

Learning Path Recommendation for MOOCs Using Sequential Patterns and Matching Similarity

Wei Song^(✉)[0000-0003-0649-8850] and Qihao Zhang

School of Information Science and Technology, North China University of Technology, Beijing 100144, China
songwei@ncut.edu.cn

Abstract. Learning path recommendation plays a pivotal role in guiding learners through a series of courses in a proper sequence based on their past learning experiences. Such recommendations are crucial for enhancing the learning outcomes of MOOCs for diverse learners. Given that both the historical learning courses and the recommended learning path can be represented as sequential patterns (SPs), it is reasonable to approach the learning path recommendation problem from the perspective of SP mining. In addition to support, we also incorporate three other factors—course learning days, grades, and engagement—to model frequent high-utility sequential patterns (FHUSPs). When recommending a learning path, FHUSPs that match the target user's learning history and are common among good learners while rare among not-so-good learners are given priority. Experimental results on two real-world datasets demonstrate the accuracy of the proposed method.

Keywords: Recommender System, MOOC, Learning Path, Frequent High-Utility Sequential Pattern, Consecutive Matching Similarity.

1 Introduction

Massive Open Online Courses (MOOCs) [6], accessible to a vast number of learners at no cost through the Internet, are transforming the method by which individuals engage with education. Unlike traditional school education, MOOC learners lack a structured curriculum, and their levels of knowledge vary. This diversity leads to the persistently high dropout rates observed in MOOC learning. As a result, there is a growing focus on recommending courses [11] or even learning paths [3] for MOOC learners.

The prevalent method for course recommendation [4] involves suggesting to learners courses that share similarities with those they have previously taken or courses that are similar to their current studies. For instance, this approach to course recommendation might suggest courses with akin titles and overlapping content, such as “Algorithm Design and Analysis”, “Computing Theory”, and “Data Structure and Application” to a learner currently study of “Data Structure”. While this recommendation method is beneficial for broadening learners' knowledge, it is evident that the majority of colleges and universities do not structure their courses in this manner.

To address this issue, one may broaden the foundation of recommendations from a single course to a sequence of courses arranged in historical learning order—that is, a learning path. Still considering the example of a learner enrolled in a “data structure” course, the learning path recommendation can offer a sequence of courses rooted in “data structure”, such as “database”, “data mining”, and “big data”, instead of suggesting courses similar to “data structure”. With reasonable learning path, a learner can undergo a systematic training within the MOOCs environment.

The most straightforward approach to suggesting a learning path is to utilize a universal talent training program. In other words, recommendations are made based on the relationship between the pre-order and post-order of courses within the talent training plan. For instance, if “Data Structure” is a course scheduled for the third semester in a computer science program, this approach might suggest to a student currently enrolled in “Data Structure” a learning path that includes “Database” in the fourth semester and “Operating Systems” in the fifth semester. While straightforward, this method may yield identical recommendations for a broad spectrum of diverse learners.

In order to offer personalized learning paths, we employ sequential pattern mining (SPM) [5] to identify course learning sequences that can be utilized for recommendations. As an important task in the field of data mining, SPM aims to discover interesting sequences of itemsets from data containing temporal or sequential information. Presently, SPM finds extensive application across various domains, including the analysis of learning behaviors in MOOCs [8, 9] and recommender systems [2, 7].

In this paper, we use SPM to discover sequences of courses for learning path recommendation. For this problem, the key issue is how to measure the interestingness of sequential patterns (SPs). We choose SPs that are more prevalent among good learners but less common among not-so-good learners for the purpose of recommendation. Furthermore, both frequency and utility are taken into consideration when measuring SPs. We test the recommendation performance of the proposed method on two real-world datasets and validate its superiority in recommendation accuracy.

2 Problem Description

2.1 Frequent SPM

An *item* is represented as a quintuple (c, t, d, g, e) , where c is a course, t is the enrollment time, d is the duration in study days, g is the grade, and e is the number of engagements. A *sequence* $S = \langle (c_1, t_1, d_1, g_1, e_1), (c_2, t_2, d_2, g_2, e_2), \dots, (c_n, t_n, d_n, g_n, e_n) \rangle$ is a list of time-ordered items, where for any $1 \leq i < j \leq n$, $t_i < t_j$ holds. $S[i]$ ($1 \leq i \leq n$) denotes the i th item in S .

