International Journal of Approximate Reasoning 146 (2022) 119–132

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

A heuristic concept construction approach to collaborative recommendation

Zhong-Hui Liu [a], Qi Zhao [a], Lu Zou [a], Wei-Hua Xu [b], Fan Min [a,c,*]

[a] School of Computer Science, Southwest Petroleum University, Chengdu, 610500, China
[b] College of Artificial Intelligence, Southwest University, Chongqing, 400715, China
[c] Institute for Artificial Intelligence, Southwest Petroleum University, Chengdu, 610500, China

ARTICLE INFO

Article history:
Received 26 May 2021
Received in revised form 8 February 2022
Accepted 11 April 2022
Available online 26 April 2022

Keywords:
Formal concept analysis
Heuristic algorithm
Local popularity
Recommender system

ABSTRACT

Formal concept analysis was first used in collaborative filtering for over one decade. Popular approaches are based on superconcept-subconcept relationship or boolean matrix factorization. In this paper, we design a heuristic approach to construct a set of approximately strong concepts for recommendation. Here *strong* refers to not only big intent to ensure the similarity among users, but also big extent to ensure the stability of user groups. First, we use the intent threshold as the constraint and the area as the optimization objective to obtain approximately strong concepts. Second, we generate pre-recommendations based on the local popularity of the items implied by each concept. Finally, we determine the actual recommendation according to the number of times the item is pre-recommended to the user. Experiments have been undertaken on five popular datasets with different versions. Results show that our algorithm has lower runtime and/or higher recommendation quality compared with approaches based on concept lattice, matrix factorization, k-nearest neighbors, item-based collaborative filtering and boolean matrix factorization.

1. Introduction

Formal concept analysis (FCA) [36] is a powerful tool for data analysis. It models the binary relational data as the formal context, strongly related object-attribute pairs as formal concepts, and the structure organizing all concepts as the concept lattice. Hence it has been widely used in machine learning [18,26,28], data mining [8,15,32] and knowledge discovery [34,51]. In addition, it is also combined with other theories such as Fuzzy sets [2,30], granular computing [22,23,38], Rough sets [35,46], and three-way decisions [39,42,47] to provide sophisticated solutions for more fields such as cognitive computing [10,40].

The construction of concept lattice is the bottleneck of FCA while applied to real data. For example, given a formal context with thousands of objects and attributes, we need to construct a lattice containing millions of concepts. One approach is to design parallel methods [16] to speed up the construct process. However, the space complexity is not decreased. The other approach, which is more popular, is to reduce the size of the concept lattice. Granular-computing-based methods [31,37] consider sub-context, especially different attribute subsets each time when generating concepts. Clustering-based methods

* Corresponding author at: School of Computer Science, Southwest Petroleum University, Chengdu, 610500, China.
E-mail addresses: liuzhonghui@swpu.edu.cn (Z.-H. Liu), zhaoqi@stu.swpu.edu.cn (Q. Zhao), zoulu@stu.swpu.edu.cn (L. Zou), datongxuweihua@126.com (W.-H. Xu), minfan@swpu.edu.cn (F. Min).

