Mining Sequential Patterns of Historical Purchases for E-commerce Recommendation

Raj Bhatta, C. I. Ezeife(✉), and Mahreen Nasir Butt

School of Computer Science, University of Windsor,
401 Sunset Avenue, Windsor, ON N9B3P4, Canada
{bhatt11y,cezeife,nasir11}@uwindsor.ca

Abstract. In E-commerce Recommendation system, accuracy will be improved if more complex sequential patterns of user purchase behavior are learned and included in its user-item matrix input, to make it more informative before collaborative filtering. Existing recommendation systems that attempt to use mining and some sequences are those referred to as LiuRec09, ChoiRec12, SuChenRec15, and HPCRec18. These systems use mining based techniques of clustering, frequent pattern mining with similarity measures of purchases and clicks to predict the probability of purchases by users as their ratings before running collaborative filtering algorithm. HPCRec18 improved on the user-item matrix both quantitatively (finding values where there were 0 ratings) and qualitatively (finding specific interest values where there were 1 ratings). None of these algorithms explored enriching the user-item matrix with sequential pattern of customer clicks and purchases to capture better customer behavior. This paper proposes an algorithm called HSPRec (Historical Sequential Pattern Recommendation System), which mines frequent sequential click and purchase patterns for enriching the (i) user-item matrix quantitatively, and (ii) qualitatively. Then, finally, the improved matrix is used for collaborative filtering for better recommendations. Experimental results with mean absolute error, precision and recall show that the proposed sequential pattern mining based recommendation system, HSPRec provides more accurate recommendations than the tested existing systems.

Keywords: E-commerce recommendation · Sequential pattern mining · Collaborative filtering · User-item matrix quality · Consequential bond

1 Introduction

The main goal of a recommender system is to generate meaningful recommendations to a user for items that might interest them given the user-item

This research was supported by the Natural Science and Engineering Research Council (NSERC) of Canada under an Operating grant (OGP-0194134) and a University of Windsor grant.

C. Ordonez et al. (Eds.): DaWaK 2019, LNCS 11708, pp. 57–72, 2019.
https://doi.org/10.1007/978-3-030-27520-4_5

rating matrix, which indicates the rating of each item by each user. An important application of recommendation system is in the E-commerce domain [9]. In E-commerce environment, the implicit ratings derived from historical purchase and/or clickstream data are used as rating of items in the user-item rating matrix [6,7]. Many users may not be ready to provide explicit ratings for many items and there is a large set of continuously growing items (products), a very small percentage of which, each user may have purchased. In addition, users' purchase behavior change with time so that sequences of clicks and purchases play important role in capturing more realistic users' purchase behavior. Thus, one of the main challenges in the field of recommendation is how to integrate sequential pattern of purchases generated from historical and/or clickstream E-commerce data to capture more complex purchase behavior.

Collaborative filtering is one of the most widely used recommendation techniques. It accepts a user-item rating matrix (R), having ratings of each item (i) by user (u_i) denoted as r_{u_i}. The goal of collaborative filtering is to predict r_{u_i}, a rating of user u on item i which may be unknown, by going through the following four major steps [1]:

1. Compute the mean rating for each user u_j using all of their rated items.
2. Calculate the similarity between a target user v and all other users u_j. Similarity can be computed with Cosine Similarity (v, u_j) or Pearson Correlation coefficient [1] function.
3. Find similar users of target user v as their Top-N users.
4. Predict rating for target user v for item i using only ratings of v's Top-N peer group.

Collaborative filtering considers the searching of closest neighbors to generate matching recommendations. However, what people want from recommender systems is not whether the system can predict rating values accurately, but recommendations that match their interests according to time span. Thus, E-commerce recommendation system accuracy will be improved if more complex sequential patterns of user historical purchase behavior are learned and included in the user-item matrix to make it quantitatively and qualitatively rich before applying collaborative filtering.

1.1 Sequential Pattern Mining

Sequential pattern mining algorithms (for example GSP [10]) discover repeating sequential patterns (known as frequent sequences) from input historical sequential databases that can be used later to analyze user purchase behavior by finding the association between sequences of itemsets. In other words, it is the process of extracting sequential patterns whose support exceeds or is equal to a pre-defined minimum support threshold. Formally, given (a) a set of sequential records (called sequences) representing a sequential database D, (b) a minimum support threshold minsup, (c) a set of k unique candidate one length events (1-items), $C_1 = i_1, i_2, \ldots, i_n$, the problem of mining sequential patterns is that

of finding the set of all frequent sequences FS in the given sequence database D of items I at the given minimum support.

Sequence database is composed of a collection of sequences $\{s_1, s_2, .., s_n\}$ that are arranged with respect to time [5]. A sequence database can be represented as a tuple ⟨SID, sequential pattern⟩, where SID: represents the sequence identifier and sequential pattern contains list of item sets with each item set containing 1 or more items. Table 2 shows an example of daily purchase sequential database. Sequential pattern mining algorithms available to find the frequent sequences from sequential databases include the GSP (Generalized Sequential Pattern Mining) [10].