A sequence $S = \langle (c_1, t_1, d_1, g_1, e_1), (c_2, t_2, d_2, g_2, e_2), \dots, (c_n, t_n, d_n, g_n, e_n) \rangle$ is called a *subsequence* of another sequence $S' = \langle (c'_1, t'_1, d'_1, g'_1, e'_1), (c'_2, t'_2, d'_2, g'_2, e'_2), \dots, (c'_m, t'_m, d'_m, g'_m, e'_m) \rangle$ ($n \leq m$), denoted by $S \sqsubseteq S'$, if there exist integers $1 \leq i_1 < \dots < i_n \leq m$ such that $S[1].c = S'[i_1].c$, $S[2].c = S'[i_2].c$, ..., $S[n].c = S'[i_n].c$, where $S[i].c$ is the course of $S[i]$. It should be noted that, at each time, only a single item rather than an

itemset is used in this paper. This is because students can only enroll on one course at one time in the MOOC data used in this study. The ordered list of pairs $\langle S^*[i_1], S^*[i_2], \dots, S^*[i_n] \rangle$ is called an *occurrence* of S in S^* , denoted by $Occ(S, S^*)$.

A *sequence database SDB* is a set of 2-tuples (sid, IS) , where sid is called a *sequence-id* and IS an *input sequence*. A tuple (sid, IS) in a sequence database **SDB** is said to *contain* a sequence S if $S \subseteq IS$. For the MOOC data used in this paper, each sequence has at most only one occurrence in one input sequence.

Consider two input sequences IS_X and IS_Y containing S . It is easy to understand that the enrollment times in $Occ(S, IS_X)$ are not equal to the enrollment times in $Occ(S, IS_Y)$. Thus, the enrollment times are omitted in the mining results in this paper. Formally, $S = \langle (c_1, d_1, g_1, e_1), (c_2, d_2, g_2, e_2), \dots, (c_n, d_n, g_n, e_n) \rangle$ is called an *SP*, where c_1, c_2, \dots, c_n are time-ordered courses without specific enrollment times. $S.c_i (1 \leq i \leq n)$ denotes the i th course of S .

The number of tuples in a sequence database **SDB** containing sequence S is called the *support* of S , denoted by $sup(S)$. The set of input sequences in tuples of **SDB** containing sequence S is called the *support set* of S , denoted by $sup_set(S)$.

Let min_sup be the user-specified *minimum support threshold*. An SP S is a *frequent SP* (FSP) in the sequence database **SDB** if $sup(S) \geq min_sup$. The *frequent SPM* problem is to find the complete set of FSPs in **SDB** with respect to min_sup .

2.2 Learning Path Recommendation Using SPM

Given a target learner's course learning sequence $S_t = \langle (c_1, d_1, g_1, e_1), (c_2, d_2, g_2, e_2), \dots, (c_k, d_k, g_k, e_k) \rangle$ and a sequence database **SDB**, the learning path recommendation using SPM typically involves the following steps.

First, interesting SPs (e.g., FSPs) are discovered from SDB. Next, among the resulting SPs, determine $sup_set(S_t)$. Then, SPs in $sup_set(S_t)$ are ranked according to a specific measure. Finally, for any $S \in sup_set(S_t)$, recommend the learning path consisting of courses located after c_k in S to the target learner. It should be noted that all recommended learning paths are still ranked in the same order as the SPs containing them.

Table 1. Example sequence database.

sid	Input sequence
1	(Artificial Intelligence, 48, 72, 792), (Data Science, 123, 90, 32), (Software Engineering, 53, 53, 145), (Cybersecurity, 21, 32, 136)
2	(Artificial Intelligence, 102, 86, 598), (Data Science, 86, 75, 308), (Machine Learning, 68, 65, 74)
3	(Computer Vision, 68, 78, 101), (Data Science, 126, 92, 971), (Software Engineering, 91, 62, 325)
4	(Artificial Intelligence, 69, 74, 268), (Data Science, 35, 42, 69), (Cybersecurity, 68, 85, 121), (Machine Learning, 43, 68, 65)