https://doi.org/10.1016/j.ijar.2022.04.004
0888-613X/

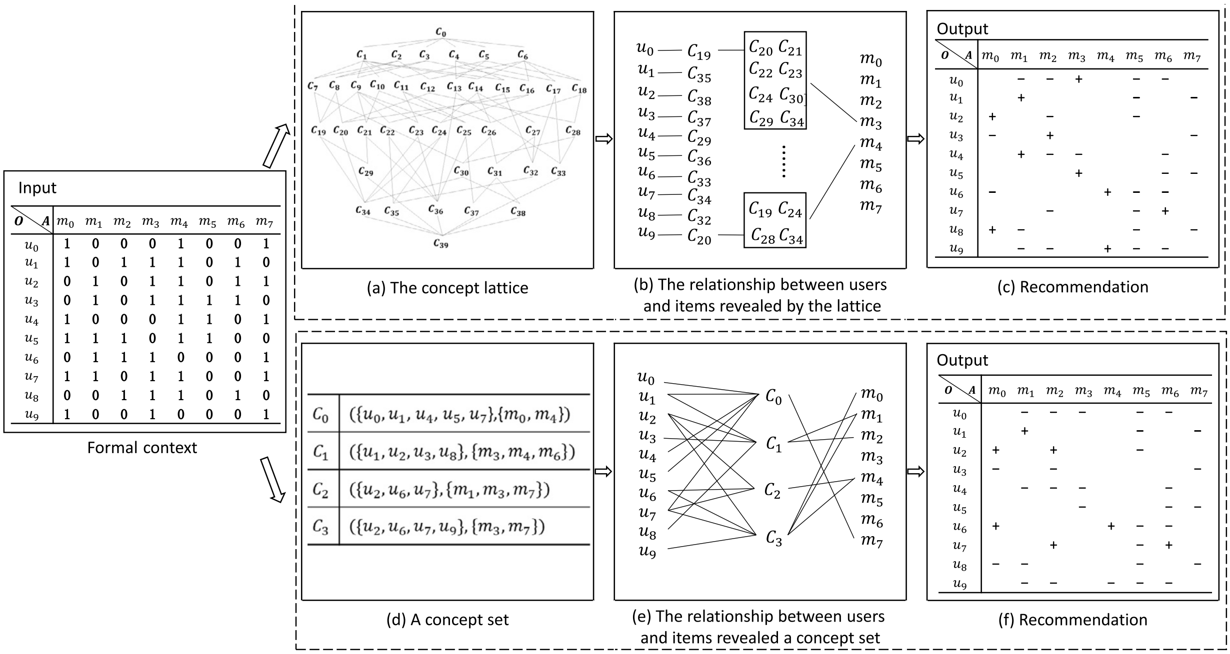


Fig. 1. Comparison of recommendations based on the concept lattice and a concept set. The left part is the original formal context. (a) is the complete lattice with 40 concepts. (b) indicates the relationship between users and items revealed by all relevant concepts and concept sets. (c) is the recommendation, where + indicates recommend, and − indicates not recommend. (d) is a concept set. (e) indicates the relationship between users and items revealed a concept set. (f) shows the recommendations with a concept set.

[17,21,33] merge the nodes of each equivalent class or each cluster into one node in the modified lattice. Similarity-based methods [1,53] remove redundant nodes according to similarity measures. Threshold-based methods [24] use thresholds of intent importance and intent deviation to filter out unqualified concepts. Naturally, they do not generate the complete concept lattice.

Recommender systems are a natural application of FCA because implicit rating data can be modeled as a formal context. At least three types of methods have been designed. Lattice-based methods [6,52] need to construct the complete concept lattice or a part of it. Entry-level concepts [6], or association rules generated from the lattice [52] are used for recommendation. These methods are relatively slow due to the construction of the lattice. Decomposition-based methods [12] use boolean matrix factorization techniques such as GreConD [3,13] to obtain user and item subspaces, and then use the user subspace to calculate the k-nearest neighbors. These methods require a heuristic algorithm to deal with the NP-hard problem. Concept-based methods [11] use efficient algorithms such as D-miner [4] to construct a set of concepts that meet the extent and intent size constraints, and then mine association rules for recommendation. Compared with mining association rules from the original context, these methods focus on part of the data each time. Therefore, the efficiency and quality of the recommendation algorithm are improved. However, these three types of methods do not directly take advantages of the information embedded in each concept for recommendation.

Three-way decision and granular computing provide new solutions to recommender systems. Three-way decision [43,41,42] is used to deal with the situation where there are three possible decisions, namely accept, reject and wait-and-see. Recommender systems based on three-way decision obtain lower cost by providing the promotion action [48], or higher accuracy by providing the delayed decision [29,49]. Granular computing is a general computational theory for information processing and affected by granular structure. Based on the multi-level structure of information, recommender systems use different granular information for recommendation [25,45]. Sequential three-way decision [41,43,50] is the integration of three-way decisions and granular computing. The model makes three-way recommendations at each granularity level and moves delayed decision to the next finer granularity level [25,44].

In this paper, we propose an efficient heuristic method to construct a concept set for recommendation. Our method takes advantage of the approximation ability of the concept set to formal context information. Therefore, it focuses on mining a high-quality concept set. Fig. 1(a)-(c) show recommendations based on the concept lattice, and (d)-(f) show the recommendation process based on a concept set.

The new scheme consists of two stages. The first stage is the concept set construction, which generates a series of concepts with big intent and extent while satisfying their coverage of the entire set of users. With N users and M items, the time complexity of the direct approach to the construction of all concepts is $O(N2^M)$ [7]. Some enhanced approaches (see, e.g., [35,20,9]) decrease it using some pruning techniques. However, they are still exponential in the worst case. In contrast, our heuristic approach takes only $O((N+M)NM)$ time, which is very efficient. The second stage is concept-based

Table 1
An exemplary formal context.

$\boldsymbol{O}$	$\boldsymbol{A}$							
	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7
u_0	1	0	0	0	1	0	0	1
u_1	1	0	1	1	1	0	1	0
u_2	0	1	0	1	1	0	1	1
u_3	0	1	0	1	1	1	1	0
u_4	1	0	0	0	1	1	0	1
u_5	1	1	1	0	1	1	0	0
u_6	0	1	1	1	0	0	0	1
u_7	1	1	0	1	1	0	0	1
u_8	0	0	1	1	1	0	1	0
u_9	1	0	0	1	0	0	0	1

recommendation. Since a concept is a pair consisting of a set of users and a set of items, the users in the concept can be considered as a group. These users share the same preferences on these items. So the same other items are recommended to users by aggregating the preferences of members in the concept. The recommendations of multiple concepts are combined for the user. As each concept is independent in the process of recommendation, the approach is also very efficient.

Experiments were conducted on sixteen datasets including sampled ones. The biggest dataset (MovieLens-1M) has 6,040 users and 3,952 items. Results show that: 1) Our algorithm was significantly more efficient than the classical algorithm [7]. 2) Our algorithm only generated a small number of concepts. 3) The recommendation performance was significantly better than the conceptual neighborhood based algorithm [5], slightly better than matrix factorization [14], and comparable to k-nearest neighbors (kNN), GreConD-kNN (the combination of GreConD and kNN as suggested by [27]) and item-based collaborative filtering (IBCF). Especially in datasets with low sparsity, the recommendation quality of our algorithm is significantly higher than the other four algorithms.

