The Timeliness Deviation: A novel Approach to Evaluate Educational Recommender Systems for Closed-Courses

Christopher Krauss
Fraunhofer FOKUS
10589 Berlin, Germany
christopher.krauss@
fokus.fraunhofer.de

Agathe Merceron
Beuth University of Applied Sciences
13353 Berlin, Germany
merceron@beuth-hochschule.de

Stefan Arbanowski
Fraunhofer FOKUS
10589 Berlin, Germany
stefan.arbanowski@
fokus.fraunhofer.de

ABSTRACT

The decision on what item to learn next in a course can be supported by a recommender system (RS), which aims at making the learning process more efficient and effective. However, learners and learning activities frequently change over time. The question is: how are timely appropriate recommendations of learning resources actually evaluated and how can they be compared?

Researchers have found that, in addition to a standardized dataset definition, there is also a lack of standardized definitions of evaluation procedures for RS in the area of Technology Enhanced Learning. This paper argues that, in a closed-course setting, a time-dependent split into the training set and test set is more appropriate than the usual cross-validation to evaluate the Top-N recommended learning resources at various points in time. Moreover, a new measure is introduced to determine the timeliness deviation between the point in time of an item recommendation and the point in time of the actual access by the user. Different recommender algorithms, including two novel ones, are evaluated with the time-dependent evaluation framework and the results, as well as the appropriateness of the framework, are discussed.

CCS CONCEPTS

• **Information systems** → **Evaluation of retrieval results**; *Recommender systems*; • **Applied computing** → *Computer-assisted instruction*;

KEYWORDS

Time-Dependent Evaluation Framework, Educational Recommender Systems, Timeliness Deviation

1 INTRODUCTION

Learning, especially Self-Regulated Learning, requires responsibility on the part of the learner which can be assisted by educational recommender systems (RSs). These Learning Analytics predictors aim at optimizing learning by making it more efficient and more effective. Thereby, "efficiency" describes the way to achieve a personal goal. In terms of learning in a closed-corpus setting like a course, a higher efficiency means optimizing the process, saving effort and time to reach the course goal. "Effectiveness", in turn, directly concerns the result achieved. A higher effectiveness means to reach, e.g., a better mark in the exam or longer lasting knowledge. Both can be improved through recommender systems that present appropriate learning resources in the given situation to the user.

Drachsler et al. note that learners and learning activities frequently change over time [9] which also directly affects the learner's goal (also see [10]). That is one reason why educational recommendations depend more on time information than traditional recommendations do. In particular, certain recommendations become obsolete after a short time span, e.g., when the recommended learning resource has been studied by the students or the next lecture focuses on a different topic. This leads to the requirement to recommend certain topics only at relevant times, which must be taken into account when recommending appropriate resources. Some recommender systems, namely the Time-Aware Recommender Systems (TARSs), include time attributes in their algorithms, such as "time of the day, day of the week, and season of the year" [3]. This improves recommendations from traditional domains [3] as well as from Technology Enhanced Learning (TEL) [13]. However, the question arises: how are *appropriate* recommendations of learning resources actually evaluated and how can they be compared?

In general, evaluation is "the identification, clarification, and application of defensible criteria to determine an evaluation object's value, quality, utility, effectiveness, or significance in relation to those criteria" [33]. Said and Bellogin [24] evaluated common evaluation frameworks and protocols for general recommender systems regardless of the particular application area. They conclude that the performance of an algorithm highly depends on the evaluation framework and, thus, cannot be compared to the performance of the same algorithm in another evaluation setting. In their experiments, the results differ by up to 10% depending on the evaluation framework. Moreover, Said and Bellogin note a lack of "rules or standards on how to evaluate a recommendation algorithm" [24]. Campos et al. noticed that the existence of a huge variety of evaluation approaches for general recommender systems results in "an increasing impediment to fairly compare results and conclusions reported in different studies" [3]. Moreover, "variations in user interfaces", "data selection" and "situational and personal characteristics of users" lead to differences between qualitative and quantitative evaluations [3].

These circumstances are even worse for RS in Technology Enhanced Learning (TEL). Chatti et al. argue that "an implementation of different recommendation algorithms within a single recommender system to compare against each other is missing in the TEL

LAK19, March 4–8, 2019, Tempe, AZ, USA

ACM ISBN 978-1-4503-6256-6/19/03…$15.00
https://doi.org/10.1145/3303772.3303774

recommenders literature" [4]. Thereby, "further evaluation procedures that complement the technical evaluation approaches" for the comparison of educational recommender systems are needed to produce reliable and comparable results [21]. This is why the underlying evaluation framework must remain the same in each experiment when it should produce comparable results.

This paper defines a time-dependent evaluation framework to investigate the precision of the Top-N recommended learning resources at various points in time in closed-course settings. Moreover, a new measure called the timeliness deviation is introduced to investigate the gap between the time when an item is recommended and the time when it is actually accessed by the user.

The remainder of the paper is structured as follows: Section 2 introduces related work on offline simulations, evaluation frameworks, data splitting, and cross-validation procedures as well as common measurement values. The next section describes a time-dependent evaluation framework which allows to appropriately analyze the algorithms' limited performances in educational courses. A novel timeliness deviation measure is introduced in Section 4. Then, five recommender systems are shortly introduced, including two novel ones, and experiments based on the time-dependent framework which includes the common precision measure, as well as the novel timeliness deviation, are described. The findings of the experiments as well as the benefits and limitations of this approach are discussed in Section 6. The paper concludes with a summary and an outlook.

2 RELATED WORK

Evaluations of educational recommender systems can be either performed online, which means directly in a real course or offline in a simulation of a course. This paper focuses on offline evaluations that utilize either simulated data or past real-world data in a simulated environment [3]. This section deals with the main criteria to evaluate recommender systems. It first discusses the necessity of real-world data, then presents general evaluation frameworks, describes common procedures for data splitting and finally introduces important measurement values.

2.1 Real-World Activity Data

Since data is not always available and it is cost intensive to conduct experiments in real learning environments, many researchers simulate their users based on machine learning technologies or self-developed student models [20, 25, 27]. These student simulators should partially overcome the problem of massive testing with real students [27]. However, the weakness of simulated student profiles comes from the unpredictability of real human behavior. If today researchers were able to model the complex learning patterns of all students, the key task of a recommender system would have been solved. Also, Campos et al. note, that "the majority of past work on [...] recommender systems has been focused on offline evaluation protocols" [3] including those conducted with past but real data.