Consider the example sequence database in Table 1. To make the explanation simple and clear, the enrollment time of each course in all the input sequences is omitted. We take IS_1 as an example to explain the input sequence. This input sequence indicates that a learner has studied four courses—“Artificial Intelligence”, “Data Science”, “Software Engineering”, and “Cybersecurity”—in order. We further explain the first item of

IS_1 , which represents that the learner spent 48 days to studying course “Artificial Intelligence”, achieved a score of 72 points, and engaged in 792 interactions. $S = \langle \text{Artificial Intelligence, Data Science} \rangle$ is an SP contained by IS_1 , IS_2 , and IS_4 . These three input sequences comprise $\text{sup_set}(S)$ and $\text{sup}(S) = 3$. Thus, if the support threshold $\text{min_sup} = 2$, $\langle \text{Artificial Intelligence, Data Science} \rangle$ is an FSP.

3 Frequent High-Utility SPs

In addition to support, utility is also an important measure for evaluating the interestingness of SPs [10]. Recently, researchers have proposed a variety of efficient algorithms for mining high-utility sequence patterns (HUSPs). In real-world big data mining scenarios, the performance of many algorithms for mining HUSPs shows minimal variations. Therefore, the effectiveness of applying HUSPs largely hinges on how the utility value is set.

In this paper, we define the utilities of courses and sequences in the sequence database based on the duration in study days, the grade, and the number of engagements. We categorize these three values into two groups: the duration in study days and grades are employed to indicate the utility of an individual course, while the mean of the number of engagements for each course in a sequence is utilized to represent the utility of the entire sequence.

Definition 1 (Course utility). Let $S = \langle (c_1, d_1, g_1, e_1), (c_2, d_2, g_2, e_2), \dots, (c_n, d_n, g_n, e_n) \rangle$ be an SP. The *course utility* of c_i ($1 \leq i \leq n$) with respect to $IS \in \text{sup_set}(S)$ is defined as

$$uc(c_i, IS) = \frac{g_i}{\log_2(d_i + 1)}. \quad (1)$$

From Eq. 1, it can be seen that the fewer days a learner spends studying course c_i and the higher the grade, the greater the utility the learner obtains from learning c_i .

Consider the first item of IS_1 in the sequence database in Table 1. This learner spent 48 days to studying the “artificial intelligence” course and achieved a score of 72 points. Thus, $u(\text{Artificial Intelligence}, IS_1) = 72/\log_2(48+1) = 12.82$. Similarly, we can calculate the utility values of the other three courses in IS_1 . That is, $u(\text{Data Science}, IS_1) = 12.94$, $u(\text{Software Engineering}, IS_1) = 9.21$, and $u(\text{Cybersecurity}, IS_1) = 7.18$.

Definition 2 (Engagement utility). Let $S = \langle (c_1, d_1, g_1, e_1), (c_2, d_2, g_2, e_2), \dots, (c_n, d_n, g_n, e_n) \rangle$ be an SP. The *engagement utility* of S with respect to $IS \in \text{sup_set}(S)$ is defined as

$$ue(S, IS) = \exp\left(-\frac{1}{n} \sum_{i=1}^n e'_i\right), \quad (2)$$

where

$$e'_i = \frac{e_i - e_{\min}}{e_{\max} - e_{\min}}, \quad (3)$$

in which e_{\max} and e_{\min} are the highest number of engagements and the lowest number of engagements in the entire sequence database, respectively.

From Eq. 2, it is evident that the fewer engagements a learner has during the entire learning process, the higher the engagement utility of the sequence. In other words, if a learner can achieve a high score by simply watching a video with few interactions, it indicates the effectiveness of the learning process.

In the example sequence database in Table 1, $e_{\max} = 971$ and $e_{\min} = 32$. For an SP $S = \langle (\text{Artificial Intelligence}, 48, 72, 792), (\text{Data Science}, 123, 90, 32) \rangle$ with respect to IS_1 , according to Eq. 3, we have $e'_1 = (792 - 32) / (971 - 32) = 0.81$. Similarly, we have $e'_2 = 0$. Thus, $us(S, IS_1) = \exp(-0.81/2) = 0.67$.