The rest of the paper is organized as follows. Section 2 first introduces the preliminary knowledge related to this research. Section 3 presents and analyzes some key issues of this study. Section 4 describes the designs of the corresponding algorithms to handle these issues. Section 5 compares our algorithm with four existing algorithms. Section 6 concludes and points out some further works.

2. Preliminaries

In this section, we first introduce the preliminary knowledge of FCA. Then we introduce FCA-based recommendation.

2.1. Formal concept analysis

Formal context, concept, partial order and concept lattice are standard terms in FCA. Here we redefine them to suit recommender systems.

Definition 1. [36] *(Formal context)* A *formal context* is a triple $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$, where $\boldsymbol{O}$ is a set of users, $\boldsymbol{A}$ is a set of items, and $\boldsymbol{R} \subseteq \boldsymbol{O} \times \boldsymbol{A}$ is a binary relation.

If $(u, m) \in \boldsymbol{R}$, we say that user u has rated the item m. We also let $r(u, m) = 1$ to denote that $(u, m) \in \boldsymbol{R}$, and $r(u, m) = 0$ otherwise. Table 1 lists an exemplary formal context representing a movie rating data with 10 users and 8 movies.

Let $\boldsymbol{E} \subseteq \boldsymbol{O}$ and $\boldsymbol{I} \subseteq \boldsymbol{A}$ be a user group and an item group, respectively. A pair of dual operators are defined as

$$\boldsymbol{E}^{\Box} = \{m \in \boldsymbol{A} | \forall u \in \boldsymbol{E}, r(u, m) = 1\}, \tag{1}$$

$$\boldsymbol{I}^{\Diamond} = \{u \in \boldsymbol{O} | \forall m \in \boldsymbol{I}, r(u, m) = 1\}. \tag{2}$$

Definition 2. [36] *(Concept)* A pair $C = (\boldsymbol{E}, \boldsymbol{I})$ is called a *concept* of $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$ iff $\boldsymbol{E} \subseteq \boldsymbol{O}$, $\boldsymbol{I} \subseteq \boldsymbol{A}$, $\boldsymbol{E}^{\Box} = \boldsymbol{I}$, and $\boldsymbol{I}^{\Diamond} = \boldsymbol{E}$.

$E(C) = \boldsymbol{E}$ is the *extent* of the concept, and $I(C) = \boldsymbol{I}$ is the *intent* of the concept. For example, $C = (\{u_1, u_7, u_9\}, \{m_0, m_3\})$ is a concept for the formal context depicted in Table 1. $\boldsymbol{E} = \{u_1, u_7, u_9\}$ is the extent and $\boldsymbol{I} = \{m_0, m_3\}$ is the intent. When the table has no full rows and no full columns, there are two special concepts $C_1 = (\boldsymbol{O}, \emptyset)$ and $C_2 = (\emptyset, \boldsymbol{A})$.

Definition 3. [36] *(Subconcept)* Let $C_1 = (\boldsymbol{E}_1, \boldsymbol{I}_1)$ and $C_2 = (\boldsymbol{E}_2, \boldsymbol{I}_2)$ be two concepts of $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$. The concept C_1 is called a *subconcept* of C_2 (denoted by $C_1 \leq C_2$) iff $\boldsymbol{E}_1 \subseteq \boldsymbol{E}_2$.

When $C_1 \leq C_2$, C_2 is also called a *superconcept* of C_1. Naturally, in some cases, neither $C_1 \leq C_2$ nor $C_2 \leq C_1$ is true. Therefore $\leq$ defines a *partial order* between concepts.

The hierarchy of all concepts given by the subconcept-superconcept relation is called the *concept lattice*. The concept lattice generated from Table 1 has 40 concepts. For real data with thousands of users and items, there are millions of concepts.

The definition of sparsity, which expresses the amount of information known about a formal context, is as follows.

Definition 4. *(Sparsity)* The *sparsity* of a formal context $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$ is given by

$$sp(T) = \frac{|\boldsymbol{R}|}{|\boldsymbol{O}| \times |\boldsymbol{A}|}, \tag{3}$$

where $|\cdot|$ represents the cardinality of the set.

For example, the sparsity of Table 1 is 0.5375. The sparsity of real data is often small, e.g., 0.0418 for the MovieLens-1M dataset.

2.2. FCA-based recommendation

There are at least three types of FCA recommended methods, namely lattice-based, decomposition-based and concept-based methods. Lattice-based [6] methods take advantage of the lattice structure, especially the superconcept-subconcept relationship. The EN-CR [6] is a lattice-based method based on an entry-level concept lattice. The entry-level concept is unique for which u/m is a member of the extent/intent and u/m is not a member of the extent/intent of any subconcept/superconcept. EN-CR walks the lattice, level-by-level, up from the target user's entry-level concept to the supremum to find neighbor users. The recommendation is determined by the preferences of neighbor users.