That is why we argue for experimenting with activity data that have been collected in real-world courses. These evaluations are not performed online in a live course setting, but rather offline using past but real-world data. Thus, the evaluated recommender algorithms do not influence the future behaviors of the learners as a live evaluation would do. It rather aims at forecasting the usage patterns that are already present in the data. This mixture is very common for evaluations of general recommender systems [3, 22] (and even for the Netflix prize [2]) and has been also applied for TEL recommender systems [10, 21, 29].

The practice of using datasets from other domains than education, and in particular from the movie domain, which is common practice, "lacks the necessary validity for proving recommendation algorithms for TEL" [29]. Reliable datasets need to "capture learner interactions in real settings" and should give the opportunity for "verification, repeatability, and comparisons of results" [29]. In an offline evaluation with real data, the recommendation tasks should directly support the tasks the users would perform anyway in the system [12]. On the other hand, researchers should "adequately define the reference variables against which the adaptivity of the system will be evaluated" [21]. Thus, it would be inappropriate to use data from a different domain or from services that do not focus on the same use case. Verbert et al. [29] argue that "the continuation of additional small-scale experiments with a limited amount of learners that rate the relevance of suggested resources only adds little contributions to an evidence driven knowledge base".

Though real-world data from the educational domain should be used, researchers point out that there is a lack of open, shareable datasets which incorporate contextual learning data and allow for a comparison of the results with common measures [8, 21, 30]. Besides the missing definition of any standardized formats, there is an issue regarding privacy and legal concerns, which differs on a country by country and institution by institution basis.

2.2 Evaluation Frameworks

Researchers have noted that besides the lack of open academic datasets, there is also a lack of standardized definitions relating to evaluation procedures for recommender systems in Technology Enhanced Learning [8, 9, 21]. Those authors suggest approaches but also comment that they must be further researched.

Weibelzahl [32] introduced a framework for a four-tiered evaluation procedure consisting of the *evaluation of the dataset*, *evaluation of the inference mechanism*, *evaluation of the adaption decision*, and *evaluation of the total interaction*. Manouselis et al. [21] abstracted this to a multi-layered evaluation approach for RSs in TEL which can also be reduced to only two layers: *the accuracy of the model* and *the effectiveness of the changes made at the interface*. The first layer corresponds to quantitative evaluations, relating to measurements of the algorithm's outputs, the second to qualitative assessments regarding user perceptions. Similar to other RS evaluation procedures, this work has a focus on the quantitative aspects.

According to Said and Bellogin [24], the four most important evaluation dimensions are (1) *the dataset*, (2) *the data splitting*, (3) *the evaluation strategies* and (4) *the metrics* (here called measurement values). This work builds on these four dimensions and follows additionally the approach of Campos et al. [3] who suggest that researchers should describe the following criteria when presenting evaluations: qualitative and quantitative details about the dataset and its composition, approach of the training-test set splitting procedure and cross-validation approach, scoring order of the items as well as the description of which items are considered as the Top-N

target items and which items are considered as relevant for each user in a Top-N recommendation task.

2.3 Data Splitting & Cross-Validation

When performing an offline evaluation with historical data, the whole dataset must be split: To guarantee an objective prediction of data, the training dataset Tr must be separated from the test dataset Te (cf. [12] [3]):

$$Tr \cap Te = \emptyset. \quad (1)$$

The training data is used as input for the algorithms, which, in turn, develop a model based on the patterns in the training data. The test data, however, is unknown for the training phase and only utilized to judge the performance of the algorithm with unknown data.

It is common sense that the split process happens randomly. One method to train and evaluate algorithms is the n-fold cross-validation. This cross-validation type is repeated n times (e.g., 10-times for $n = 10$). During every iteration the whole dataset is split into 90% of training data and 10% of evaluation data (cf. [12] [1]).

Campos et al. [3] suggested an alternative evaluation procedure which is appropriate for systems whose recommendations depend on time. Recommendations that are calculated and presented in a closed-corpus environment for a limited time window only, such as in a half-year course, depend more on temporal effects than open-corpus recommendations without time limitations. Thus, evaluation frameworks for Time-Aware Recommender Systems seem to be relevant in such a context. While a time-based evaluation for TEL recommender systems was also mentioned in a survey by Erdt et al. [10], it is not clear whether this approach has been used so far. Studies on TEL recommender systems are just based, if at all, on the standard n-fold cross-validation setting until now (e.g., see [21, 29]).

2.4 Measurement Values

A critical question regarding evaluation is: how to measure "appropriate" or "good" recommendations. Campos et al. [3] pointed out that there is no definition of what constitutes a "good" recommendation, but "a commonly used approach is to establish the quality (goodness) of recommendations by computing different measures that assess various desired characteristics of an RS output". Following the same idea, Manouselis et al. [21] introduces four high-level measures to define success criteria of recommender systems in TEL:

(1) *Effectiveness* describes the percentage of consumed items during a learning phase (here a course).
(2) *Efficiency* indicates the time needed by the user to reach the learning goal.
(3) *Satisfaction* is a subjective measure that must be assessed by discussion with users.
(4) *Drop-out rates* represent the percentage of users who stop participating in the learning setting and thus do not reach the course goals.

Erdt et al. [10] classified similar measures into the groups of *recommender system Performance*, *User-Centric Effects* and *Effects on Learning*. According to them, popular performance measures in offline experiments are the Mean Absolute Error, Root Mean Square Error, precision, recall and f-score.

Moreover, Manouselis et al. [21] incorporated some further measures from Social Network Analysis, such as Variety, Centrality, Closeness, and Cohesion, as they seem also to be valid for learning networks. Due to the course setting of the collected datasets and by following the considerations of Rada [23], the evaluations in this paper have a special focus on efficiency and effectiveness. According to Bellogin et al. [1], each measure itself is insufficient for a fair comparison of different approaches. For instance, "putting more relevant items in the top-N is more important for real recommendation effectiveness than being accurate with predicted rating values" [1].

The effectiveness of a recommender system mostly refers to its prediction accuracy [12]. Thereby, measurements focus on the accuracy of the predicted relevance score – for instance through a value often presented as the *error* [26] or through the *precision* of the Top-N list [7]. Both approaches are introduced in the following.

Error Measurements to determine the prediction errors of the underlying relevance score, such as the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE) are commonly applied in the RS domain [3], [12]. Thereby, the range of the error first and foremost depends on the algorithm's score used. In Technology Enhanced Learning, an RS score can be a traditional rating, a predicted numerical value representing the knowledge level, the number of item accesses or even a Boolean value indicating whether an item has been consumed or not.