Table 1. Daily click sequential database

SID	Click sequence
1	<(1,2,3), (7,5,3), (1,6), (6), (1,5)>
2	<(1,4), (6,3), (1,2), (1,2,5,6)>
3	<(1,5), (6,5,2), (6), (5)>
4	<(2,7), (6,6,7)>
5	<(1,5)>

Table 2. Daily purchase sequential database

SID	Purchase sequence
1	<(1,2), (3), (6), (7), (5)>
2	<(1,4), (3), (2), (1,2,5,6)>
3	<(1), (2), (6), (5)>
4	<(2), (6,7)>

An Example Sequential Pattern Mining with the GSP: Let us consider daily purchase sequential database (Table 2) as input, minimum support = 2, 1-candidate set $(C_1) = \{1, 2, 3, 4, 5, 6, 7\}$.

1. Find 1-frequent sequence (L_1) satisfying minimum support: Check the minimum support threshold of each singleton item and remove items that do not have occurrence count in the database greater than or equal to the minimum support threshold, to generate large or frequent 1-sequences L_1. In our case, $L_1 = \{<(1)>, <(2)>, <(3)>, <(5)>, <(6)>, <(7)>\}$.
2. Generate candidate sequence (C_2) using $L_1 \bowtie_{GSPjoin} L_1$: In this step, to generate larger 2-candidate sequence set, use 1-frequent sequence found in previous step to join itself GSPjoin way. The GSPjoin of L_{k-1} with L_{k-1} generates larger k-candidate set, by having every sequence (W_i) found in first L_{k-1} join with the other sequence (W_k) in the second L_{k-1} if sub-sequences obtained by removal of the first element of W_i and the last element of W_k are the same.
3. Find 2-frequent sequences (L_2) by scanning the database and counting the support of each C_2 sequence to retain only those sequences with support greater than or equal to the minimum support threshold.
4. Repeat the processes of candidate sequence generate (C_k) and frequent sequence count (L_k) steps until the result of either a candidate sequence generate step or a frequent sequence count step yields an empty set.
5. Output: Output frequent sequences as the union of all L's or L = $\{L_1 \cup L_2 \cup \ldots \cup L_m\}$ if the last found L is L_m.

1.2 Historical Data

E-commerce historical data consist of a list of items clicked and/or purchased by a user over a specific period of time. A fragment of E-commerce historical database data is presented in Table 3 with schema {Uid, Click, Clickstart, Clickend, Purchase, Purchasetime} where Uid represents User identity, Click represents list of items clicked by the user. Clickstart and Clickend represent the timestamps when the user started clicking items and when the click is terminated. Furthermore, purchase contains list of items purchased by the user and Purchasetime represents timestamp when the purchase happened.

Table 3. Historical e-commerce data showing clicks and purchases

Uid	Click	Clickstart	Clickend	Purchase	Purchasetime
1	1, 2, 3	2014-04-04 11:25:14	2014-04-04 11:45:19	1, 2	2014-04-04 11:30:11
1	7, 5, 3	2014-04-05 15:30:07	2014-04-05 15:59:36	3	2014-04-05 15:56:32
1	1, 6	2014-04-06 4:10:01	2014-04-06 04:30:29	6	2014-04-06 4:18:26
1	6	2014-04-07 8:50:29	2014-04-07 9:50:07	7	2014-04-07 8:59:21
1	1, 5	2014-04-08 14:10:24	2014-04-08 14:25:18	5	2014-04-08 14:19:55
2	1, 4	2014-04-13 4:01:11	2014-04-13 4:30:15	1, 4	2014-04-13 04:04:34
2	6, 3	2014-04-15 9:30:34	2014-04-15 9:40:11	3	2014-04-15 09:34:37
2	1, 2	2014-04-17 13:40:11	2014-04-17 13:59:11	2	2014-04-17 13:54:48
2	1, 2, 5, 6	2014-04-17 11:30:18	2014-04-17 11:50:19	1, 2, 5, 6	2014-04-17 11:44:55
3	1, 5	2014-04-20 09:40:45	2014-04-20 10:10:15	1	2014-04-20 10:02:53
3	6, 5, 2	2014-04-21 11:59:59	2014-04-21 12:10:39	2	2014-04-21 12:07:15
3	6	2014-04-22 17:05:19	2014-04-22 17:30:06	6	2014-04-22 17:10:28
3	5	2014-04-23 11:00:05	2014-04-23 11:20:15	5	2014-04-23 11:06:37
4	2, 7	2014-04-23 12:00:11	2014-04-23 12:30:10	2	2014-04-23 12:06:37
4	6, 6, 7	2014-04-26 9:45:11	2014-04-26 10:20:13	6, 7	2014-04-26 10:06:37
5	1, 5	2014-04-27 16:30:25	2014-04-27 16:45:45	?	

