u_2, A)$;
    $topics[u_2] \leftarrow findMatchingTopics(u_1, u_2, R)$;
    $roles[u_2] \leftarrow findMatchingRoles(u_1, u_2, R)$;
    $RevRoles \leftarrow findMatchingRoles(u_1, u_2, R)$;
    if $(time[u_2] == NULL)$ $or(topics[u_2] == \emptyset)$ **then**
      $score[u_2] \leftarrow \epsilon$
    else
      $Su_1u_2 \leftarrow Compatibility(u_1, u_2, C, P$ $topics[u_2], roles[u_2], \tau, \alpha, \sigma)$;
      $Su_2u_1 \leftarrow Compatibility(u_2, u_1, C, P$ $topics[u_2], RevRoles, \tau, \alpha, \sigma)$;
      $score[u_2] \leftarrow \frac{2}{Su_1u_2^{-1} + Su_2u_1^{-1}}$
    end
  end
  $topK \leftarrow topKScores(score)$;
  foreach $u \in topK$ **do**
    if *score[u]>0* **then**
      $u1Recoms.append([u_1, u, topics[u], roles[u], time[u], score[u]])$;
    end
  end
  $Recoms.extend(u_1Recoms)$
end
return *Recoms*;

3.5 Simple Example

In this section an example with 5 learners ($L1, \ldots, L5$) is presented to illustrate the calculation of the reciprocal score between pairs of learners. Table 2 includes the availability, topic competence, the role the learner is seeking and the specified role preference scores for each learner. The proposed algorithm is able to provide recommendations for multiple topics with the learner specifying a role preference for each topic (i.e., $R_{U \times L \times Q}$ and $C_{U \times L}$) but in order to create a simple illustrative example, the role preferences and associated competency are only provided for a single topic. The following parameter values are provided to the reciprocal recommender algorithm in this example: the number of recommendations (k) is set to 1 and the minimum desired threshold (τ) for the joint competency function is set to 30. As described in Section 3.3, the joint competency between learners needs to be above τ to ensure that learners with a low joint competency in a topic are not matched.

Table 2: Student preferences used in example

	A		C	Roles			P		
	t_1	t_2	l_1	$q = 1$	$q = 2$	$q = 3$	$q = 1$	$q = 2$	$q = 3$
L1	1	0	80	1	0	0	-80	0	5
L2	1	0	20	0	1	0	0	40	10
L3	1	0	60	1	0	0	-30	0	15
L4	1	1	10	0	1	0	0	60	-5
L5	0	1	60	1	0	0	-10	10	0

Four of the five learners in Table 2 share an explicit preference match for timeslot t_1 but have different role preferences and preference scores for providing and seeking peer learning support for Topic l_1. The algorithm proposed in this paper also provides learners with the ability to set their own competency preferences for each role, which is captured in P. A learner is able to set a competency score for a role preference by providing a competency score differential. A learner such as $L1$, who has a high competency in Topic l_1 can choose to mentor learners with very low competency in a topic. Preferences make the proposed algorithm very flexible especially within an educational context allowing students to express their personal preference in providing and seeking support. In Table 2 $L1$ specifies a negative competency score (i.e. -80 points) indicating a preference to provide learning support to learners whose competency matches the difference between their score and the specified preference value. $L2$ sets a positive score (i.e., +60 points) indicating that they would like to receive learning support from learners with competency greater than their own by the specified value.

Table 3 contains the reciprocal scores returned by the recommendation algorithm. $L2$ and $L4$ are seeking peer learning support and both $L1$ and $L3$ are available in the timeslot and have specified their ability to provide peer learning support. The reciprocal score matching $L1$ and $L4$ is higher than the score between $L3$ and $L4$, because the learner specified preference scores are a better match for $L1$ and $L4$. The reciprocal score matching $L2$ and $L3$ is higher than the score between $L2$ and $L1$, again due to the preference scores specified by the learners. $L5$ is not available in Timeslot t_1 and can only be matched with $L4$; however due to a mismatch between their specified preferences their reciprocal scores is quite low.