While error measures are appropriate to compare deviations between predictions and the actually given relevance scores, they can only be applied for the same type of scoring approach. For instance, an error for a rating-based algorithm (from one to five stars) should not be compared to the error of a knowledge-level-based algorithm (with knowledge levels given in percent). Due to the different meanings and ranges of the relevance scores, the resulting errors of the algorithms differ in their meaning, as well. Moreover, errors do not reveal anything about the appropriateness of the resulting recommendations for a specific learner nor do they allow for a comparison of different algorithms that are based on different scoring values. Cremonesi et al. [6] note that improvements in the error values often do not translate into accuracy improvements. Taking these points into consideration, the following measures might fit better when comparing algorithms in the areas of Time-Aware Recommender Systems and Technology Enhanced Learning.

Ranking Precision values are introduced in the following. Campos et al. [3] argue that the measure *ranking precision* is more appropriate for recommender system tasks than error measures. Ranking precision values represent the coverage of relevant recommendations within the presented Top-N list which are typically given as precision or recall.

Del Olmo and Gaudioso [7] introduce a confusion matrix to explain the possible states which a recommendation might have – see Table 1. The matrix shows the 4 item state categories: they can either be recommended (a & b) or not (c & d) and at the same time be relevant (a & c) or not (b & d). Here, the term "relevant" comes from the Information Retrieval domain: each learning resource that has been consumed by the user (e.g., clicked, watched, answered) is considered as relevant. As the TEL RS aims at also predicting the future item consumption, a "relevant" and "recommended" item is an item that has been accessed by the learner after it has been

Table 1: Confusion matrix for item classification

	Relevant	Non-relevant
Recommended	a	b
Non-recommended	c	d

recommended. Thus, a relevant item for user u is an item that has been accessed by the same user in the test set.

Based on the confusion matrix, *precision* is the portion of recommended relevant items in the set of all recommendations (cf. [1, 7, 11, 31]):

$$precision = \frac{a}{a+b}. \quad (2)$$

Recall, in turn, represents the share of recommended relevant items in the set of all relevant items (cf. [7, 11, 31]):

$$recall = \frac{a}{a+c}. \quad (3)$$

In order to express both values with a single measure, the *F-score* (also known as the F1-measure or F1) is introduced, which represents a harmonic mean of both (cf. [7, 11, 31]):

$$Fscore = \frac{2 * precision * recall}{precision + recall}. \quad (4)$$

Nevertheless, a separate analysis of the precision and recall values allows for a more differentiated interpretation of the results. That is why the F-score plays a minor role in the proposed evaluation framework.

Additional RS Measurement Values have recently been introduced which are not focusing on accuracy or ranking precision tasks: among others novelty [5, 12, 14], diversity [5], sensitivity [31], and specificity [31] as well as serendipity [12]. While they are not utilized in the evaluation framework of the present work, they could be considered in a future work. The new proposed measurement value, however, extends this list of measures. It focuses on the information on how much time it takes on average for an item to be accessed after it has been first recommended.

3 TIME-DEPENDENT CROSS-VALIDATION

Again, evaluation results are only comparable when applying the same methodological framework to all objects of investigations. This is also reflected by the work of Said and Bellogin [24], who identified discrepancies in the determined measures due to the following evaluation dimensions: dataset, data splitting, evaluation strategy, and metrics [24]. Each of the four evaluation dimensions can be seen as a variable, where only one variable is allowed to be changed within a reliable evaluation setting.

When, for example, the analyzed datasets are collected in a different educational context with other context features, an evaluation must apply the same data splitting approach, the same evaluation strategies (in terms of algorithms) and the same measures in order to produce reliable results. Moreover, the overall recommendation goal of the algorithms must be explained. For instance, how does a forecast of a future item's rating support learners in their learning process? That means, for instance, when a dataset does not comprise rating data, it should not be compared to other datasets based only on ratings.

The evaluation of activity data in courses shows an additional restriction: To simulate real-world behavior, the split of the training set Tr and the test set Te must not be entirely random but should depend on a particular point in time – which might be chosen randomly in a given time interval [3]. This second restriction makes it unfair to compare the recommender systems that have to produce time-sensitive recommendations with other recommender systems. Currently, traditional recommender systems are mostly evaluated without taking into account time information for splitting (e.g., by using the n-fold cross-validation).

Therefore, Campos et al. [3] suggest different specialized validation approaches for recommender systems with time-sensitive recommendations. Two definitions seem to be appropriate for the evaluation in closed-course settings. The *increasing-time window* approach splits the whole dataset Tr and Te according to a variable time threshold $t_{threshold}$ so that all data in Tr are older than $t_{threshold}$ and all data in Te are younger. Of course, the time threshold needs to be set within a reasonable interval, e.g., for a course, between the start of the course $t_{CouseStart}$ and the end of the course $t_{CourseEnd}$, so that Tr and Te are not empty – see Figure 1a.

The second approach, the *fixed time-window* cross-validation, works in a similar manner to the *increasing time-window* split, but uses a fixed time *int* representing the interval for both the training dataset and the test dataset. In an educational course, the threshold $t_{threshold}$ is still variable, but the data in Tr are restricted. The time of each training data point t_{Tr} must be in the range $t_{threshold} - int < t_{Tr} < t_{threshold}$ and the time of each test data point t_{Te} must be in $t_{threshold} < t_{Te} < t_{threshold} + int$ (see Figure 1b).

Yi et al. [34] conducted an implicit fixed time-window cross-validation for a search engine evaluation task (without stating it as such). Thereby, the authors analyzed common measures for different sizes of the time window: one month, one week and one day. Interestingly, some measures (e.g., mean absolute precision) were 40 % better for the weekly time windows compared with the monthly time window. The weekly and daily settings, in contrast, showed almost similar results. In conclusion, it is clear that the splitting approach, as well as the size of the fixed time-window interval, must be well-considered as they influence the evaluation results.

It is important to notice that a time-window evaluation is not comparable to a standard cross-validation evaluation as the latter does not reflect temporal effects. The cold start problem, for instance, could not be analyzed with the standard n-fold cross-validation. Thus, in our experiments, the precision results of time-window cross-validations were always below the results of the standard procedure. However, the introduced time-dependent evaluation framework allows for a better understanding of temporal aspects in the data.