1.3 Consequential Bond (CB)

E-commerce data contains information showing some relationship between user clicks and purchases of products, used to derive a consequential bond between clicks and purchases as introduced by Xiao and Ezeife [12] in their HPCRec18 system. The term consequential bond originated from the concept that a customer who clicks on some items will ultimately purchase an item from a list of clicks in most of the cases. For example, the historical data in Table 3 shows that user 1 clicked items $\{1, 2, 3\}$ and purchased $\{1, 2\}$. Thus, there is a relationship between clicks and purchases used to derive the consequential bond between clicks and purchases.

1.4 Problem Definition

Given E-commerce historical click and purchase data over a certain period of time as input, the problem being addressed by this paper is to find the frequent periodic (daily, weekly, monthly) sequential purchase and click patterns in the first stage. Then, these sequential purchase and click patterns can be used to make user-item matrix qualitatively (specifying level of interest or value for already rated items) and quantitatively (finding possible rating for previously unknown ratings) rich before applying collaborative filtering (CF) to improve the overall accuracy of recommendation.

1.5 Contributions

A main limitation of existing related systems such as (HPCRec18 [12]) is that they treated the entire clicks and purchases of items equally and did not integrate frequent sequential patterns to capture more real life customer sequence patterns of purchase behavior inside consequential bond. Thus, in this paper, we propose a system called Historical sequential pattern recommendation (HSPRec) to discover frequent historical sequential pattern from clicks and purchases so that discovered frequent sequential patterns are first used to improve consequential bond and user-item frequency matrix to further improve recommendations.

HSPRec Feature Contributions

1. Using sequential patterns to enhance consequential bonds between clicks and purchases.
2. Using sequential patterns to enhance user-item rating matrix quantitatively by filling missing ratings.
3. Improving accuracy of recommendation qualitatively and quantitatively with consequential bond and sequential patterns respectively to generate a rich user-item matrix for collaborative filtering to improve recommendations.

Procedural Contribution
To make the specified feature contributions, this paper proposes the HSPRec system (Algorithm 1), which first creates sequential databases of purchases and clicks, finds sequential patterns of purchases and clicks with the GSP algorithm [10]. Then, it uses sequential pattern rules to find next possible purchases and consequential bonds between clicks and purchases before collaborative filtering.

2 Related Work

1. Segmentation Based Approach-LiuRec09 ([7]): This approach is based on forming a segmentation of user on the basis of Recency, Frequency, Monetary (RFM) using K-means clustering method, where Recency is the period since the last purchase, Frequency is the number of purchases and Monetary is the amount of money spent. Once the RFM segmentation is created, users are further

segmented using transaction matrix. The transactions matrix captures the list of items purchased or not purchased by users over a monthly period in a given product list. From the transaction matrix, users' purchases are further segmented into T-2, T-1, and T, where T represents the current purchases, T-1 and T-2 represents two previous purchases. Finally, association rule mining is used to match and select Top-N neighbors from the cluster to which a target user belongs using binary choice to derive the prediction score of an item not yet purchased by the target user based on the frequency count of the item, scanning the purchase data of k-neighbors. The major drawbacks of LiuRec09 [7] are: (a) It does not learn sequential purchases for user-item matrix creation (b) The utility of item such as price are ignored during the recommendation generation.

2. User Transactions Based Preference Approach-ChoiRec12 ([3]): Users are not always willing to provide a rating or they may provide a false rating. Thus, ChoiRec12 developed a system that derives preference ratings from a transaction data by using the number of times $user_u$ purchased $item_i$ with respect to total transactions of users. Once preference ratings are determined, they are used to formulate user-item rating matrix for collaborative filtering. Major drawbacks of ChoiRec12 are: (a) Sequential purchase pattern is not included from historical or clickstream data during user-item matrix creation. (b) No provision for recommending item to infrequent users.

3. Common Interest Based Approach-ChenRec15 ([11]): It is based on finding the common interest similarity (frequency, duration, and path) between purchase patterns of users to discover the closest neighbors. For the frequency similarity computation, it computes total hits that occurred in an item or category with respect to the total length of users' browsing path. For duration similarity computation, it computes the total time spent on each category with respect to total time spent by users. Finally, for path similarity computation, it uses the longest common subsequence comparison. By selecting Top-N similar users from three indicators, the CF method is used to select close neighbors. The major drawbacks are: (a) It requires domain knowledge for categories, and only supports category level recommendations. (b) Fails to integrate sequential purchase patterns during formation of user-item rating matrix.

4. Historical and Clickstream Based Recommendation-HPCRec18 ([12]): Xiao and Ezeife in [12] proposed HPCRec18 system, which normalizes the purchase frequency matrix to improve rating quality, and mines the session-based consequential bond between clicks and purchases to generate potential ratings to improve the rating quantity. Furthermore, they used frequency of historical purchased items to enrich the user-item matrix from both quantitative (finding possible value for 0 rating), and qualitative (finding more precise value for 1 rating) by using normalization of user-item purchase frequency matrix and consequential bond between clicks and purchases. They also did not integrate mining of sequential patterns of purchases to capture even better customer historical behavior.