Definition 3 (Input sequence utility). Let $S = \langle (c_1, d_1, g_1, e_1), (c_2, d_2, g_2, e_2), \dots, (c_n, d_n, g_n, e_n) \rangle$ be an SP. The *input sequence utility* of S with respect to $IS \in \text{sup_set}(S)$ is defined as

$$iu(S, IS) = ue(S, IS) \times \sum_{i=1}^n uc(c_i, IS). \quad (4)$$

Considering an SP $S = \langle (\text{Artificial Intelligence}, 48, 72, 792), (\text{Data Science}, 123, 90, 32) \rangle$ with respect to IS_1 in the example database presented in Table 1. $iu(S, IS_1) = 0.67 \times (12.82 + 12.94) = 17.26$.

Definition 4 (SP utility). Let **SDB** be a sequence database, $S = \langle c_1, c_2, \dots, c_n \rangle$ be an SP. The *SP utility* of S is defined as

$$u(S) = \sum_{IS \in \text{sup_set}(S)} iu(S, IS). \quad (5)$$

Since different learners spend varying durations in study days, achieve different grades, and have a different number of engagements, when expressing the utility of an SP in the entire database, like the support, only the course title is retained.

Considering the example sequence database shown in Table 1, for $S = \langle \text{Artificial Intelligence}, \text{Data Science} \rangle$, $\text{sup_set}(S) = \{IS_1, IS_2, IS_4\}$. According to Eq. 5, $u(S) = iu(S, IS_1) + iu(S, IS_2) + iu(S, IS_4) = 50.36$.

Definition 5 (Frequent High-Utility SP, FHUSP). Let min_sup be the user-specified *minimum support threshold*, min_util be the user-specified *minimum utility threshold*. An SP S is an *FHUSP* in the sequence database **SDB** if $\text{sup}(S) \geq \text{min_sup}$ and $u(S) \geq \text{min_util}$.

In this paper, we exclusively employ FHUSPs for learning path recommendation. FHUSPs can be discovered by adjusting the utility calculation definition in an algorithm that can simultaneously discover SPs while considering both support and utility, such as the HUSRM Algorithm [12].

Consider the example sequence database presented in Table 1. Given $\min_sup = 2$ and $\min_util = 40$, $S = \langle \text{Artificial Intelligence, Data Science} \rangle$ is an FHUSP, because $sup(S) \geq \min_sup$ and $u(S) \geq \min_util$.

4 Importance Measuring for SPs Based on Matching and Learners Groups

In Sect.3, we utilize support and utility to evaluate the importance of an SP. In this section, we further evaluate SPs for learning path recommendation from two other perspectives. One is how closely the SP used for recommendation matches the historical learning path of the target learner, and the other is the popularity of the SP among good learners and not-so-good learners.

4.1 Consecutive Matching Similarity

The premise we adopt to measure the similarity between an SP and the target learner's historical learning path is that the greater the number of consecutive matching courses, the higher the similarity. To measure similarity of two SPs, we also exclude the duration in study days, grade, and number of engagements. An SP is represented as $S = \langle c_1, c_2, \dots, c_n \rangle$. In other words, if one or more courses are consecutively identical in two SPs, it is considered that there is a matching block between them. Two SPs may share several matching blocks.

Definition 6 (Matching block). Let $S_A = \langle a_1, a_2, \dots, a_m \rangle$ and $S_B = \langle b_1, b_2, \dots, b_n \rangle$ be two SPs. If there are $S'_A = \langle a_i, a_{i+1}, \dots, a_{i+k-1} \rangle \sqsubseteq S_A$ and $S'_B = \langle b_j, b_{j+1}, \dots, b_{j+k-1} \rangle \sqsubseteq S_B$, such that $a_i = b_j, a_{i+1} = b_{j+1}, \dots, a_{i+k-1} = b_{j+k-1}$. We call S'_A or S'_B a *matching block* of S_A and S_B , and the number of courses in this matching block the *length* of this matching block, denoted by $len(S'_A)$ or $len(S'_B)$.

In the case of $S_1 = \langle \text{Artificial Intelligence, Data Science} \rangle$ and $S_2 = \langle \text{Artificial Intelligence, Cybersecurity} \rangle$, $\langle \text{Artificial Intelligence} \rangle$ is a matching block for both S_1 and S_2 , with a length of one.