Table 3: Reciprocal scores

	L1	L2	L3	L4	L5
L1		61	ϵ	88	ϵ
L2	61		91	ϵ	ϵ
L3	ϵ	91		69	ϵ
L4	88	ϵ	69		23
L5	ϵ	ϵ	ϵ	23	

Table 4: Recommendations

Recommendations
[$L1$, $L4$, [l_1], [q=1], t_1, 88]
[$L4$, $L1$, [l_1], [q=2], t_1, 88]
[$L2$, $L3$, [l_1], [q=2], t_1, 91]
[$L3$, $L2$, [l_1], [q=2], t_1, 91]
[$L5$, $L4$, [l_1], [q=1], t_2, 23]

Table 4 shows the recommendations that will be made by the platform. It is interesting to note that while the reciprocal scores between two users are symmetric, it does not necessarily mean symmetric recommendations will be made (e.g., $L4$ and $L5$). Recall that in this simple example the number of recommendations (k) is constrained to 1.

It is evident from the resulting reciprocal scores that the proposed reciprocal recommendation algorithm is successfully able to incorporate both hard and soft constraints.

4 EVALUATION

4.1 Experimental Environment Setup

Data Sets. The synthetic data algorithm generated matrices for R, C, A, and P by specifying U, L, and T for the system. Additional user inputs included the minimum and maximum number of topics (L_{min}, L_{max}) and time slots (T_{min}, T_{max}) as well as the standard deviation for the competencies of the users σ_c. Within the system, competencies where expressed as a value between zero and 100 inclusive. The competency of each user on a topic was generated using a truncated normal distribution using a randomly generated mean competency and the competency standard deviation input. These topics where then sorted by ascending competency and a random number of roles between the bounds supplied in the input were assigned to create a request. Providing support roles were assigned to the highest competencies, seeking support roles to the lowest competencies and co-studying roles to the topic with the median competency of the available topics. Time availability was determined by randomising the number of available time slots within the bounds provided by the input. In the absence of an empirical basis for the amount by which two users should differ in competency to be successful in a peer learning support partnership, we generated learners' competency difference using a vector p_q, the size of which depends on users' preferences for who they are comfortable peer supporting and who they wish to receive peer support by in terms of competency difference, and assume a Normal distribution with a standard distribution of σ_p for generating values in P.

Evaluation Metrics. The outcome of the system is evaluated using the following criteria:

- **Scalability**: Based on the time taken for running algorithm 1.
- **Reciprocality**: Based on precision of reciprocal recommender systems as described in [45] and Section 4.3.
- **Coverage**: Based on the percentage of users that have been recommended at least once to other users.
- **Quality**: Based on the average joint competency of learners that are recommended to each other across all learners.

parameter settings. In all experiments the parameters are set using the following default values if not otherwise stated: U = 1000, L = 10, T = 10, R=3, k=5, τ=40, α = 10, σ=10, ϵ = 1, L_{min}=1, L_{max}=5, T_{min}=1, T_{max}=10, σ_c=20, p_q=[-20, 40, 0], and σ_p = 5.

4.2 Scalability

There are different parameters that affect the runtime of Algorithm 1, among which the number of learners, U, has the highest impact. The runtime of RiPPLE with respect to different values of U under different settings for k are reported in Figure 5. As expected, the runtime of the algorithm increases as U is increased. The runtime of the algorithm is quadratic, $O(n^2)$, which is also confirmed by the experimental results.

The value of k does have a significant impact on the runtime of the algorithm. This is expected as the for loop adding the k recommendations runs in constant time.