3 Proposed Historical Sequential Recommendation (HSPRec) System

The major goal of the proposed HSPRec is to mine frequent sequential patterns from E-commerce historical data to enhance user-item rating matrix from discovered pattern. Thus, HSPRec takes minimum support, historical user-item purchase frequency matrix and consequential bond as input to generate rich user-item matrix as output, (see Algorithm 1).

Algorithm 1. Historical sequential recommendation (HSPRec) system

input : minimum support(s), historical user-item purchase frequency matrix (M), consequential bond (CB), historical purchase database (DB), historical click database (CDB)

output : user-item purchase frequency matrix (M_2)

intermediates: historical sequential purchase database (SDB), weighted purchase pattern (WP), historical sequential click database (SCDB), rule recommended purchase items (RPI), each user u's rating of item i in the matrices is referred to as r_{ui}

```
1  purchase sequential database (SDB)← SHOD (DB) using Algorithm 2
   present in section 3.1;
2  user-item purchase frequency matrix (M1) ← M modified with Sequential
   Pattern Rule (SDB) using section 3.2 ;
3  for each user u do
4  |  weighted purchase pattern for user u, (WP_u) ← null ;
5  end
6  for each user u do
7  |  if u has both click and purchase sequences then
8  |  |  compute Click Purchase similarity CPS(click sequence, purchase
   |  |  sequence) from SCDB and SDB using section 3.3;
9  |  |  weighted purchase pattern for user u, (WP_u) ← CPS(click sequence,
   |  |  purchase sequence);
10 |  else
11 |  |  rule recommended purchase items (RPI) ← Sequential Pattern Rule
   |  |  (SCDB) ;
12 |  |  weighted purchase pattern for user u, (WP_u) ← CPS(click sequence,
   |  |  purchase sequence) using section 3.3;
13 |  end
14 |  rating of item i by user u (r_ui) ← Weighted Frequent Purchase Pattern
   |  Miner (WP_u) ;
15 |  M2 ← M1 modified with rating r_ui;
16 end
```

Steps in the Proposed HSPRec System

1. Convert historical purchase information (Table 3) to user-item purchase frequency matrix (Table 4) by counting the number of each item purchased by each user. For example, User 2 purchased items 1 and 2 twice but other items once.

2. Create daily purchase sequential database (Table 2) of customer purchase (Table 3) by applying sequential historical periodic database generation (SHOD) algorithm presented in Sect. 3.1 to the customer purchase database as input. For example, User 2 daily purchase sequence is <(1, 4), (3), (2), (1, 2, 5, 6)>, which shows User 2 purchased item 1 and item 4 together on the same day and purchased item 3 on the next day then purchased item 2 on another day and finally purchased items 1, 2, 5 and 6 together on the next day.
3. Input daily purchase sequential database (Table 2) to Sequential Pattern Rule (SPR) module presented in Sect. 3.2 to generate sequential rule from frequent purchase sequences. For example, 1-frequent purchase sequences = {<(1)>, <(2)>, <(3)>, <(5)>, <(6)>, <(7)>}. Similarly, some of the 2-frequent purchase sequences = {<(6), (5)>, <(3), (6)>, <(3), (5)>, <(2), (7)>, <(2), (6)>, <(2), (5)>} and some of the 3-frequent purchase sequences = {<(2), (6), (5)>, <(1), (6), (5)>, <(1), (3), (6)>, <(1), (3), (5)>, <(1), (2), (6)>}. Thus, some of the possible sequential purchase pattern rules based on frequent purchase sequences are: (a) 1, 5 → 3 (b) 2, 6 → 1 (c) 2, 6 → 5, where rule (a) states that if user purchases items 1 and 5 together then user will purchase item 3 in next purchase, which will be applied in case of for user 3 in user-item purchase frequency matrix (Table 4).
4. Apply purchase sequential rule in user-item purchase frequency matrix to improve quantity of ratings.
5. For each user, where click happened without a purchase such as user 5 in Table 6, we use consequential bond between clicks and purchases derived as sequential pattern rules mined from the click sequence database. The consequential bond for user 5 in Table 4 is computed by finding frequent click sequential patterns using the SPR (click sequence database) call. From the generated sequential patterns, we filter out only strong rules containing the sequences in user 5 clicks so that they can be used to derive user 5 is possible future purchases.
6. Next, compute Click and Purchase Similarity (click sequence, purchase sequence) using longest common subsequence rate (LCSR) and frequency similarity (FS) Eq. 4 presented of Sect. 3.3.
7. Assign Click Purchase Similarity (click sequence, purchase sequence) value to purchase patterns present in consequential bond (Table 6) to create weighted purchase patterns (Table 7).
8. Input weighted purchase patterns to Weighted Frequent Purchase Pattern Miner (WFPPM) presented in Sect. 3.4 to calculate the weight for each frequent individual item based on its occurrence in weighted purchase patterns.
9. Repeat the steps 5, 6, 7, and 8 if there are more users with clicks but without purchases, otherwise assign computed items weights to modify enhanced user-item frequency matrix (Table 5) and apply collaborative filtering.