Definition 7 (Matching block similarity, MBS). Let $S_A = \langle a_1, a_2, \dots, a_m \rangle$ and $S_B = \langle b_1, b_2, \dots, b_n \rangle$ be two SPs, $M = \langle m_i, m_{i+1}, \dots, m_{i+k-1} \rangle$ be a matching block of S_A and S_B . The MBS is defined as

$$sim_B(M) = len(M) / |S_A \cup S_B|, \quad (6)$$

where $S_A \cup S_B$ is the union of courses in both S_A and S_B , and $|S_A \cup S_B|$ is the number of courses in $S_A \cup S_B$.

Consider $S_1 = \langle \text{Artificial Intelligence, Data Science} \rangle$ and $S_2 = \langle \text{Artificial Intelligence, Cybersecurity} \rangle$ with a matching block $M = \langle \text{Artificial Intelligence} \rangle$, $sim_B(M) = 1/|\{\text{Artificial Intelligence, Data Science, Cybersecurity}\}| = 1/3$.

Indeed, two SPs can potentially share multiple matching blocks. To capture this complexity and provide a comprehensive measure of similarity, we introduce a generalized similarity metric.

Definition 8 (Consecutive matching similarity, CMB). Let $S_A = \langle a_1, a_2, \dots, a_m \rangle$ and $S_B = \langle b_1, b_2, \dots, b_n \rangle$ be two SPs with P matching blocks M_1, M_2, \dots, M_P . The MBS is defined as

$$sim_C(S_A, S_B) = \frac{1}{abs(|S_A| - |S_B|) + 1} \times \sum_{l=1}^P (w_l \times sim(M_l)), \quad (7)$$

Where $abs(|S_A| - |S_B|)$ is the absolute value of $(|S_A| - |S_B|)$, and w_l is the weight of the l th block defined as

$$w_l = \frac{len(M_l)}{\sum_{j=1}^P len(M_j)}. \quad (8)$$

Consider two SPs $S_1 = \langle \text{Artificial Intelligence, Data Science, Software Engineering, Cybersecurity} \rangle$ and $S_2 = \langle \text{Artificial Intelligence, Data Science, Cybersecurity} \rangle$. According to Definition 6, there are two matching blocks for S_1 and S_2 , $M_1 = \langle \text{Artificial Intelligence, Data Science} \rangle$ and $M_2 = \langle \text{Cybersecurity} \rangle$. According to Definition 7, $sim_B(M_1) = 1/2$ and $sim_B(M_2) = 1/4$. According to Definition 8, $w_1 = 2/3$ and $w_2 = 1/3$. Thus, $sim_C(S_1, S_2) = (2/3 \times 1/2 + 1/3 \times 1/4) / (abs(4 - 3) + 1) = 5/24$.

4.2 Performance in Different Learner Groups

Al-Twijri et al. [1] recommended courses to learners by mining course learning SPs that are more common in good students and less common in not-so-good students. In their approach, the selection of an SP for recommendation depends on whether its support significantly differs between the two groups of good students and not-so-good students, and such SPs are called emerging SPs.

Definition 9 (Database CMB). Let $S = \langle a_1, a_2, \dots, a_m \rangle$ be an SP, and SDB be a sequence database. The Database CMB of S with respect to SDB is defined as

$$sim_D(S, SDB) = \sum_{S' \in SDB} sim_C(S, S'). \quad (9)$$

It can be seen from Definition 9 that the CMB between an SP S and a sequence database is the sum of the CMB between S and each input sequence in the sequence database.

Definition 10 (Emergence of SP). Let $S = \langle a_1, a_2, \dots, a_m \rangle$ be an SP, SDB_g and SDB_n be sequence database composed of good learners and not-so-good learners. The *emergence* of S with respect to SDB_g and SDB_n is defined as

$$Em(S) = \begin{cases} sim_D(S, SDB_g) / sim_D(S, SDB_n), & \text{if } sim_D(S, SDB_n) \neq 0 \\ sim_D(S, SDB_g), & \text{if } sim_D(S, SDB_n) = 0 \end{cases}. \quad (10)$$

It can be seen from Definition 10 that the emergence of an SP S is the ratio of its database CMB with the sequence database composed of good learners to its database CMB with the sequence database composed of not-so-good learners. If S has no CMB with any sequence in the database composed of not-so-good learners, then its emergence is equal to its database CMB with the database composed of good learners. The higher the value of emergence of S , the more common S appears among good learners, and the rarer it appears among not-so-good learners.