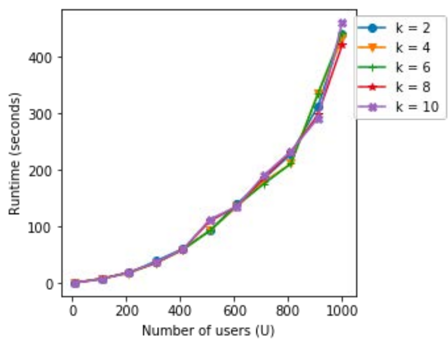


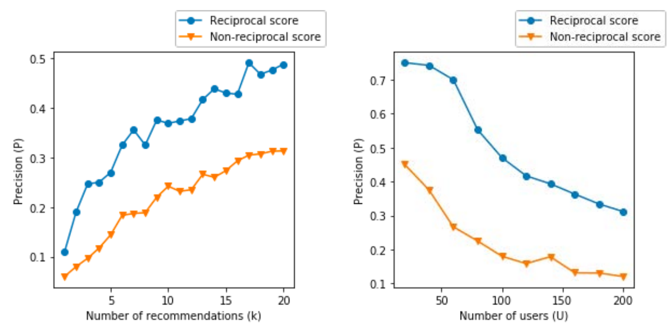
Figure 5: Runtime of Algorithm 1 as the number of learners is increased.

In the current implementation, the recommendation algorithm is executed once by RiPPLE at the end of each week, providing recommendations for the week ahead. As such, the runtime of the algorithm does not have a significant impact on the scalability of the system. In future versions of the platform, our goal is to be able to provide recommendations on demand, where runtime of the algorithm would have a more significant impact on the usability of the system. The use of caching techniques will allow the runtime of the platform to be further improved; however, even with the current setting in a class of 1000 students, the platform is able to provide recommendations to a single user in $\frac{450}{1000} = 0.045$ of a second.

4.3 Reciprocality

In this experiment, we evaluate the effectiveness of the use of our reciprocal score against a baseline non-reciprocal score. For the baseline score we have used the $S_{u_1u_2}$, as described in Section 3.3, which only considers the preference of the learner receiving the recommendation without the condition of reciprocality. For this experiment, we use the precision of reciprocal recommender systems defined by Prabhakar [45] as follows: "learner u_1 is a successful (reciprocal) recommendation (out of the K-total) for learner u_2, if and only if u_1 is also in the top k recommendations of learner u_2". Precision for learner u_1 is obtained by dividing the number of successful recommendations by k, and the precision of the system is defined as the average precision across all users.

Figure 6a compares the precision of the platform for U=200 based on different values of k using the defined reciprocal and non-reciprocal scores. Our results matches the results reported by Prabhakar [45] in that the precision of the reciprocal score is notably higher than the baseline. It is interesting to note that an increase in k increases the fraction of the successful recommendations, which is shown by precision. Figure 6b compares the precision of the platform for k=5 based on different values of U using the defined reciprocal and non-reciprocal scores. Again, the precision score of the reciprocal score far exceeds the non-reciprocal baseline score. In addition, increasing U leads to a lower precision for both scores. This is expected as an increase in U, while k is kept constant, reduces the probability of a successful reciprocal recommendation.

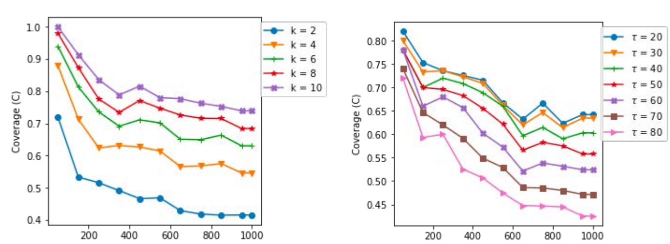


(a) Changing k **(b) Changing U**

Figure 6: Precision of the platform for a reciprocal and a non-reciprocal score.

4.4 Coverage

In this experiment we evaluate the coverage of the reciprocal recommender system. Figure 7a presents the coverage of the platform as U is increased under different settings for k. For smaller values of U, the coverage of the platform is close to 0.9 for most reported values of k. Under much stricter conditions for τ and k and for larger U, coverage remains above 0.4. Recall that coverage is defined as the proportion of users that have been recommended at least once to other users; even under these strict conditions almost all of the learners are still receiving recommendations.