Table 4. User-item purchase frequency matrix (M)

User/item	1	2	3	4	5	6	7
User1	1	1	1	**?**	1	1	1
User2	2	2	1	1	1	1	**?**
User3	1	1	**?**	**?**	1	1	**?**
User4	**?**	1	**?**	**?**	**?**	1	1
User5	?	?	?	?	?	?	?

Table 5. Enhanced user-item frequency matrix with Sequential Pattern Rule (M_1)

User/item	1	2	3	4	5	6	7
User1	1	1	1	?	1	1	1
User2	2	2	1	1	1	1	?
User3	1	1	1	?	1	1	?
User4	1	1	?	?	1	1	1
User5	?	?	?	?	?	?	?

Table 6. Consequential bond table showing clicks and purchases from historical data

UID	Click sequence	Purchase sequences
1	<(1,2,3), (7,5,3), (1,6), (6), (1,5)>	<(1,2), (3), (6), (7), (5)>
2	<(1,4), (6,3), (1,2), (1,2,5,6)>	<(1,4), (3), (2), (1,2,5,6)>
3	<(1,5), (6,5,2), (6), (5)>	<(1), (2), (6), (5)>
4	<(2,7), (6,6,7)>	<(2), (6,7)>
5	<(1,5)>	?

3.1 Historical Periodic Sequential Database Generation (SHOD) Module

The proposed sequential historical pattern generation (SHOD) module takes historical (click or purchase database) data as input and produces periodic (daily, weekly, monthly) sequential (click or purchase) database as output as present in Algorithm 2. This algorithm is based on checking timestamp between items to form periodic sequences if the difference between purchase times by the same user are within 24 h, 24 * 7 h respectively.

Example of creating periodic sequential database Check time difference between purchased products.

(a) If the time difference between two product purchases is less than 24 h, it adds itemID to itemset in daily.txt file. In our case, purchased time difference between two products {1, 2} purchased by user {Tuserid = 1} is less than 24 h. So, it adds two items to itemset in daily.txt as: **1, 2**.
(b) However, if the time difference between purchased items is more than 24 h, it adds -I to indicate the end of itemset and adds itemID after -I.
For example, **1, 2 -I 3**.

If user identity is not similar, then add -I and -S after item to indicate the end of itemset and sequence respectively and goto step 2 by updating temporary variable. Repeat steps with each record read until there is no record in the historical database. In our case, the created daily purchase sequential database

Algorithm 2. Historical periodic sequential database (SHOD)

input : historical click or purchase data
output : periodic (daily, weekly, monthly) sequential database
intermediate: Tuserid= temporary userid, Ttimestamp= temporary timestamp

```
historical.txt ← extract userid, itemid, timestamp from historical data;
read first line from historical.txt and store userid, timestamp into Tuserid,
Ttimestamp ;
for each user u, u ∈ historical.txt do
    if userid==Tuserid then
        Tdur ← (Ttimestamp - timestamp) ;
        if Tdur <= 24 hrs then
            add itemid to daily-sequence-database.txt and goto step 3 ;
        else
            add -I to indicate end of itemset and goto step 3 ;
        end
        if Tdur > 24 hrs and Tdur <=168 hrs then
            add itemid to weekly-sequence-database.txt and goto step 3;
        else
            add -I to indicate end of itemset and goto step 3 ;
        end
        add -S to indicate end of sequence and goto step 3 ;
    else
        add -I after itemid to indicate end of itemset, -S to indicate end of
        sequence and update Tuserid and goto step 3 ;
    end
end
```

is presented in Table 2 and the same steps can be repeated to generate daily click sequential database by using click item database as input.

3.2 Sequential Pattern Rule (SPR) Module

Sequential Pattern Rule (SPR) is based on the use of frequent sequential patterns created from periodic sequential database. Thus, input of SPR is periodic historical sequential database and output is recommended possible purchase items derived from generated sequential patterns rules. The major steps in SPR are:

1. **Frequent sequences generation:** Generates frequent sequences using Generalized Sequential Pattern Mining (GSP) algorithm [10]. Let us consider **input**: Daily click sequential database in Table 1, **minimum support** = 2 and **candidate set** $(C_1) = \{1, 2, 3, 4, 5, 6, 7\}$ and **algorithm** = GSP as defined in Sect. 1.1. **output**: frequent sequential patterns. Some example frequent sequential patterns are: 1-frequent sequences: {<(1)>, <(2)>, <(3)>, <(5)>, <(6)>}. Some of the 2-frequent sequences: {<(1), (2)>, <(1), (3)>, <(3), (6)>, <(5), (6)>, <(1, 5)>}. Some of 3-frequent sequences: {<(1), (2), (6)>, <(3), (1), (5)>, <(1), (5), (6)>, <(6), (1, 5)>}