Decomposition-based methods [12] obtain user and item subspaces using boolean matrix factorization techniques such as GreConD. Formally, let $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$ be the formal context, and $\mathcal{F} = \{(E_k, I_k)\}_{k=1}^{K}$ be a set of concept. Let further $A_{\mathcal{F}}$ be an $N \times K$ matrix where $(A_{\mathcal{F}})_{ij} = 1$ if $u_i \in E_j$ and 0 otherwise. Similarly, let $B_{\mathcal{F}}$ be a $K \times M$ matrix where $(B_{\mathcal{F}})_{ij} = 1$ if $m_j \in I_i$ and 0 otherwise. According to [3], one can always construct $\mathcal{F}$ such that

$$\boldsymbol{R} = A_{\mathcal{F}} \times B_{\mathcal{F}}. \tag{4}$$

Since finding the minimal $\mathcal{F}$ is NP-hard, a heuristic algorithm is often required. With the user subspace $A_{\mathcal{F}}$, one can find the k-nearest neighbors of users for recommendation.

Concept-based methods [11] use efficient algorithms such as D-miner [4] to construct a set of concepts that meet the extent and intent size constraints, and then mine association rules for recommendation. Compared with mining association rules from the original context, these methods focus on part of the data each time. Therefore, the efficiency and quality of the recommendation algorithm are improved.

3. Problem decomposition and analysis

The aim of a recommender system is to provide users with accurate recommendations based on existing information. Given a rating system represented by a formal context, the recommendation problem can be stated as follows.

Problem 1. Formal context based recommendation
Input: A formal context $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$, the training set $\boldsymbol{R}_r \subset \boldsymbol{R}$, the testing set $\boldsymbol{R}_t = \boldsymbol{R} - \boldsymbol{R}_r$;
Output: Recommendations $\boldsymbol{L} \subset \boldsymbol{R} \setminus \boldsymbol{R}_r$;
Optimization objective: $\max F1(\boldsymbol{L}, \boldsymbol{R}_t)$.

The inputs include the formal context, the training and testing sets. The training set is randomly selected 80% from the formal context, and the rest is the testing set. As a standard of the training-stage scenario, $\boldsymbol{R}_r$ is used to construct concepts. $\boldsymbol{R}_t$ is employed to evaluate the performance of the recommendation at the testing stage.

The output is the recommendations to all users. Naturally, there should be no intersection between recommended results and known ratings, hence $\boldsymbol{L} \subset \boldsymbol{R} \setminus \boldsymbol{R}_r$.

The optimization objective should consider both correct and missed recommendations. The precision is given by

$$pre(\boldsymbol{L}, \boldsymbol{R}_t) = \frac{|\boldsymbol{L} \cap \boldsymbol{R}_t|}{|\boldsymbol{L}|}, \tag{5}$$

the recall is given by

$$rec(\boldsymbol{L}, \boldsymbol{R}_t) = \frac{|\boldsymbol{L} \cap \boldsymbol{R}_t|}{|\boldsymbol{R}_t|}, \tag{6}$$

and the $F1$ measure is given by

$$F1(\boldsymbol{L}, \boldsymbol{R}_t) = \frac{2 \times pre(\boldsymbol{L}, \boldsymbol{R}_t) \times rec(\boldsymbol{L}, \boldsymbol{R}_t)}{pre(\boldsymbol{L}, \boldsymbol{R}_t) + rec(\boldsymbol{L}, \boldsymbol{R}_t)}. \tag{7}$$

Here, a concept is a pair consisting of a set of users and a set of items. These users share the same preferences on these items. From the perspective of CF, they may also share the same preferences on some other items. Hence we will decompose Problem 1 into two sub-problems, which will be illustrated in the following two subsections.

3.1. Minimal concept set construction

Constructing a set of concepts representing the formal context is the first sub-problem, as stated follows.

Problem 2. Minimal concept set construction
Input: The training formal context $T_r = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R}_r)$, the intent threshold α;
Output: Concept set $\boldsymbol{C}$;
Constraints:

1) $\bigcup_{(\boldsymbol{E},\boldsymbol{I})\in \boldsymbol{C}} \boldsymbol{E} = \boldsymbol{O}$;
2) $\forall(\boldsymbol{E}, \boldsymbol{I}) \in \boldsymbol{C}, |\boldsymbol{I}| \geq \alpha$;

Optimization objective: $\min |\boldsymbol{C}|$.

The inputs include the training data $T_r = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R_r})$ and the user-specified intent threshold α. We require that α be no greater than the minimum number of movies rated by any user. Its setting will be discussed in the experimental section. The output is a concept set. A user/item may be included in different concepts. In fact, a user may share some preferences with a group of users in some aspects and other preferences with other user groups. This phenomenon is also valid for items.

The first constraint indicates that the concept set covers all users of T. It guarantees that each user belongs to at least one concept. The second constraint indicates that the intent scale of the concept should not be less than α. From the perspective of the recommender system, recommendations only make sense when there are enough common items among users.

The optimization objective is to minimize the number of concepts. The goal is to obtain a more concise representation to improve the generalization of the model.

Now we analyze the complexity of Problem 2. Let $\boldsymbol{C}_\alpha$ be the set of all concepts with intent no less than α. For simplicity, we do not consider the time complexity of constructing this set. Since each user has rated at least α movie, we have $\cup_{(\boldsymbol{E},\boldsymbol{I})\in \boldsymbol{C}_\alpha} \boldsymbol{E} = \boldsymbol{O}$. In addition, due to the uncertainty of user behavior, $\boldsymbol{E}$ can be any non-empty subset of $\boldsymbol{O}$. Now we need to handle the classic set covering problem (SCP), which is NP-complete. Because a partial case of the Problem 2 is NP-complete, the problem itself is NP-hard.