(a) Changing k **(b) Changing τ**

Figure 7: Coverage of the platform as U is increased under different settings for k and τ

4.5 Quality

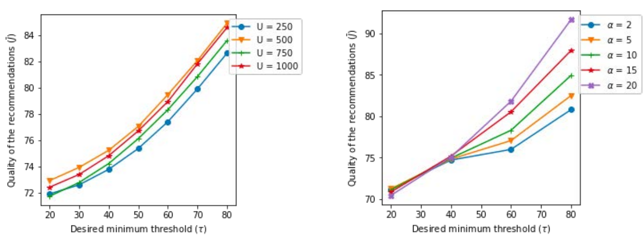
We assess the quality of a pairing between u_1 and u_2, $J_{u_1 u_2}$, based on their average joint competencies across topics that they are matched on using the following formula:

$$J_{u_1 u_2} = \sum_{l \in [l]} \frac{J_{u_1 u_2 l}}{size([l])} \tag{5}$$

Using this definition, we assess the quality of the provided recommendations for the entire cohort as the average quality of the pairings in $Recoms_N$ using the following formula:

$$\bar{J} = \frac{\sum_{(u_1,u_2) \in Recoms_i}^{Recoms_N} J_{u_1 u_2}}{N} \tag{6}$$

Figure 8a shows the impact on $\bar{J}$ with the number of users U and changes to τ, the minimum competency threshold. Figure 8b shows the impact on $\bar{J}$ of α, the leniency parameter. The results are not unexpected. Increasing τ results in a higher $\bar{J}$ independently of the number of users, and relaxing α produces comparatively lower $\bar{J}$ for values of $\tau > 40$.



(a) Changing U **(b) Changing τ**

Figure 8: Approximating the quality of the recommendations as τ is increased under different settings for U and α

5 CONCLUSION AND FUTURE WORK

A novel learning platform called RiPPLE that supports peer learning and reciprocal peer recommendation was introduced. RiPPLE uses learners' topic-level competencies, approximated through students' responses to tagged multiple-choice questions within the platform, alongside their explicit preferences to recommend peers for the purposes of providing or receiving learning support and finding study partners. The defining feature of any recommender system is the user preference model which must reflect the recommendation goals and available data. The formulation of a competency-based user preference model and reciprocal scoring function was detailed, designed with the aim of promoting recommendations that benefit both users.

An experimental validation of RiPPLE used synthetic data sets to determine the scalability and feasibility of the platform before it is introduced to real users. The synthetic data sets evaluated the behaviour of RiPPLE across the range of parameter settings using the key criteria of scalability, reciprocality, coverage and quality. The results of the evaluation were not unexpected. Coverage was heavily dependent on the number of users, number of recommendations per user and threshold settings. Importantly, the precision of recommendations was higher compared to non-reciprocal recommendations for all cohort sizes and variations to the number of recommendations per user.

The absence of anomalous findings in these primary results indicates that RiPPLE can help learners connect with their peers in the manner proposed across a wide range of competency levels and cohort sizes. The platform is planned for use in large introductory courses from three different faculties next semester. Subsequent evaluations will address the most significant limitation of the current study - that the recommendations are not validated using real users who can provide compelling evidence of RiPPLE's capacity to make meaningful recommendations.

RiPPLE is designed to support A/B testing so that parallel-group double-blind randomised experiments may be conducted. Our goal is to validate the platform with a control group that would receive random peer recommendations and an experimental group that would receive targeted peer recommendations using the proposed algorithm. This will allow us to determine whether or not the recommendations lead to measurable gains and also provide insight into the manner in which users select preferences, the conditions upon which they accept or ignore recommendations, how they choose from a list of k recommendations, and otherwise interact with the system.

Variable parameters allow for the recommender to be highly customisable for both administrators (as in setting a minimum joint competency) and for users who can specify their competency preferences according to their perceived needs and/or confidence in providing support. This flexibility was intended to address the key limitations identified by Olakanmi and Vassileva [41] regarding inflexibility and orphaned learners. Empirical studies of the competency differences between users providing and receiving peer support through RiPPLE recommendations, and the relationship of these differences with successful learning, will not only serve to address Olakanmi and Vassileva's other criticisms of self-report evaluation and promoting learning activities ahead of learners' skills, but will also primarily become the basis for determining appropriate default parameter settings in RiPPLE.

The consequences for future experimental work go beyond addressing recognised limitations of peer recommending however. The findings of such investigations have further potential to inform new theories of peer learning and subsequently shape this aspect of technology enhanced learning systems broadly.