2. **Rule generation:** Represent frequent sequences in the form of $U_{\text{click}} \rightarrow U_{\text{purchase}}$, where the left-hand side of the rule refers to clicked item set while the right-hand side is the recommended item to be purchased. Furthermore, to verify the validity of SPR, confidence of SPR is defined as:

$$Confidence(U_{\text{click}} \rightarrow U_{\text{purchase}}) = \frac{Support(U_{\text{click}} \cup U_{\text{purchase}})}{Support(U_{\text{click}})} \quad (1)$$

   Here, some of the rules from frequent clicks sequences are: (a) $(1,5) \rightarrow (1),(3)$, (b) $(1,5) \rightarrow (3),(1)$, (c) $(1,2) \rightarrow (1,6)$, (d) $(1)(5) \rightarrow (6),(5)$
3. **Rule section:** Assume we are ony interested in rules that satisfy the following criteria: (1) At least two antecedents. (2) Confidence $> 50\%$. (3) Select one rule with highest confidence. Let us consider rule (a) recommends User 5 to purchase item 1 and item 3 when item 1 and item 5 are clicked together.

3.3 Click Purchase Similarity (CPS)

To compute the CPS similarity between click sequence and purchase sequence of each user, we have used sequence similarity and frequency similarity of the two sequences. **Sequence Similarity:** It is based on longest common subsequence rate (LCSR) [2] and presented in Eq. (2).

$$LCSR(X,Y) = \frac{LCS(X,Y)}{max(X,Y)} \quad (2)$$

In our case, X represents click sequence and Y represents purchase sequence and LCS is defined as:

$$LCS(Xi,Yj) = \begin{cases} \phi \quad if \quad i=0 \quad or \quad j=0 \\ LCS(X_{i-1},Y_{j-1}) \cap X_i \quad if \quad x_i = y_i \\ longest(LCS(X_i,Y_{j-1}), LCS(X_{i-1},Y_j)) \quad if \quad x_i != y_i \end{cases}$$

Frequency Similarity: First, form distinct sets of items from both click and purchase sequential patterns and count the number of items occurring in each sequence to form the vectors specifying the number of times a user clicked or purchased a particular item. Then, apply Cosine frequency similarity (Eq. (3)) to the click and purchase vectors.

$$Cosine(X,Y) = \frac{X_1 * Y_1 + X_2 * Y_2 + ...X_n * Y_n}{\sqrt{X_1^2 + X_2^2 + ... + X_n^2}\sqrt{Y_1^2 + Y_2^2 + ... + Y_n^2}} \quad (3)$$

Thus,

$$CPS = \alpha * LCSR(X,Y) + \beta * FS(X,Y) \quad (4)$$

where $\alpha + \beta = 1, 0 < \alpha, < \beta < 1, \alpha$ and β are weights assigned to reflect the importance of the two sequences of similarity and frequency.

Example of CPS (Click Sequence, Purchase Sequence)

To compute CPS similarity between click sequence (X) = $<(2,7),(6,6,7)>$ and purchase sequence (Y) = $<(2),(6,7)>$, we take the following steps:

1. Compute the longest common subsequences, LCS(X, Y) between click and purchase sequence. For example, LCS (<(2, 7), (6, 6, 7)>, <(2), (6, 7)>) is 3, because of common subsequence (2), (6, 7).
2. Find the maximum number of items occurring in click or purchase sequence as Max(X, Y). In our case, MAX(X, Y) is 5.
3. Compute sequence similarity of click (X) and purchase (Y) sequences as LCS(X, Y)/Max(X, Y)= 3/5 = 0.6.
4. Compute the frequencies of items in click and purchase sequences. In our case, we have used the format [(item): number of occurrence]. So, frequency counts of clicks is: [(2):1, (6):2, (7):2]. Similarly, frequency counts of purchases is: [(2):1, (6):1, (7):1].
5. Use Cosine similarity function in Eq. 3 to get the frequency similarity between click sequence (X) and purchase sequence (Y) as Cosine (X, Y). In our case, Cosine (X, Y) = 0.96.
6. The Click Purchase Similarity of user click and purchase sequences CPS(X, Y) is presented in Eq. 4. In our case, $CPS(X,Y) = 0.8 * 0.6 + 0.2 * 0.96 = 0.67$, where $\alpha = 0.8$ and $\beta = 0.2$.

This CPS(X, Y) can be used as weight or probability that user u will purchase the entire sequence as shown in Table 7.