5 The Overall Recommendation Roadmap

5.1 Ranking for Candidate SPs

Ranking the recommendation results is crucial for a recommender system. In this paper, we use the ranking strategy based on both FHUSPs and Emergence of SPs.

Definition 11 (Ranking order, RO). Let $S = \langle a_1, a_2, \dots, a_m \rangle$ be an SP. The RO of S is defined as

$$RO(S) = (\alpha \times sup(S) + (1-\alpha) \times u(S)) \times Em(S), \quad (11)$$

where $\alpha \in [0, 1]$ is the *balance factor*.

It can be seen from Definition 11 that the RO takes into account three factors. The first is the support of an SP; the second is the utility value, jointly defined by the duration in study days, the grade, and the number of engagements; and the third is the disparity in the matching of an SP among good learners and not-so-good learners. For an SP S , the higher the value of $RO(S)$, the higher it is ranked in recommendation list.

5.2 Learning Path Recommendation

Based on the preceding discussion, Algorithm 1 outlines the proposed learning path recommendation method, referred to as Learning Path Recommendation using SPs and Matching similarity (LPR-SPM).

Algorithm 1 LPR-SPM	
Input	The target learner's course learning sequence S_t , sequence database SDB , minimum support threshold min_sup , minimum utility threshold min_util , balance factor α .
Output	The learning paths recommended to the target user
1	Discover all FHUSPs.
2	Divide SDB into SDB_g and SDB_n according to learners' average grades.
3	$sup_set(S_t) \leftarrow$ FHUSPs contain S_t .
4	for each $S \in sup_set(S_t)$ do

```

5      Calculate  $Em(S_i)$  and  $RO(S_i)$ .
6  end for
7      Rank all FHUSPs in  $sup\_set(S_i)$  in descending order of  $RO$ .
8      Recommend the top- $K$  learning paths to the target user.

```

In Algorithm 1, Line 1 mines all FHUSPs that meet both support and utility thresholds. Subsequently, Line 2 divides the original sequence database into two subsets based on the average grades of learners: one for good learners and the other for not-so-good learners. Line 3 then constructs the support set for the target learner's learning sequence solely using FHUSPs. Following this, the loop from Line 4 to Line 6 calculates the emergence values and ranking orders of all FHUSPs within the support set of the target learner's learning sequence. In Line 7, the FHUSPs are arranged in descending order of their RO values. Finally, the top- K FHUSPs with the highest RO values are utilized for learning path recommendation. As described in Sect. 2.2, the recommended learning paths consist of a sequence of courses within each FHUSP within $sup_set(S_i)$, positioned after the completion of the last course taken by the target user.

6 Performance Evaluation

6.1 Data Preprocessing

We used two real MOOC datasets, namely the Canvas Network and HarvardX, to verify the recommendation performance of the LPR-SPM method. The datasets were divided into two parts, with 80% used as the training set and 20% used as the testing set.

The Canvas Network dataset (<https://doi.org/10.7910/DVN/1XORAL>) consists of de-identified data derived from Canvas Network open courses, spanning from January 2014 to September 2015. The HarvardX dataset (<https://doi.org/10.7910/DVN/26147>) comprises de-identified data from the inaugural year (Academic Year 2013: Fall 2012, Spring 2013, and Summer 2013) of HarvardX courses hosted on the edX platform.

Both datasets contain numerous zero or null values. We preserved the data records containing more values and discarded those with fewer values. Specifically, for the Canvas Network dataset, records with a value of zero were retained while records with a null value were discarded. Conversely, for the HarvardX dataset, records with a null value were retained while records with a zero value were discarded.

Another critical aspect of data preprocessing is establishing criteria to differentiate between student groups. Given the varying sparsity levels of the datasets, in the Canvas Network dataset, good learners completed more than 3 courses with an average score of at least 90, while in the HarvardX dataset, good learners completed more than 2 courses with an average score of at least 90. Learners failing to meet these criteria are classified as not-so-good learners.