REFERENCES

[1] Jane Andrews and Helen Higson. 2008. Graduate employability, soft skills versus hard business knowledge: A European study. *Higher education in Europe* 33, 4 (2008), 411–422.
[2] Employment Australia. Department of Education, Workplace Relations, and Denise Bradley. 2008. *Review of Australian higher education: Discussion paper.* Department of Education, Employment and Workplace Relations.
[3] Roger Bennett. 2002. Employers' demands for personal transferable skills in graduates: A content analysis of 1000 job advertisements and an associated empirical study. *Journal of Vocational Education and training* 54, 4 (2002), 457–476.
[4] Phyllis C Blumenfeld, Ronald W Marx, Elliot Soloway, and Joseph Krajcik. 1996. Learning with peers: From small group cooperation to collaborative communities. *Educational researcher* 25, 8 (1996), 37–39.
[5] Fatiha Bousbahi and Henda Chorfi. 2015. MOOC-Rec: a case based recommender system for MOOCs. *Procedia-Social and Behavioral Sciences* 195 (2015), 1813–1822.
[6] Susan Bull, Jim Greer, Gordon McCalla, Lori Kettel, and Jeff Bowes. 2001. User modelling in i-help: What, why, when and how. In *International Conference on*

User Modeling. Springer, 117–126.
[7] Xiongcai Cai, Michael Bain, Alfred Krzywicki, Wayne Wobcke, Yang Sok Kim, Paul Compton, and Ashesh Mahidadia. 2011. Learning to make social recommendations: a model-based approach. In *International Conference on Advanced Data Mining and Applications*. Springer, 124–137.
[8] S.C. Cazella, E.B. Reategui, and P.A. Behar. 2010. Recommendation of Learning Objects Applying Collaborative Filtering and Competencies. In *Key Competencies in the Knowledge Society*. Springer, 35–43.
[9] Lin Chen and Richi Nayak. 2013. A Reciprocal Collaborative Method Using Relevance Feedback and Feature Importance. In *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) - Volume 01 (WI-IAT '13)*. IEEE Computer Society, Washington, DC, USA, 133–138. https://doi.org/10.1109/WI-IAT.2013.20
[10] Gráinne Conole. 2010. Review of Pedagogical Models and their use in e-learning. *Milton Keynes: Open University* (2010).
[11] Geoffrey Crisp, Edward Palmer, Deborah Turnbull, Ted Nettelbeck, Lynn Ward, Amanda LeCouteur, Aspa Sarris, Peter Strelan, and Luke Schneider. 2009. First year student expectations: Results from a university-wide student survey. *Journal of University Teaching and Learning Practice* 6, 1 (2009), 11–26.
[12] Shane Dawson. 2006. A study of the relationship between student communication interaction and sense of community. *The Internet and Higher Education* 9, 3 (2006), 153–162.
[13] Rouxelle De Villiers. 2010. The incorporation of soft skills into accounting curricula: preparing accounting graduates for their unpredictable futures. *Meditari Accountancy Research* 18, 2 (2010), 1–22.
[14] Seth Deepa and Manisha Seth. 2013. Do Soft Skills Matter?-Implications for Educators Based on Recruiters' Perspective. *IUP Journal of Soft Skills* 7, 1 (2013), 7.
[15] Hendrik Drachsler, Katrien Verbert, Olga C Santos, and Nikos Manouselis. 2015. Panorama of recommender systems to support learning. In *Recommender systems handbook*. Springer, 421–451.
[16] Mojisola Erdt, Alejandro Fernández, and Christoph Rensing. 2015. Evaluating Recommender Systems for Technology Enhanced Learning: A Quantitative Survey. *IEEE Transactions on Learning Technologies* 8, 4 (2015), 326–344.
[17] Nancy Falchikov. 2001. *Learning together: Peer tutoring in higher education*. Psychology Press.
[18] David J Finch, Leah K Hamilton, Riley Baldwin, and Mark Zehner. 2013. An exploratory study of factors affecting undergraduate employability. *Education+ Training* 55, 7 (2013), 681–704.