4 THE TIMELINESS DEVIATION

A major aspect of recommender systems in closed-course settings is to present efficient recommendations at a reasonable time. This means that recommendations should respect the needs of the learners in a timely fashion – supporting the decision process with

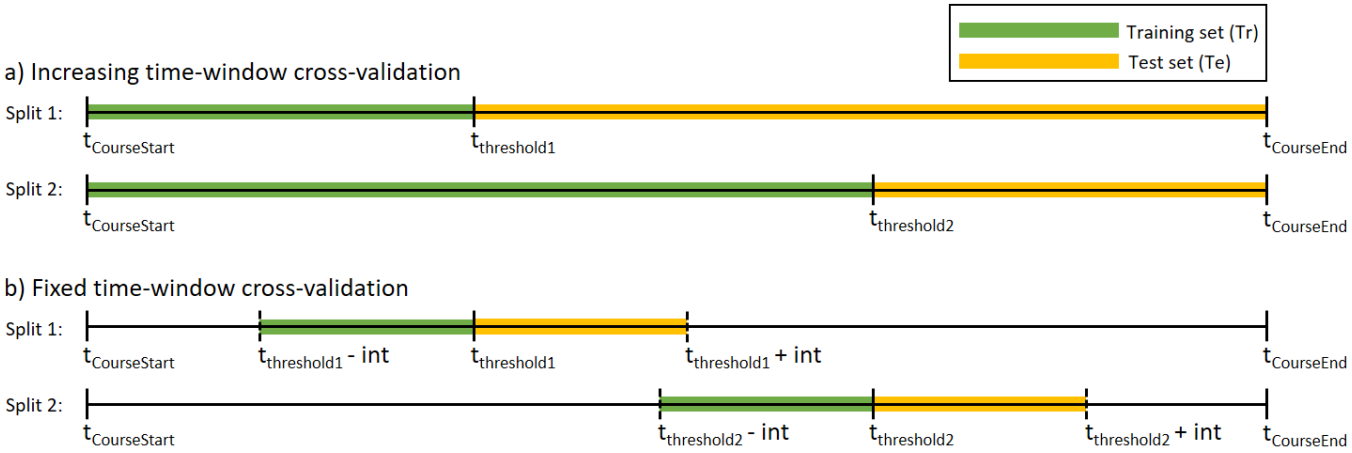

Figure 1: Examples of (a) an increasing time-window cross-validation and (b) a fixed time-window cross-validation

appropriate recommendations for the given situation. In a time-dependent evaluation setting where recommendations should be consumed after they were recommended, the precision value only indicates how many recommended items are relevant to the user.

For the precision value, it does not matter if an item is relevant directly after it has been recommended or only at the end of the course. While effectiveness can be reported with precision and recall, efficient TEL recommendations correspond to the aspect of the timeliness. That is why this work introduces a novel time-dependent evaluation measure: the *timeliness deviation* which borrows from established concepts like RMSE and MAE. The basic idea is that the new timeliness measure indicates how long it takes between the presentation of a recommended item and the time at which the user accesses this item.

4.1 The Mean Absolute Timeliness Deviation

The Mean Absolute Timeliness Deviation (MATD), short *timeliness*, indicates the mean absolute elapsed time for all existing recommendation consumption value pairs $< tr, tc_i >$ of the presented Top-N list. Thereby, only item category a of the confusion matrix which represents all recommended relevant items (see Table 1) is considered for the analysis of $< tr, tc_i >$:

$$MATD = \frac{\sum_{i=1}^{K} tc_i - tr}{K}, \tag{5}$$

where tr represents the point in time of the recommendation of item i presented to a user u and tc_i represents the point of time of the next consumption of item i by the same user u. K is the cardinality of the set of recommended and relevant items. Because of the required time-dependent cross-validation setting, which splits the prediction and test datasets by a time threshold $t_{threshold}$, tr must occur before tc_i ($tc_i \in Te$) – see the *MATD* in Figure 2 for an example.

All time values must be in the same time unit and refer to the same relative point in time (e.g., as a Unix Timestamp in seconds since January 1, 1970). Information on the time unit must be given alongside the timeliness measure. This allows researchers to better compare different timeliness deviations by converting the given time unit appropriately. In a course setting with course start $t_{CourseStart}$ and course end $t_{CourseEnd}$ points, the following definition applies:

$$t_{CourseStart} < tr \leq t_{threshold} < tc_i \leq t_{CourseEnd}. \tag{6}$$

If an item has not been consumed after being recommended, it must not be considered for this calculation as $tc_i - tr$ is then undefined. The number K reduces in this case to the amount of existing recommendation–consumption value pairs $< tr, tc_i >$. Formally, an item i is only considered if it has been recommended and has been consumed after its recommendation. $r(u, i, tr)$ is a binary function that returns true if user u is given a recommendation for item i at point in time tr. $c(u, i, tc_i)$ is a binary function that returns true if user u consumed item i at time tc_i. The set of recommended and relevant items has a cardinality of K and is defined as:

$$\{i \mid r(u, i, tr) \wedge c(u, j, tc_i) \wedge (tr < tc_i)\}. \tag{7}$$

If no recommendation of the Top-N list is relevant, the *MATD* value should not be considered for further averaging, e.g., for all Top-N lists of all users. However, the share of the non-relevant recommendations within the Top-N item list (that is neglected by the timeliness value) is indicated by the precision value, defined above. In this case, the precision would be 0. This is why a timeliness measure should always be presented in combination with the precision.

Similar to other accuracy measurements, single MATD values can be combined, as the mean average, to obtain more general results. It might represent the timeliness of all Top-N recommendations for one user, the timeliness of the recommendations of all users at a specific point in time or even a total timeliness for all Top-N lists of all users over the entire period considered.

4.2 The Cleaned Timeliness Deviation

Practical experiments with real-world data show that the composition of the dataset can also influence the timeliness value. In the following example, the dataset comprises only three users where two are of particular interest (see Figure 2). User 1 logs into the course at the very beginning of the analyzed period (e.g., during

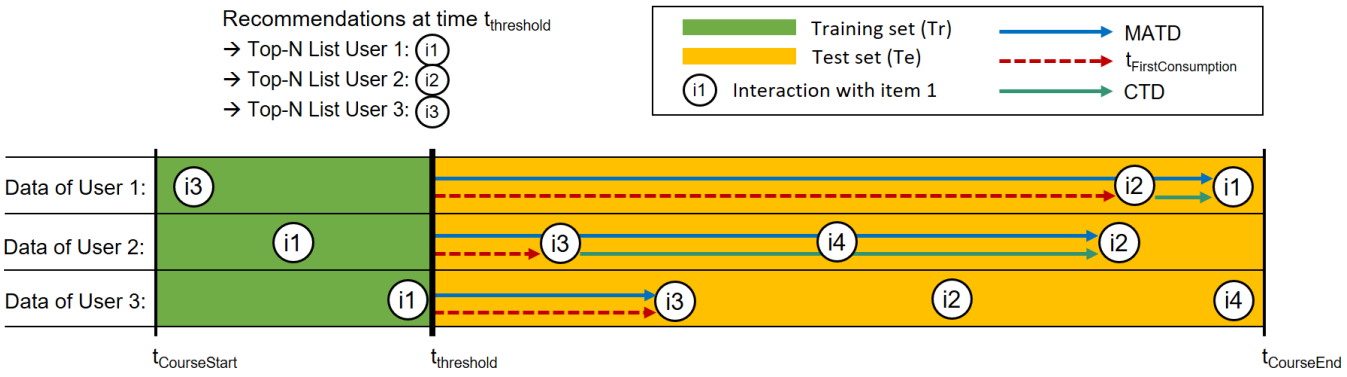

Figure 2: Examples of the MATD and CTD in an increasing time-window cross-validation with $tr = t_{threshold}$. The red dotted-line represents the time-span until a user is first active after the recommendation. The MATD measure represents the time between consumption of the recommended item and recommendation, while the CTD measure represents the time between consumption of the recommended item and first action after the recommendation.