Table 7. Weighted purchased patterns

Purchase	<(1,2), (3), (6), (7), (5)>	<(1,4), (3), (2), (1,2,5,6)>	<(1), (2), (6), (5)>	<(2), (6,7)>	<(1), (3)>
CPS	0.624	0.834	0.636	0.67	0.5

3.4 Weighted Frequent Purchase Pattern Miner (WFPPM) Module

Weighted Frequent Purchase Pattern Miner(WFPPM) takes weighted purchase pattern as input (present in Table 7) and weighted purchase patterns is created by assigning CPS (click sequence, purchase sequence) value to purchase patterns in consequential bond Table 6 to generate frequent purchase patterns with weight under user specified minimum threshold as output. Major steps of WFPPM are:

1. **Count support of items:** Count occurrence of items presented in weighted purchase pattern (Table 7). For example, {support(1):5, support(2):5, support(3):3, support(4):1, support(5):3, support(6):4, support(7):2}.
2. **Calculate weight for individual item:** Compute weight of individual item from weighted purchase pattern (Table 7) using CPS module (Eq. (4)).

$$R, item_i = \frac{\sum_{i=1}^{n} CPS \in item_i}{Support(item_i)} \tag{5}$$

For example, $R_1 = \frac{0.624+0.834+0.834+0.636+0.5}{5} = 0.68$ Similarly, $R_2 = 0.71$, $R_3 = 0.65, R_4 = 0.834, R_5 = 0.698, R_6 = 0.691, R_7 = 0.647$,

3. **Test item weight with minimum threshold:** Define minimum threshold rating for user, here in our case, minimum threshold = 0.2. So, all rating of item are frequent.

3.5 User-Item Matrix Normalization

Normalization in recommendation system helps to predict the level of interest of user on a particular purchased item. Thus, normalization function for a user u's rating of an item i (r_{ui}) takes user-item frequency matrix (Table 8) as input and provides the level of user interest between 0 and 1 using unit vector formula (Eq. (6)).

$$Normalization(r_{ui}) = \frac{r_{ui}}{\sqrt{r_{ui1}^2 + r_{ui2}^2 + ... + r_{uin}^2}} \tag{6}$$

The normalization of enhanced user-item matrix (Table 8) using Eq. 5 is presented in Table 9. For example, normalization of $User_1$ rating on $Item_1 = \frac{1}{\sqrt{1^2+1^2+1^2+1^2+1^2+1^2}} = 0.40$.

Table 8. Enhanced user-item purchase frequency matrix with rating for user 5 (M_1)

User/item	1	2	3	4	5	6	7
User1	1	1	1	?	1	1	1
User2	2	2	1	1	1	1	?
User3	1	1	1	?	1	1	?
User4	1	1	?	?	1	1	1
User5	0.68	0.71	0.65	0.834	0.698	0.691	0.647

Table 9. Normalized user-item purchase frequency matrix (M_2)

User/item	1	2	3	4	5	6	7
User1	0.40	0.40	0.40	?	0.40	0.40	0.40
User2	0.57	0.57	0.28	0.28	0.28	0.28	?
User3	0.44	0.44	0.44	?	0.44	0.44	?
User4	0.44	0.44	?	?	0.44	0.44	0.44
User5	0.68	0.71	0.65	0.834	0.698	0.691	0.647

4 Experimental Design

We implemented our proposed historical sequential pattern mining based recommendation system, HSPRec with two other existing recommendation systems of ChoiRec12 [3] and HPCRec18 [12] in user-based collaborative filtering to evaluate the performance our proposed system in comparison to others. First, the same data obtained from Amazon product data [8] is converted into user-item matrix for each of the approaches of the three algorithms (ChoiRec12, HPCRec18 and HSPRec) before applying collaborative filtering on the user-item matrix to obtain the recommendations for each of the systems. During this conversion, data is modified into intermediate form, which means, when the value is larger than the minimum threshold; this value would be set to one (highest rating). When the value is less than the minimum threshold, this value would be set to zero (lowest rating). To test user-based collaborative filtering, we have used

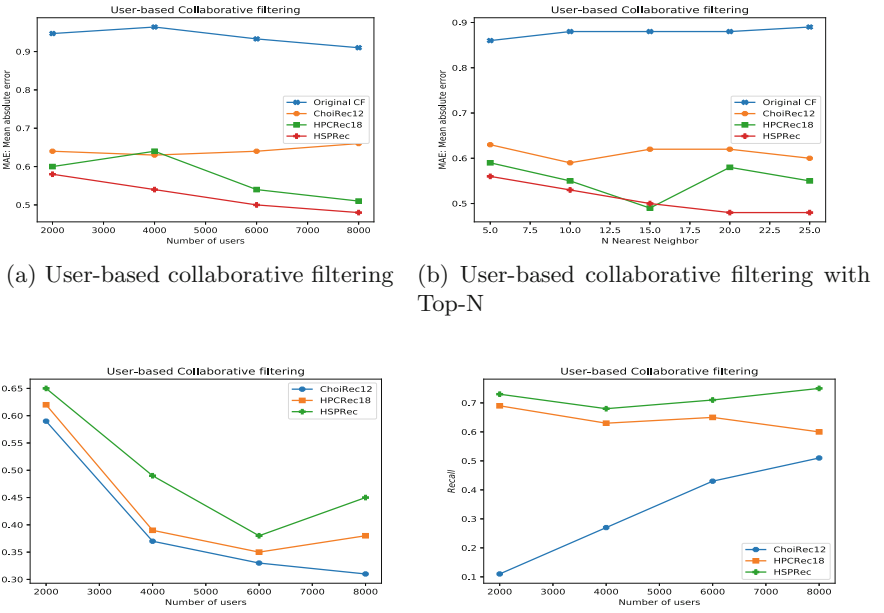
Pearson Correlation Coefficient (PCC). Furthermore, 80% of data is used in training and 20% is used in testing performance. To evaluate the performance of the recommendation system, we have used a different number of users and nearest neighbors using three different evaluation parameters (a) mean absolute error (MAE) (b) precision and (c) recall with https://www.librec.net/ LibRec [4] library available in Java.