[19] Isaac R Galatzer-Levy, Charles L Burton, and George A Bonanno. 2012. Coping flexibility, potentially traumatic life events, and resilience: A prospective study of college student adjustment. *Journal of Social and Clinical Psychology* 31, 6 (2012), 542–567.
[20] Jim Greer, Gordon McCalla, John Cooke, Jason Collins, Vive Kumar, Andrew Bishop, and Julita Vassileva. 1998. The intelligent helpdesk: Supporting peer-help in a university course. In *Intelligent tutoring systems*. Springer, 494–503.
[21] Ido Guy. 2015. Social recommender systems. In *Recommender Systems Handbook*. Springer, 511–543.
[22] Claudia M Heijde and Beatrice IJM Van Der Heijden. 2006. A competence-based and multidimensional operationalization and measurement of employability. *Human resource management* 45, 3 (2006), 449–476.
[23] Jeremy Herbert, Elliot Smith, Carl Reidsema, and Lydia Kavanagh. 2013. Helping students find answers: Algorithmic interpretation of student questions. In *AAEE 2013: 24th Annual Conference of the Australasian Association for Engineering Education*. Griffith School of Engineering, Griffith University, 1–6.
[24] Wenxing Hong, Siting Zheng, Huan Wang, and Jianchao Shi. 2013. A job recommender system based on user clustering. *Journal of Computers* 8, 8 (2013), 1960–1967.
[25] Zoran Jeremić, Jelena Jovanović, and Dragan Gašević. 2009. Semantic web technologies for the integration of learning tools and context-aware educational services. *The Semantic Web-ISWC 2009* (2009), 860–875.
[26] Lydia Kavanagh, David Neil, and John Cokley. 2011. Developing and disseminating team skills capacities using interactive online tools for team formation, learning, assessment and mentoring: Final report 2011. (2011).
[27] Hassan Khosravi. 2017. Recommendation in Personalised Peer-Learning Environments. *arXiv preprint arXiv:1712.03077* (2017).
[28] Hassan Khosravi, Kendra Cooper, and Kirsty Kitto. 2017. RiPLE: Recommendation in Peer-Learning Environments Based on Knowledge Gaps and Interests. *JEDM-Journal of Educational Data Mining* 9, 1 (2017), 42–67.
[29] Sangeetha Kutty, Richi Nayak, and Lin Chen. 2014. A people-to-people matching system using graph mining techniques. *World Wide Web* 17, 3 (2014), 311–349.
[30] Patrick C Kyllonen. 2013. Soft skills for the workplace. *Change: The Magazine of Higher Learning* 45, 6 (2013), 16–23.
[31] Daniel Lemire, Harold Boley, Sean McGrath, and Marcel Ball. 2005. Collaborative filtering and inference rules for context-aware learning object recommendation. *Interactive Technology and Smart Education* 2, 3 (2005), 179–188.
[32] Matthew Lipman. 2003. *Thinking in education*. Cambridge University Press.
[33] Eleni Mangina and John Kilbride. 2008. Evaluation of keyphrase extraction algorithm and tiling process for a document/resource recommender within e-learning environments. *Computers & Education* 50, 3 (2008), 807–820.
[34] Nikos Manouselis, Hendrik Drachsler, Riina Vuorikari, Hans Hummel, and Rob Koper. 2011. Recommender systems in technology enhanced learning. In *Recommender systems handbook*. Springer, 387–415.
[35] Nikos Manouselis and Katrien Verbert. 2013. Layered evaluation of multi-criteria collaborative filtering for scientific paper recommendation. *Procedia Computer Science* 18 (2013), 1189–1197.
[36] Nikos Manouselis, Riina Vuorikari, and Frans Van Assche. 2007. Simulated analysis of MAUT collaborative filtering for learning object recommendation. In *Proceedings of the 1st Workshop on Social Information Retrieval for Technology Enhanced Learning*. 27–35.
[37] Terry Mayes and Sara De Freitas. 2004. Review of e-learning theories, frameworks and models. JISC e-learning models study report. (2004).
[38] Joanne M McInnerney and Tim S Roberts. 2004. Online learning: Social interaction and the creation of a sense of community. *Educational Technology & Society* 7, 3 (2004), 73–81.