week one) and at the very end (e.g., week 4). User 3, in contrast, logs into the course one day per week – for example, every Thursday. Two issues arise:

First issue: In a time-dependent cross-validation with dataset splitting on a per week basis, the chosen week for the splitting threshold affects the timeliness value. If the data are split every Friday, the minimum timeliness for user 3 would be at least one week until the next item consumption (from Friday until Thursday). If, in contrast, the chosen splitting day is a Wednesday, the minimum timeliness would be only one day (from Wednesday until Thursday). Thus, the chosen splitting threshold would have a huge impact on the timeliness value.

Second issue: When user 1 (who accessed items only at the beginning and the end of the evaluation) and user 3 (who accessed items on a weekly basis) are compared, their timeliness values would differ dramatically. That means that the timeliness deviation of user 1 in week two would be three weeks. However, in reality, when this user was offline the user would not obtain any recommendations before the next use of the system (here in week 4). Summing up, the distribution of consumption data over time affects the evaluation.

Taking both of these extreme cases into account, it makes sense to subtract the next point in time tc_f when the user has consumed any item after the recommendation from the point in time of the actual recommendation tr. Since a productive recommender would use all the available data to train the model, the time of the recommendation would be the same as the time of the splitting threshold: $tr = t_{threshold}$. Item f is the first item that was consumed after tr by the same user ($tr < tc_f$). This period can be formulated per user as:

$$t_{FirstConsumption} = tc_f - tr. \tag{8}$$

On the one hand, the time span $t_{FirstConsumption}$ between the recommendation and the first consumption of any item can be seen as optimal value for the timeliness measure. It is the lowest possible value a timeliness deviation can take ($t_{FirstConsumption} \leq MATD$). Thus, a recommender system aims at forecasting the next consumed item which corresponds to an MATD of $t_{FirstConsumption}$. On the other hand, it can also be subtracted from the MATD in a so-called Cleaned Timeliness Deviation (CTD). This allows for a better comparison of the algorithm results independently of the login patterns of the users:

$$CTD = \frac{\sum_{i=1}^{K} tc_i - tr}{K} - (tc_f - tr) = \frac{\sum_{i=1}^{K} tc_i - tc_f}{K}. \tag{9}$$

Figure 2 presents examples of CTD and $t_{FirstConsumption}$. The intuitive approach to this alternative timeliness version is that the time of a recommendation is shifted to be the same as the next item consumption by the user. This circumstance enables researchers to aim at reducing the timeliness measure to zero, which corresponds to the best possible Cleaned Timeliness Deviation.

4.3 The Normalized Timeliness Deviation

Remember: In an *increasing time-window* cross-validation, the test set size decreases over time by the same number of activities as the training set increases. When further analyzing the introduced example with three weekly splits between the four weeks of the course, another issue arises: the timeliness value decreases by definition according to the *increasing time-window* cross-validation. The smaller the test set (after week 1 the test data comprises three weeks; after week 3 the test data comprises only one week), the smaller the maximum possible timeliness deviation. This is why the timeliness can additionally be normalized by taking the total duration of the current test set Δt_{Te} into account. The Normalized Timeliness Deviation (NTD) builds on the definition of the CTD:

$$NTD = \frac{\sum_{i=1}^{K} tc_i - tc_f}{K * \Delta t_{Te}}. \tag{10}$$

As a result, the timeliness is given as a percentage of the total available duration. The normalized version lacks information regarding the actual time difference (e.g., the time unit), but can be better expressed in relation to the results of other evaluations. This might help to compare the performance of algorithms in different course settings – for instance, for various course periods.

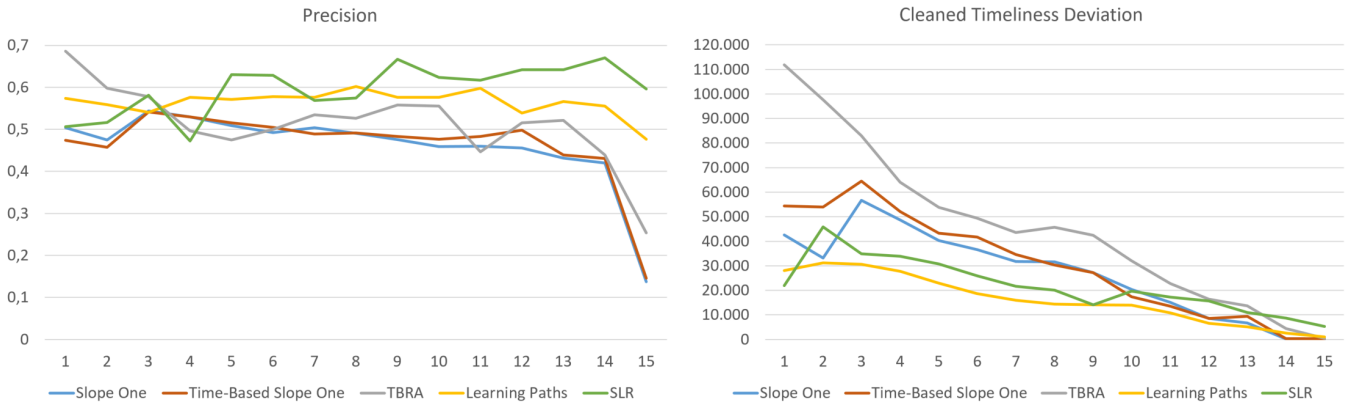

Figure 3: Comparison of the evaluated approaches (each given in the best setting) on the AWT data per week (indicated on the x-axis; left: average precision (y-axis); right: Timeliness values for averaged CTD given in minutes (y-axis).