6.2 Parameter Settings

In the proposed LPR-SPM method, the minimum support threshold min_sup , minimum utility threshold min_util , and balance factor α must be set to appropriate values. We

first outlined the approximate range of these parameters, and then determined their optimal values by progressive refinement. Specifically, we set $min_sup = 0.8$, $min_util = 80$, and $\alpha = 0.7$. It should be noted that the support used for evaluation was the ratio of the number of input sequences containing the target pattern to the total number of input sequences in the sequence database; that is, the support values used in experiments were in the range $[0, 1]$.

For the two comparison methods, we set $min_sup = 0.2$ for the FAST-USP-based recommendation and $min_util = 40$ for the USpan-based recommendation.

6.3 Experimental Results

We compared LPR-SPM method with two methods: one method uses unexpected SPs discovered by FAST-USP [8] for recommending learning path, and the other method uses high-utility SPs mined by USpan [10] for recommendation. For the FAST-USP-based recommendation, FAST-USP can directly mine unexpected SPs from the Canvas Network and HarvardX datasets. Next, the discovered unexpected SPs that match the target learner's course learning sequence are ranked in descending order of unexpected support [8], and then used for recommending learning paths. The flow of the USpan-based recommendation is similar to that of the FAST-USP-based recommendation. The difference lies in that SP utility described in Definition 4 is used to replace the general-purpose utility in USpan [10], and the SPs for recommendation are sorted in utility descending order.

We first compared the precision and recall performance when recommending different numbers of learning paths and present the comparison results in Tables 2 and 3. All values reported in the two tables are the averages obtained from five independent runs. The best results of each set of experiments are underlined.

Table 2. Comparison results for the Canvas Network dataset.

Metric	Method	Top-2	Top-4	Top-6	Top-8	Top-10
Precision	FAST-USP	0.186	0.168	0.154	0.146	0.132
	USpan	0.162	0.153	0.145	0.138	0.123
	LPR-SPM	<u>0.213</u>	<u>0.183</u>	<u>0.173</u>	<u>0.158</u>	<u>0.149</u>
Recall	FAST-USP	0.158	0.184	0.208	0.228	0.231
	USpan	0.142	0.162	0.193	0.206	0.216
	LPR-SPM	<u>0.169</u>	<u>0.208</u>	<u>0.241</u>	<u>0.276</u>	<u>0.292</u>

Table 3. Comparison results for the HarvardX dataset.

Metric	Method	Top-2	Top-4	Top-6	Top-8	Top-10
Precision	FAST-USP	0.268	0.242	0.228	0.208	0.185
	USpan	0.242	0.222	0.203	0.186	0.171
	LPR-SPM	<u>0.292</u>	<u>0.286</u>	<u>0.246</u>	<u>0.222</u>	<u>0.216</u>
Recall	FAST-USP	0.182	0.232	0.243	0.271	0.291
	USpan	0.163	0.186	0.212	0.236	0.246
	LPR-SPM	0.203	0.239	0.289	0.308	0.325

Tables 2 and 3 show that the recommendation performance of LPR-SPM consistently outperformed the other two comparison methods under different number of recommended learning paths. This shows that LPR-SPM accurately used duration in study

days, grade, and the number of engagements to model the utility of learning SPs, and fully considered those SPs that appear commonly among good learners but less frequently among not-so-good learners. Consequently, the selected SPs for recommendation are effective in improving the recommendation accuracy.

For the other two methods, the recommendation accuracy of the FAST-USP-based method outperformed that of the USpan-based method. This is primarily due to the fact that the FAST-USP algorithm is specifically tailored for extracting unexpected SPs from MOOC data. It takes into account factors such as length, the number of both course types and transitions between course types. In contrast, USpan is a general-purpose SPM algorithm that only considers utility.

We further compared the F1-score values of the three methods, as depicted in Fig. 1. Here, we no longer compared the specific values for different numbers of recommended paths, but instead, we compared the F1-score calculated using the averages of the five groups of precision and recall values from Tables 2 and 3.

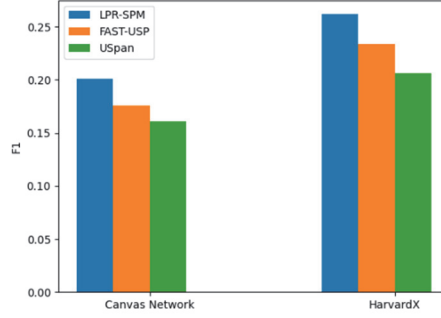


Fig. 1. Comparison results for the F1-score.