[39] Geana W Mitchell, Leane B Skinner, and Bonnie J White. 2010. Essential soft skills for success in the twenty-first century workforce as perceived by business educators. *The Journal of Research in Business Education* 52, 1 (2010), 43.
[40] Julián Moreno, Demetrio A Ovalle, and Rosa M Vicari. 2012. A genetic algorithm approach for group formation in collaborative learning considering multiple student characteristics. *Computers & Education* 58, 1 (2012), 560–569.
[41] Oluwabunmi Adewoyin Olakanmi and Julita Vassileva. 2017. Group Matching for Peer Mentorship in Small Groups. In *CYTED-RITOS International Workshop on Groupware*. Springer, 65–80.
[42] Mark Palmer, Paula O'Kane, and Martin Owens. 2009. Betwixt spaces: student accounts of turning point experiences in the first-year transition. *Studies in Higher education* 34, 1 (2009), 37–54.
[43] Luiz Pizzato, Tomasz Rej, Joshua Akehurst, Irena Koprinska, Kalina Yacef, and Judy Kay. 2013. Recommending people to people: the nature of reciprocal recommenders with a case study in online dating. *User Modeling and User-Adapted Interaction* 23, 5 (2013), 447–488.
[44] Luiz Pizzato, Tomek Rej, Thomas Chung, Kalina Yacef, Irena Koprinska, and Judy Kay. 2010. Reciprocal recommenders. In *8th Workshop on Intelligent Techniques for Web Personalization and Recommender Systems, UMAP*.
[45] Sankalp Prabhakar, Gerasimos Spanakis, and Osmar Zaiane. 2017. *Reciprocal Recommender System for Learners in Massive Open Online Courses (MOOCs)*. Springer International Publishing, Cham, 157–167. https://doi.org/10.1007/978-3-319-66733-1_17
[46] Elizabeth Rainsbury, David Leslie Hodges, Noel Burchell, and Mark C Lay. 2002. Ranking workplace competencies: Student and graduate perceptions. (2002).
[47] Carl A Reidsema, Lydia Kavanagh, Emmi Ollila, Stephanie Otte, and Julie E McCredden. 2016. Exploring the quality and effectiveness of online, focused peer discussions using the MOOCchat tool. In *27th Australasian Association for Engineering Education Conference*. AAEE.
[48] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. *Introduction to Recommender Systems Handbook*. Springer US, Boston, MA, 1–35. https://doi.org/10.1007/978-0-387-85820-3_1
[49] Elliot Smith, Jeremy Herbert, Lydia Kavanagh, and Carl Reidsema. 2013. The effects of gamification on student learning through the use of reputation and rewards within community moderated discussion boards. In *AAEE 2013: 24th Annual Conference of the Australasian Association for Engineering Education*. Griffith School of Engineering, Griffith University, 1–9.
[50] Nguyen Thai-Nghe, Lucas Drumond, Tomáš Horváth, Artus Krohn-Grimberghe, Alexandros Nanopoulos, and Lars Schmidt-Thieme. 2011. Factorization techniques for predicting student performance. *Educational Recommender Systems and Technologies: Practices and Challenges* (2011), 129–153.
[51] Beatrice Van Der Heijden, Jo Boon, Marcel Van der Klink, and Ely Meijs. 2009. 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 (2009), 19–37.
[52] Katrien Verbert, Hendrik Drachsler, Nikos Manouselis, Martin Wolpers, Riina Vuorikari, and Erik Duval. 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.
[53] Katrien Verbert, Nikos Manouselis, Xavier Ochoa, Martin Wolpers, Hendrik Drachsler, Ivana Bosnic, and Erik Duval. 2012. Context-aware recommender systems for learning: a survey and future challenges. *Learning Technologies, IEEE Transactions on* 5, 4 (2012), 318–335.
[54] Paula Wilcox, Sandra Winn, and Marylynn Fyvie-Gauld. 2005. âĂŸIt was nothing to do with the university, it was just the peopleâĂŹ: the role of social support in the first-year experience of higher education. *Studies in higher education* 30, 6 (2005), 707–722.
[55] Chun-Mei Zhao and George D Kuh. 2004. Adding value: Learning communities and student engagement. *Research in higher education* 45, 2 (2004), 115–138.