5 CONDUCTED EXPERIMENTS

We analyzed five different algorithms for the recommendation of appropriate learning resources on a dataset from a university course. The first one is a traditional rating prediction algorithm and serves for comparison. The other four algorithms are time-aware. Thereby, algorithm two and three have been designed and published by other researchers and algorithm four and five are novel ones.

The main dataset has been collected in a course about Advanced Web Technologies (AWT) and comprises 99 active students. In this course, 6 teachers present various topics regarding trends in web programming, such as web apps, multiscreen development, and the web of things. A self-designed learning app gives access to the course materials which comprise 1,006 learning objects grouped into 106 learning units [16]. Thereby, a single learning object is restricted to small learning periods of, at the most, five minutes. A learning unit groups ten learning objects on average. We collected 44,421 Experience API (xAPI) statements which represent the learning object and learning unit accesses of the 99 students. That is an average of 449 item accesses per user.

We used the "increasing time-window" cross-validation since it utilizes the whole past activity data for training the model. The course data is split into 17 sub-datasets where the time threshold shifts by seven days and is defined per week to be on Mondays at midnight. The threshold definition is chosen in order to align the training and test set weeks to calendar weeks. Thereby, with each split, the duration of the training dataset increases by seven days, and the test dataset decreases by the same amount. The evaluations determine numbers on precision and timeliness (here: Cleaned Timeliness Deviation).

5.1 Recommender Algorithms

The first two recommender systems are the basic Slope One algorithm as Collaborative Filtering (CF)-baseline[1] and the extended version with incorporated time-weights which makes it a Time-Aware Recommender System[2]. Another Item-based Collaborative Filtering approach, the Time-based Recommender Approach for Lecture Materials (TBRA) [13], which is based on time-dependent item similarities for the recommendation of study materials, is adapted and evaluated[3]. The fourth and the fifth algorithm are novel ones. The fourth algorithm [17] builds an oriented graph of the learning resources based on the prerequisites between the resources given by teachers and on the navigation activities of classmates. It generates personalized learning paths based on the activities of classmates and the previous interactions of the concerned learner. The next items on the predicted learning path are, therefore, considered as Top-N recommendations. The last algorithm is called Smart Learning Recommender (SLR) [16]. It calculates the learning need of user u at time t for each learning unit using nine factors which include the last access, the performance on exercises and the forgetting effect. The idea is to compensate for the typical drawbacks of Collaborative Filtering, such as the cold-start phase, with more detailed data, and to better adapt the recommendations to the user's needs.

5.2 Results

In the context of all evaluated recommender systems, the SLR performs best on average regarding precision. Figure 3 visualizes the precision and timeliness; for reasons of comparability, each result is presented in the best setting for the Top-3 recommendations for the AWT course. Because all systems are evaluated on the same course, CTD is used. As seen on the precision chart, on a weekly basis, the SLR (green line) is more scattered than the Learning Path algorithm that shows a worse but more stable trend. Regarding timeliness, the Learning Path algorithm still outperforms the SLR. However, the SLR also shows low timeliness values that are the second best on average. Table 2 compares the average results of the algorithms over all course weeks.

[1]The traditional Slope One algorithm has been introduced by Lemire et al. [19]. However, Lemire et al. considered this approach only theoretically for TEL [18] and, at all, it has only been rarely applied in education contexts (e.g., by Verbert et al. [29]).

[2]The time-weighted Slope One algorithm [15] is applied in the context of Technology Enhanced Learning for the first time.

[3]The actual algorithm of presenting similar items [13] has been adopted in this work to fit the recommender's goal and the evaluation approach.

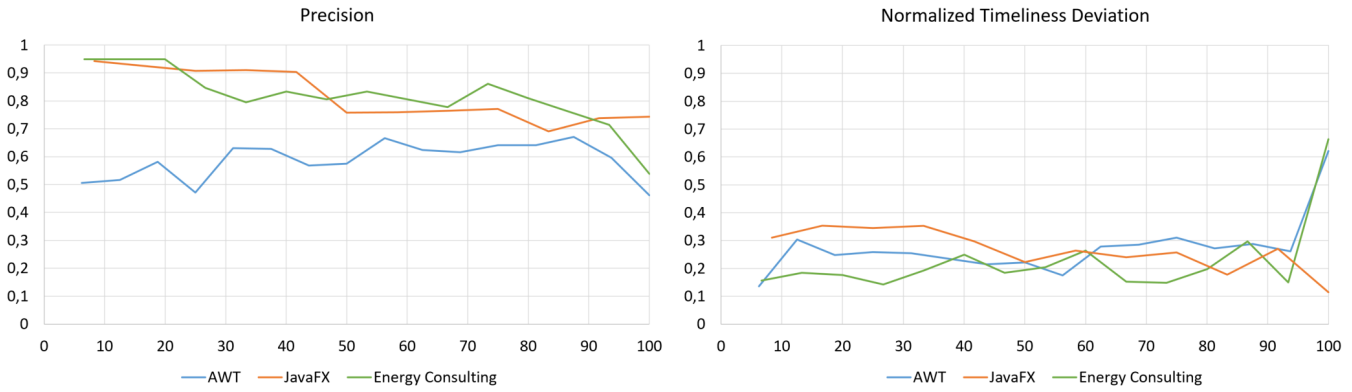

Figure 4: Comparison of the precision (y-axis in left figure) and Normalized Timeliness Deviation (y-axis in right figure) of the SLR reached in the three courses AWT, JavaFX and Energy-Consultant Training. For the sake of comparability, the time presented on the x-axis is given in percent of the course progress instead of course weeks.

Table 2: Precision and Timeliness Deviation for all approaches on the AWT data

Algorithm	Precision	CTD
Slope One	0.459	26,648
TB Slope One	0.465	29,390
TBRA	0.512	36,852
Learning Paths	0.564	18,340
SLR	0.587	20,607

In total over 500 learners used the learning app in various course settings from different educational institutions. Two additional courses serve for comparisons of the timely effects: an open online training on JavaFX with 51 students and a blended-learning course on energy-consulting for 12 craftsmen of a chamber of crafts.

For a cross course comparison, the SLR algorithm has been applied to all three courses: the Advanced Web Technologies lecture, the JavaFX online course and the blended learning course of the Energy-Consultant Training. Figure 4 visualizes the precision and timeliness results. As these courses do not have the same length in weeks, NTD is used instead of CTD. At all, the average precision values are much higher for JavaFX (0.818) and the Energy-Consultant Training (0.815) than for AWT (0.587). This confirms the findings of Verbert et al. [29] that the precision results of educational recommender systems highly depend on the dataset selection and, thus, implicitly on the course setting and the course participants. Interestingly, the timeliness results for all courses are similar to some extent, but the two courses with a final exam (AWT and the energy consultant training) show a significant increase at the end. The same negative effect can be observed in the precision results. This is due to the fact that in both courses the participants learn massively in the final days of the course. The users generate more activity data and especially repeat items more often, which seems to be hard to predict for the algorithm.