The key issue of Problem 2 is how to build high-quality concepts. We define an indicator to measure the quality of concept.

Definition 5. *(Area of concept)* The *area* of concept $C = (\boldsymbol{E}, \boldsymbol{I})$ is

$$S(C) = S(\boldsymbol{E}, \boldsymbol{I}) = |\boldsymbol{E}| \times |\boldsymbol{I}|. \tag{8}$$

The area of a concept is determined by its extent and intent. A large extent indicates that the users have many neighbors, while a large intent indicates that the concept has high similarity and strong stability. Consequently, the strong concept is defined as follows.

Definition 6. *(Strong concept)* Let $\mathcal{C}(T)$ be the set of all concepts of a formal context $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$, $\mathcal{C}(T, u, \alpha) = \{C \in \mathcal{C}(T) | u \in E(C), |I(C)| \geq \alpha\}$ be all concepts satisfying the intent threshold α for user u. The *strong concept* for user u is

$$\underset{C\in\mathcal{C}(T,u,\alpha)}{\operatorname{argmax}} \; S(C). \tag{9}$$

Note that the strong concept may not be unique to the user.

3.2. Concept set based recommendation

Concept set based recommendation is the second sub-problem. Before analyzing it, we should first introduce item popularity which is widely applied to various recommendation scenarios especially user cold start. It usually refers to the number of users who rated it. For concept-based recommendation, we are more concerned with the popularity of the item within the concept.

Algorithm 1 Heuristic concept set construction (HCSC).

Input: A formal context $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$, the intent threshold α.
Output: A concept set $\boldsymbol{C}$.

$\boldsymbol{C} \leftarrow \emptyset$, $\boldsymbol{P} \leftarrow \boldsymbol{O}$; //Initialization
while ($\boldsymbol{P} \neq \emptyset$) **do**
  $s = 0$; //Current area
  $\boldsymbol{I} \leftarrow \emptyset$; //The intent of the current concept
  $u^* = \arg\max_{u \in \boldsymbol{P}} |\{p\}^{\Box}|$; //The user who rated most movies
  $m^* = \arg\max_{m \in \{u\}^{\Box}} |\{m\}^{\Diamond}|$; //The most popular item rated by u
  $\boldsymbol{I} = \boldsymbol{I} \cup \{m^*\}$;
  while (true) **do**
    $m^* = \arg\max_{m \in (\{u\}^{\Box} \setminus \boldsymbol{I})} |(\boldsymbol{I} \cup \{m\})^{\Diamond}|$;
    $s' = S((\boldsymbol{I} \cup \{m^*\})^{\Diamond}, \boldsymbol{I} \cup \{m^*\})$; //New area
    if ($(s' > s) \vee (|\boldsymbol{I}| + 1 < \alpha)$) **then**
      $\boldsymbol{I} = \boldsymbol{I} \cup \{m^*\}$; //Update
      $s = s'$;
    else
      break;
    end if
  end while
  $C = (\boldsymbol{I}^{\Diamond}, \boldsymbol{I})$; //A new concept
  $\boldsymbol{C} = \boldsymbol{C} \cup \{C\}$; //Add to the concept set
  $\boldsymbol{P} = \boldsymbol{P} - E(C)$; //Remove processed users
end while
return $\boldsymbol{C}$;

Definition 7. *(Local popularity)* In a formal context $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$, the *local popularity* of $m \in \boldsymbol{A}$ wrt. concept $C = (\boldsymbol{E}, \boldsymbol{I})$ is

$$lp(C, m) = \frac{|\{m\}^{\Diamond} \cap \boldsymbol{E}|}{|\boldsymbol{E}|}. \tag{10}$$

The problem of concept-based recommendation is defined as follows.

Problem 3. Concept set based recommendation
Input: The training formal context $T_r = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R}_r)$, the testing formal context $T_t = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R}_t)$, concept set $\boldsymbol{C}$ and recommendation threshold β;
Output: Recommendations $\boldsymbol{L}$;
Constraint: $\forall C \in \boldsymbol{C}, \forall u \in E(C), m \in \boldsymbol{A} - \{u\}^{\Box}, lp(C, m) \geq \beta$;
Optimization objective: $\max F1(\boldsymbol{L}, \boldsymbol{R}_t)$.

The inputs include training and testing sets, a concept set and the recommendation threshold. In this stage, T_r is used to determine which items need to be recommended. T_t is used to evaluate the performance of the recommendation. The concept set is derived from the first stage. The recommendation threshold β is specified by the expert.

The output is the recommendations represented by a matrix. Naturally, existing ratings should not be considered.

The constraint indicates the requirement for recommending items to users. It compares the item's local popularity with the recommendation threshold β. The recommendation for each user is determined by related concepts.

The optimization objective is to maximize the recommendation metrics $F1$, which is calculated through comparing the testing set with the recommendations.

4. The proposed approach

In this section, we first present two algorithms to handle Problems 2 and 3, respectively. Then we analyze their time complexities. Finally we illustrate them with a running example.