This set of experiments was consistent with the experiments on precision and recall. LPR-SPM still performed best, followed by FAST-USP. Although FAST-USP also considered the relevant information of each course, since its main purpose is to analyze MOOC learning behavior rather than recommend learning paths, it did not consider the matching information with the target sequence. Therefore, the recommended performance was not as good as LPR-SPM.

7 Conclusions

Recommending suitable learning paths in the MOOC environment holds great promise. In this paper, we introduce an SP-based learning path recommendation method. To mine SPs for recommendation, we consider three factors: the duration of study days, grades, and the number of engagements. These factors are used to calculate the utility of SPs, which, combined with support, determine FHUSPs for recommendation. Then, only those FHUSPs matching the target users are used for recommending learning paths, and FHUSPs that are common among good learners but rare among not-so-good

learners are ranked high in the recommendation list. The experimental results demonstrate that the recommendation accuracy of the proposed LPR-SPM method is high.

Acknowledgments. This work was partially supported by the National Natural Science Foundation of China (61977001).

References

1. Al-Twijri, M. I., Luna, J. M., Herrera, F., Ventura, S.: Course recommendation based on sequences: an evolutionary search of emerging sequential patterns. *Cogn. Comput.* **14**(4), 1474–1495 (2022). <https://doi.org/10.1007/s12559-022-10015-5>
2. Gan, M., Tan, C.: Mining multiple sequential patterns through multi-graph representation for next point-of-interest recommendation. *World Wide Web* **26**(4), 1345–1370 (2023). <https://doi.org/10.1007/s11280-022-01094-3>
3. Khalid, A., Lundqvist, K., Yates, A., Azam, M. A.: Online learning path recommender system for MOOCs. In: *Proceedings of the IEEE Global Engineering Education Conference*, pp. 1–10 (2023).
4. Li, S., Zhao, Y., Guo, L., Ren, M., Li, J., Zhang, L., Li, K.: Quantification and prediction of engagement: Applied to personalized course recommendation to reduce dropout in MOOCs. *Inf. Process. Manag.* **61**(1), (2024).
5. Li, Y., Zhang, C., Li, J., Song, W., Qi, Z., Wu, Y., Wu, X.: MCoR-Miner: Maximal co-occurrence nonoverlapping sequential rule mining. *IEEE Trans. Knowl. Data Eng.* **35**(9), 9531–9546 (2023).
6. Song, W., Wang, Z.: Improved clustering strategies for learning style identification in massive open online courses. In: Tan, Y., Shi, Y. (eds) *DMBD 2022. Communications in Computer and Information Science*, vol. 1744. Springer, Singapore. https://doi.org/10.1007/978-981-19-9297-1_18
7. Song, W., Yang, K.: Personalized recommendation based on weighted sequence similarity. In: Wen Z., Li T. (eds) *Practical Applications of Intelligent Systems. Advances in Intelligent Systems and Computing*, vol. 279, pp. 657–666. Springer, Berlin, Heidelberg. (2014). https://doi.org/10.1007/978-3-642-54927-4_62
8. Song, W., Ye, W.: Mining unexpected sequential patterns from MOOC data. In: *Proceedings of the IEEE International Conference on Big Knowledge*, pp. 434–439 (2021).
9. Song, W., Ye, W., Fournier-Viger, P.: Mining sequential patterns with flexible constraints from MOOC data. *Appl. Intell.* **52**(14), 16458–16474 (2022). <https://doi.org/10.1007/s10489-021-03122-7>
10. Yin, J., Zheng Z., Cao, L.: USpan: an efficient algorithm for mining high utility sequential patterns. In: *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 660–668 (2012).
11. Zhang, H., Huang, T., Lv, Z., Liu, S., Zhou, Z.: MCRS: A course recommendation system for MOOCs. *Multim. Tools Appl.* **77**(6), 7051–7069 (2018). <https://doi.org/10.1007/s11042-017-4620-2>
12. Zida, S., Fournier-Viger, P., Wu, C.W., Lin, J.C.W., Tseng, V.S: Efficient mining of high-utility sequential rules. In: Perner, P. (eds) *MLDM 2015. LNCS*, vol. 9166, pp. 157–171. Springer, Cham. https://doi.org/10.1007/978-3-319-21024-7_11