6 DISCUSSION

The findings of Verbert et al. [29] in their "Dataset-driven research for improving recommender systems for learning" indicate that the evaluation of recommender systems depend highly on the dataset. They measured the F1-score for Top-10 recommendations based on the Tanimoto-Jaccard Coefficient [28] for 4 different datasets (3 TEL datasets and the MovieLens dataset). The authors state F1-score values which range, depending on the number of considered users, between about 0.05 up to almost 0.3. While the present work theoretically produces better F1-scores of up to 0.39 (for Top-30 recommendations of the TBRA approach), the results of the two evaluations are not comparable at all. In the following, we present the reasons.

The datasets differ in their application area, service origin, quantity, and density – which massively impact the measurements, as also noticed by Verbert et al. [29]. Therefore, it is important to evaluate RSs based on common data – which leads to a high demand for open educational, academic datasets [8].

The algorithms and feature selections of Verbert's approach and the analysis in this work differ significantly. However, evaluations should have at least the same goal when they are compared: While Verbert et al. want to prove the appropriateness of the analyzed datasets [29], the evaluation in this work analyzes the appropriateness of different RS algorithms for the same dataset. As both evaluations do not have consistency between the evaluation setting regarding algorithm selection and data, the results are not comparable.

While the standard cross-validation setting typically produces reliable results, it does not work within a closed-course setting, where, for instance, most items are relevant (e.g., because they must be learned to pass the final exam). If every item in the course would be marked as relevant, the cross-validation would produce precision results of 100% every time, because no matter what is recommended, it is automatically relevant. That is why, it is important to carefully define the set of relevant items – in this work, it is the set of all items that have been accessed after the point in time of the recommendation (item accesses in the test set). This, of course, is

only possible in a time-window cross-validation. Thus, results of a traditional evaluation procedure like the n-fold cross-validation for the analysis of course item recommendations are not comparable with the results introduced in this work.

6.1 Data Splitting

The validation procedure in this work is an "increasing time-window" cross-validation that better represents real-life conditions because it splits the data according to the time sequence of the collected data [3]. Usually, this manner of splitting gives worse results for precision. However, it better represents the real-world conditions of courses. For instance, the cold start phase is not considered by a time-independent n-fold splitting where user–item activity is always selected randomly from the whole period of the available data. During our evaluations, different course phases have been determined that influence precision and timeliness:

(1) Cold Start Phase at the beginning of a course.
(2) Guided Learning Period within the period of the regular lectures or assessment submissions.
(3) Holidays and Breaks: When there are no face-to-face meetings or submissions.
(4) Learning Phases for courses that end with a final assessment.
(5) After Course Phase where learners infrequently access the course contents again.

The analyzed courses indicate also a high time dependency when analyzing the different course weeks separately. Therefore, a recommender system for course items should be evaluated with a time-dependent evaluation framework.

6.2 Measurement Values

In contrast to error and deviation measurements (such as MAE and RMSE), precision, recall, and F1-score indicate the quality of the Top-N composition independently of the underlying score range of the algorithm. Since, especially for settings with a low number of N items in the Top-N list, the recall value is comparatively low, it has a huge effect on the F1-score. To some extent, the greater the number of items presented in the Top-N list, the higher the recall value and the higher the resulting F1-score. Thus, analyses of recall values in different Top-N settings would indicate that it is better to present more items to the user as higher recall values are reached. However, a very large number of presented recommendations offered at a single glance is counterproductive to the main aim of a recommender system. When the user is overwhelmed by the number of recommendations, the learner's item selection process is not supported at all. Thus, in the case of a closed-corpus recommender system, the recall value can be neglected when the learner should learn all relevant items. It is not important to indicate how many of the relevant items are presented in the Top-N list (recall), but how many presented items are relevant (precision).

Precision, recall, and F-measures do not yield information about the timely relevance of the recommendations. Although relevant items are in the test set, they might only become relevant at the end of a course. The introduced timeliness measures take this point into account and indicate average time deviations which can be stated as an absolute value (in order to classify the results as in the MATD and CTD) or as a normalized value presenting the timeliness in relation to the possible time range (as done via the NTD). In contrast to precision and recall, the timeliness measure must be as low as possible. However, it works only in conjunction with the precision value, as all non-relevant (or not-accessed) recommendations are not covered by the timeliness measures. Thus, its main aim is to support the differentiation between algorithms when they show similar precision results.

6.3 Limitations

The time-window cross-validations also have some drawbacks – especially when the timespan of the dataset is small. For closed-corpus courses, an increasing time-window cross-validation can only be applied in the time between course start and course end. Thereby, the first and the last threshold splits are likely to show worse results compared to the rest. This is an effect of small dataset sizes: In the case of the conducted courses, the first split (week 1 for training and the rest for testing) does not allow for an adequate training of the algorithms. Moreover, the last split (all but the last week for training and just the last week for testing) does not allow the recommendations to be tested in the same way as done before. The test set comprises a much smaller dataset than the training set and thus it is challenging to identify items for the final week. In turn, the recall value increases, and in the end indicates the same effect, because there are fewer items in the test set that have a higher probability of being part of the Top-N list. However, the conducted experiments show that the number of activities increases in the final few weeks of a course (especially for courses with final assessments). This lessens the effect at the end. Moreover, the presented limitations represent precisely the conditions of a real course where a recommender system does not have very much data at the beginning and where it must recommend very focused items at the end.

7 CONCLUSION

Time is important for recommender systems for learning items in a closed-course. Thus, these recommendations should be analyzed with the help of a time-dependent evaluation framework, as traditional evaluation procedures, such as the common n-fold cross-validation setting, randomly split the item data regardless of any time constraints. However, a course typically comprises different time phases that must be taken into account for the evaluation. The paper proposed a novel evaluation framework for recommender systems in closed-courses, and for TARSs in general. It is adapted from other research areas and incorporates qualitative and quantitative criteria. Therefore, researchers must describe their methodological procedure and can utilize common measures if their meaning for the evaluation is well defined. The goal of presenting *appropriate* recommendations was translated into those measurable values, such as the precision and timeliness deviation. The latter is a new measure that indicates the time accuracy of the recommendations. The Mean Absolute Timeliness Deviation, Cleaned Timeliness Deviation and Normalized Timeliness Deviation present different flavors of the average timespan when a recommended relevant item has been accessed after its recommendation.