4.1. Algorithm description

As discussed earlier, the problem of finding the optimal solution of Problem 2 is NP-hard. Hence we need to design a heuristic algorithm to handle it. Our main idea is to construct a concept with the greatest possible extent. The purpose is to cover the extent with as few concepts as possible. This construction requires only polynomial time. Since each new concept covers at least one user who has not been covered before, at least N concepts are needed. Consequently, the algorithm is polynomial.

Algorithm 1 lists a heuristic concept set construction (HCSC) algorithm which constructs a set of strong concepts covering all users. While constructing each concept, the heuristic information is the area of the concept. When the intent threshold is reached and the area reaches the maximal, a concept will be generated for the user.

Algorithm 2 Concept set based recommendation (CSBR).

Input: A formal context $T = (\boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R})$, a concept set $\boldsymbol{C}$ and the recommendation threshold $0 < \beta < 1$.
Output: Prediction matrix $\boldsymbol{L} = (l_{ij})_{|\boldsymbol{O}|\times|\boldsymbol{A}|}$.

1: $\boldsymbol{L}_{|\boldsymbol{O}|\times|\boldsymbol{A}|} \leftarrow \mathbf{0}$, $\mathbf{Q}_{|\boldsymbol{O}|\times|\boldsymbol{A}|} \leftarrow \mathbf{0}$; // Initialization
2: **for** (each $u \in \boldsymbol{O}$, $m \in \boldsymbol{A}$) **do**
3:     **for** ($u \in C \in \boldsymbol{C}$) **do**
4:         **if** $((r(u,m) = 0) \wedge lp(C,m) \geq \beta)$ **then**
5:             $q_{um} = q_{um} + 1$; //Record the number of recommended concepts
6:         **end if**
7:     **end for**
8: **end for**
9: **for** (each $u \in \boldsymbol{O}$, $m \in \boldsymbol{A}$) **do**
10:     **if** $(q_{um} \geq 2)$ **then**
11:         $l_{um} = 1$; //Recommend item m to u
12:     **end if**
13: **end for**
14: **return** $\boldsymbol{L}$;

Table 2
Time complexity of Algorithm 1.

Lines	Complexity	Description
Line 2	$O(N)$	User set $\boldsymbol{O}$ has N users.
Line 5	$O(NM)$	Select a user with the most number of items.
Line 6	$O(M)$	Select the most popular item rated by the user.
Lines 8-17	$O((N+M)M)$	Construct a strong concept.
Total	$O(N) \times (O(NM) + O(M) + O((N+M)M)) = O((N+M)NM)$	

Lines 2-21 show the process of building the concept set. Lines 3-4 correspond to the initialization phase. Line 5 chooses a user with the maximal number of items as the representative. Lines 18-19 obtain a new concept and add it to the concept set. Line 20 removes those users included in the extent from the representative set.

Specifically, Lines 6-17 construct a strong concept for user u. Lines 6-7 show the process of selecting the most popular item rated by u as the initial intent. Lines 8-17 scan the user's item set and update the intent. Line 9 selects an item that combines the intent to cover the most users. This item comes from the item set rated by the user but not included in the intent. Line 10 computes the area of the current concept. Lines 11-16 correspond to the heuristic method. For lines 11-14, if the conceptual area increases or the intent size is less than α, they will all be updated.

It should be noted that by considering the area of concept, we are essentially dealing with a more complex problem. When the intents of all concepts are equal, this problem coincides with Problem 2.

We note that HCSC is similar to GreConD [3] in that: 1) both mine a set of concepts from a formal context; and 2) both are heuristic algorithms. However, they have the following differences. First, their optimization objectives are different. HCSC aims to generate a minimal set of concepts covering all objects, while GreConD aims to cover the formal context. Second, their heuristic information is different. HCSC uses the area of concept, while GreConD uses $\{y\}^{\downarrow\uparrow} \oplus y$ where y is an attribute. Therefore, HCSC is beyond a variant of GreConD.

Algorithm 2 is a concept set based recommendation (CSBR) algorithm. Concept set $\boldsymbol{C}$ is constructed by Algorithm 1, so a user is covered by at least one concept. The item recommended to the user needs to satisfy two conditions. On the one hand, an item will only be recommended using a concept if its local popularity in the concept exceeds the recommendation threshold. On the other hand, an item will only be recommended to the user if at least two concepts support it.

Line 1 is the initialization of $\boldsymbol{L}$ and $\boldsymbol{Q}$. Lines 3-7 count the number of times each item is recommended. Line 3 performs calculations on each concept of the user. Lines 4-6 calculate the local popularity of the item within the concept. When the local popularity is greater than β, this concept recommends the item. Lines 9-13 remove items that have only one concept recommendation. Line 14 returns a collection of recommended items for respective users.

4.2. Time complexity analysis

Solving Problem 1 requires two stages. One is to generate a set of concepts, and the other is to make recommendations. Thus, the time complexity is equal to the sum of Algorithms 1 and 2.

Proposition 1. *Let N and M be the number of users and items, respectively. For Algorithm 1, the time complexity is $O((N+M)NM)$.*

Proof. Table 2 lists the time complexity of each step in Algorithm 1.

1) According to Line 20, each time at least one user is removed from $\boldsymbol{P}$. Hence the main loop indicate by Line 2 takes $O(N)$ of time.