4.1 Dataset Selection

To perform experiment, we used data available from Amazon product data [8]. The Amazon data sets consists of 23 different categories such as Books, Electronics, Home and Kitchen, Sports and Outdoors, Cell Phones and Accessories, Grocery and Gourmet Food and many more. Data contains 142.8 million transactional records spanning May 1996–July 2014 but in our experiments we have used data of 2013 and 2014.

4.2 Experimental Results

Figure 1 presents results of our experiments. First, we implemented original CF algorithm with explicit rating available from Amazon, and found very low performance. Then, we implemented choiRec12 [3] with rating derived from each



(a) User-based collaborative filtering

(b) User-based collaborative filtering with Top-N

(c) Precision in user-based collaborative filtering

(d) Recall in user-based collaborative filtering

Fig. 1. Experimental result showing evaluation recommendation systems using MAE, precision and recall

user purchases with respect to total purchases by each user and got better result compared to original CF. Then, we implemented HPCRec18 [12] by using user-item frequency first and used consequential bond in user-item frequency matrix and found better result than choiRec12. Finally, for our proposed HSPRec, we constructed user-matrix purchase frequency matrix at first. Then, we discovered frequent sequences of purchased data to create sequential rules and used sequential rule to enhance user-item matrix quantitatively and we applied enhanced user-item frequency matrix to collaborative filtering and found better result compared to choiRec12, and HPCRec18.

5 Conclusions and Future Work

Many recommendation systems neglect sequential patterns during recommendation. Thus, to verify the necessity of sequential patterns in recommendation, we generated sequential patterns from historical E-commerce data and fed them into collaborative filtering to make user-item matrix rich from quantitative and qualitative perspectives. Furthermore, after evaluation with different systems, we got better results with a sequential pattern based recommendation. Thus, some of the possible future works are: (a) Finding more possible ways of integrating sequential patterns to collaborative filtering. (b) Incorporating multiple data sources based sequential pattern with different data schema, and making recommendations based on the overall data set. (c) Finding the more possible ways of integrating sequential pattern in user-item matrix from online data.

References

1. Aggarwal, C.C.: Recommender Systems. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-29659-3
2. Bergroth, L., Hakonen, H., Raita, T.: A survey of longest common subsequence algorithms. In: Seventh International Symposium on String Processing and Information Retrieval, pp. 39–48 (2000)
3. Choi, K., Yoo, D., Kim, G., Suh, Y.: A hybrid online-product recommendation system: combining implicit rating-based collaborative filtering and sequential pattern analysis. Electron. Commer. Res. Appl. **11**(4), 309–317 (2012)
4. Guo, G., Zhang, J., Sun, Z., Yorke-Smith, N.: LibRec: a java library for recommender systems. In: Posters, Demos, Late-breaking Results and Workshop Proceedings of the 23rd International Conference on User Modeling, Adaptation and Personalization (2015)
5. Han, J., Pei, J., Kamber, M.: Data Mining: Concepts and Techniques. Elsevier, New York (2011)
6. Kim, Y.S., Yum, B.J., Song, J., Kim, S.M.: Development of a recommender system based on navigational and behavioral patterns of customers in e-commerce sites. Expert Syst. Appl. **28**(2), 381–393 (2005)
7. Liu, D.R., Lai, C.H., Lee, W.J.: A hybrid of sequential rules and collaborative filtering for product recommendation. Inf. Sci. **217**(20), 3505–3519 (2009)
8. McAuley, J.: Amazon product data (2019). http://jmcauley.ucsd.edu/data/amazon/

9. Schafer, J.B., Konstan, J.A., Riedl, J.: E-commerce recommendation applications. Data Min. Knowl. Discov. **5**(1–2), 115–153 (2001)
10. Srikant, R., Agrawal, R.: Mining sequential patterns: generalizations and performance improvements. In: Apers, P., Bouzeghoub, M., Gardarin, G. (eds.) EDBT 1996. LNCS, vol. 1057, pp. 1–17. Springer, Heidelberg (1996). https://doi.org/10.1007/BFb0014140
11. Su, Q., Chen, L.: A method for discovering clusters of e-commerce interest patterns using click-stream data. Electron. Commer. Res. Appl. **14**(1), 1–13 (2015)
12. Xiao, Y., Ezeife, C.I.: E-Commerce product recommendation using historical purchases and clickstream data. In: Ordonez, C., Bellatreche, L. (eds.) DaWaK 2018. LNCS, vol. 11031, pp. 70–82. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98539-8_6