The findings of the discussion shall encourage researchers to follow this special formal evaluation framework. Further work needs

to be done to evaluate existing RS algorithms in TEL with this new framework as well as to analyze and compare the appropriateness of these algorithms in different course settings and phases.

ACKNOWLEDGMENTS

The authors thank the entire Smart Learning team for their outstanding work and many constructive ideas. Special thanks to the instructors of the different courses. This work is partly supported by the German Federal Ministry of Education and Research (funding codes 01PD14002D and 01PD17002D).

REFERENCES

[1] Alejandro Bellogin, Pablo Castells, and Ivan Cantador. 2011. Precision-oriented evaluation of recommender systems: an algorithmic comparison. In *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 333–336.

[2] James Bennett and Stan Lanning. 2007. The netflix prize. In *Proceedings of KDD cup and workshop, San Jose, California*, Vol. 2007. 35.

[3] Pedro G Campos, Fernando Díez, and Iván Cantador. 2014. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction* 24, 1-2 (2014), 67–119.

[4] Mohamed Amine Chatti, Simona Dakova, Hendrik Thus, and Ulrik Schroeder. 2013. Tag-based collaborative filtering recommendation in personal learning environments. *IEEE Transactions on Learning Technologies* 6, 4 (2013), 337–349.

[5] Charles LA Clarke, Maheedhar Kolla, Gordon V Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 659–666.

[6] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 39–46.

[7] Félix Hernández del Olmo and Elena Gaudioso. 2008. Evaluation of recommender systems: A new approach. *Expert Systems with Applications, Elsevier* 35, 3 (2008), 790–804.

[8] Hendrik Drachsler, Toine Bogers, Riina Vuorikari, Katrien Verbert, Erik Duval, Nikos Manouselis, Guenter Beham, Stephanie Lindstaedt, Hermann Stern, Martin Friedrich, et al. 2010. Issues and considerations regarding sharable data sets for recommender systems in technology enhanced learning. *Procedia Computer Science, Elsevier* 1, 2 (2010), 2849–2858.

[9] Hendrik Drachsler, Hans GK Hummel, and Rob Koper. 2009. Identifying the goal, user model and conditions of recommender systems for formal and informal learning. *Journal of Digital Information* 10, 2 (2009).

[10] Mojisola Erdt, Alejandro Fernandez, and Christoph Rensing. 2015. Evaluating recommender systems for technology enhanced learning: a quantitative survey. *IEEE Transactions on Learning Technologies* 8, 4 (2015), 326–344.

[11] Asela Gunawardana and Guy Shani. 2009. A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research, JMLR* 10, Dec (2009), 2935–2962.

[12] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)* 22, 1 (2004), 5–53.

[13] Christoph Hermann. 2010. Time-based recommendations for lecture materials. In *2010 World Conference on Educational Multimedia, Hypermedia and Telecommunications*. 1028–1033.

[14] Neil Hurley and Mi Zhang. 2011. Novelty and diversity in top-n recommendation–analysis and evaluation. *ACM Transactions on Internet Technology (TOIT)* 10, 4 (2011), 14.

[15] Tong Qiang Jiang and Wei Lu. 2013. Improved Slope One Algorithm Based on Time Weight. In *Instruments, Measurement, Electronics and Information Engineering (Applied Mechanics and Materials)*, Vol. 347. Trans Tech Publications, 2365–2368. https://doi.org/10.4028/www.scientific.net/AMM.347-350.2365

[16] Christopher Krauss. 2018. Time-dependent recommender systems for the prediction of appropriate learning objects. *Dissertation at TU Berlin* (2018).

[17] Christopher Krauss, Andreas Salzmann, and Agathe Merceron. 2018. Branched Learning Paths for the Recommendation of Personalized Sequences of Course Items. *The 16th E-Learning Symposium on Computer Science (DeLFI 2018) of the E-Learning Section of the German Informatics Society (Gesellschaft fuer Informatik), Frankfurt, Germany.* (2018).

[18] 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, Emerald Group Publishing Limited* 2, 3 (2005), 179–188.

[19] Daniel Lemire and Anna Maclachlan. 2005. Slope One Predictors for Online Rating-Based Collaborative Filtering. *Proceedings of SIAM Data Mining (SDM'05)* (2005).

[20] Ankit Malpani, Balaraman Ravindran, and Hema Murthy. 2011. Personalized Intelligent Tutoring System Using Reinforcement Learning.. In *International FLAIRS Conference of the The Florida Artificial Intelligence Research Society (FLAIRS'2011), AAAI*.

[21] 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.

[22] Bradley N Miller, Istvan Albert, Shyong K Lam, Joseph A Konstan, and John Riedl. 2003. MovieLens unplugged: experiences with an occasionally connected recommender system. In *Proceedings of the 8th international conference on Intelligent user interfaces*. ACM, 263–266.

[23] Roy Rada. 1998. Efficiency and effectiveness in computer-supported peer-peer learning. *Computers and Education, Elsevier* 30, 3 (1998), 137 – 146. https://doi.org/10.1016/S0360-1315(97)00042-0

[24] Alan Said and Alejandro Bellogín. 2014. Comparative recommender system evaluation: benchmarking recommendation frameworks. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 129–136.

[25] BH Sreenivasa Sarma and Balaraman Ravindran. 2007. Intelligent tutoring systems using reinforcement learning to teach autistic students. In *Home Informatics and Telematics: ICT for The Next Billion*. Springer, 65–78.

[26] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-Based Collaborative Filtering Recommendation Algorithms. In *GroupLens Research Group/Army HPC Research Center*.

[27] Carlotta Schatten and Lars Schmidt-Thieme. 2014. Adaptive Content Sequencing without Domain Information. In *International Conference on Computer Supported Education (CSEDU14)*. 25–33.

[28] Taffee T Tanimoto. 1958. Elementary mathematical theory of classification and prediction. *International Business Machines Corp.* (1958).

[29] 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.

[30] 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. *IEEE Transactions on Learning Technologies* 5, 4 (2012), 318–335.

[31] Emmanouil Vozalis and Konstantinos G Margaritis. 2003. Analysis of recommender systems algorithms. In *The 6th Hellenic European Conference on Computer Mathematics & its Applications*. 732–745.

[32] Stephan Weibelzahl. 2001. Evaluation of adaptive systems. *User Modeling, Springer* (2001), 292–294.

[33] Blaine R Worthen, James R Sanders, and Jody L Fitzpatrick. 1997. Program evaluation. *Alternative approaches and practical guidelines, Prentice Hall* 2 (1997).

[34] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond clicks: dwell time for personalization. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 113–120.