Table 3
Time complexity of Algorithm 2.

Lines	Complexity	Description
Line 2	$O(NM)$	Check all user-item pairs.
Line 3	$O(N)$	Traverse all concepts that contain the user.
Lines 4-6	$O(N)$	Compute the local popularity of the item.
Lines 9-13	$O(NM)$	Traverse all items of all users.
Total	$O(NM) \times O(N) \times O(N) + O(NM) = O(N^3M)$	

2) Calculating the number of items for one user takes $O(M)$ of time. Hence finding the user who rated most users takes $O(NM)$ of time.
3) Selecting the most popular item rated by the user as the intent takes $O(M)$ of time.
4) Obtaining the intent of a strong concept takes $O(NM)$ of time. Hence, the time complexity for obtaining a strong concept is $O((N+M)M)$.

Therefore, the time complexity of Algorithm 1 is

$$O(N) \times (O(NM) + O(M) + O((N+M)M)) = O((N+M)NM)). \tag{11}$$

This completes the proof.

Proposition 2. *For Algorithm 2, the time complexity of the recommendation is $O(N^3M)$.*

Proof. Table 3 lists the time complexity of each step in Algorithm 2.

1) Checking all user-item pairs takes $O(NM)$ of time.
2) A user is contained in at most N concepts. Hence checking all concepts that contain the user takes $O(N)$ of times.
3) Computing the local popularity of an item requires traversing all users in the concept. Hence the time complexity is $O(N)$.
4) Checking all user-item pairs again takes $O(NM)$ of time.

Therefore, the time complexity of Algorithm 2 is

$$O(M) \times O(N) \times O(N) + O(NM) = O(N^3M). \tag{12}$$

This completes the proof.

4.3. A running example

We show a running example of HCSC and CSBR, respectively. Given the formal context listed in Table 1, let the intent threshold $\alpha = 2$, and recommendation threshold $\beta = 0.5$. Fig. 2 illustrates the process of the algorithms.

In HCSC, we select the user with the largest number of items as the representative and construct a concept for it. Through the computation in Step 1, we first take u_1 as a representative. Then we construct a strong concept for user u_1 in Step 2. User u_1 purchased five items which are m_0, m_2, m_3, m_4 and m_6. Among them, m_4 is the most popular item with 8 users. Hence m_4 is the first item to be added into the intent $\boldsymbol{I}$. At the same time, the corresponding user set $\{m_4\}^\diamond = \{u_0, u_1, u_2, u_3, u_4, u_5, u_7, u_8\}$, and the area $S(\{m_4\}^\diamond, \{m_4\}) = 8$. Then, we combine the remaining items of u_1 with the intent. We can find the combination of m_0 and m_4 with the most users. Adding m_0 into the intent, the updated extent $\{m_0, m_4\}^\diamond = \{u_0, u_1, u_4, u_5, u_7\}$. At this time, the alternative concept area is 10, which is larger than the previous area. Hence, according to the above selection strategy, we continue to construct concept on the basis of $(\{u_0, u_1, u_4, u_5, u_7\}, \{m_0, m_4\})$. Finally, a strong concept $C_0 = (\{u_0, u_1, u_4, u_5, u_7\}, \{m_0, m_4\})$ is obtained. Repeating Step 2, we can get a set of strong concepts covering all users.

In CSBR, we first connect each user with their concept. Then we calculate the popularity of the item. When the local popularity of item is no less than β, it would be recommended. For example, the local popularity of m_2 in concept C_0 is 0.4, which is greater than β. Therefore, concept C_0 supports the recommendation of item m_2. Through the above calculation, we can get the number of supporting concepts for each item. If this number is less than 2, the recommendation is discarded. For example, the number of supporting concepts for m_2 and m_5 are 2 and 1, respectively. Hence only m_2 is recommended to user u_7.

5. Experiments

In this section, we conduct a series of experiments to address the following questions:

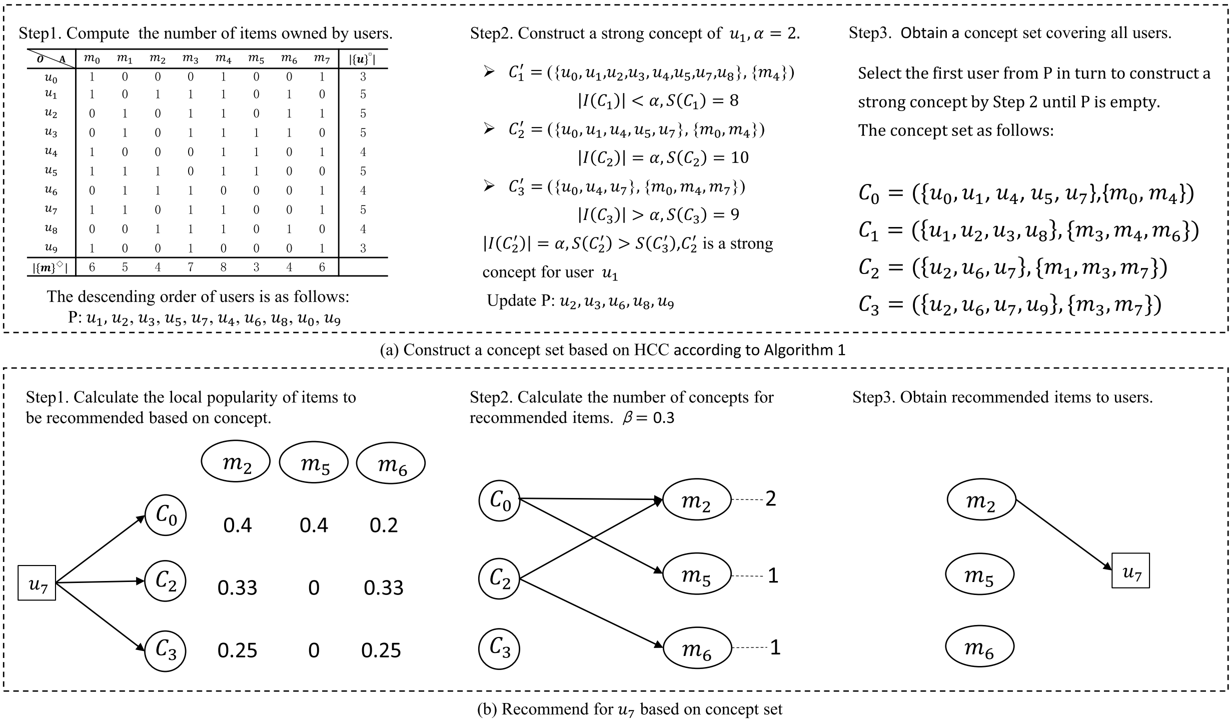


Fig. 2. A running example of our algorithm.

Table 4
Datasets information.

Dataset	Users	Items	Sparsity
MovieLens1	200	420	0.0514
MovieLens2	200	420	0.0725
MovieLens3	200	420	0.0915
MovieLens-100K	943	1,682	0.0630
FilmTrust	1,508	2,071	0.0113
EachMovie-2ku	2,000	1,648	0.0114
EachMovie-3ku	3,000	1,648	0.0117
MovieLens-1M	6,040	3,952	0.0418
Jester1	2,000	100	0.0750
Jester2	2,000	100	0.1000
Jester3	2,000	100	0.1250
Jester4	2,000	100	0.1500
Jester5	2,000	100	0.1750
Jester6	2,000	100	0.2000
Jester7	2,000	100	0.2250
Jester8	2,000	100	0.2500

1) Does HCSC reduce the number of concepts and speed up the concept construction process?
2) Is there an optimal setting of β on any dataset?
3) Is the concept set appropriate for recommendation?
4) Can CSBR improve the quality of recommendation?

5.1. Datasets

Table 4 summarizes 8 movie recommendation datasets and 8 joke datasets, including sampled ones, used in our experiments. They are divided into two types depending on their size. MovieLens1 to MovieLens3 are small datasets with 200 users and 420 items. They are used in comparison experiments with two classical concept lattice-based algorithms. The remaining datasets have more users and items. EachMovie-2ku and EachMovie-3ku are randomly selected from EachMovie. FilmTrust, MovieLens-100K and MovieLens-1M are existing movie recommendation datasets. These five datasets are used for comparison with the collaborative filtering algorithms. Jester1 through Jester8 are sampled from the Jester dataset. They have the same number of users and items, and have different data sparsity.

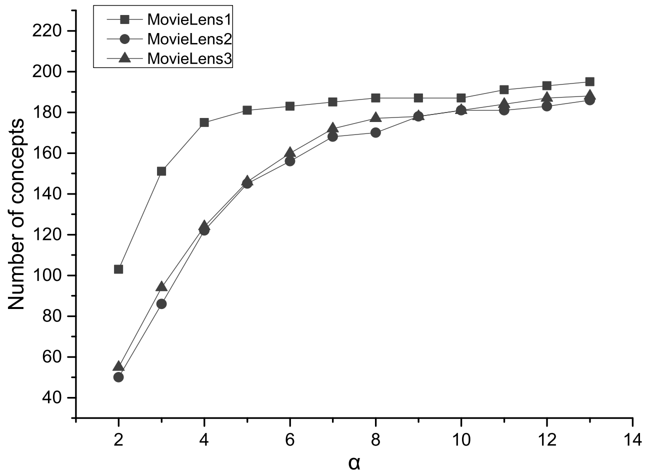


Fig. 3. Trend in the number of concepts related to α.

Table 5
Comparison with ICFL and CbO in concept construction.

Dataset	Number of concepts			Runtime		
	HCSC	ICFL	CbO	HCSC	ICFL	CbO
MovieLens1	103	34,503	34,503	1 s	2 min	8 min
MovieLens2	50	269,893	269,893	1 s	161 min	66 min
MovieLens3	55	1,365,753	1,365,753	1 s	24+h	235 min

5.2. Concept set construction

In experiment, we first observe the effect of the intent threshold α on concept construction. Then we compare HCSC with ICFL [7] and CbO [19] in terms of the number of concepts and the runtime. Fig. 3 illustrates the trend in the number of concepts related to α. Here we observe that the number of concepts increases with the increase in α. This trend is obvious when α is relatively small. However, when α is greater than 8, the number of concepts becomes relatively stable. For datasets of the same size, sparsity also affects the number of concepts. The sparsest dataset MovieLens1 has the largest number of concepts.

Table 5 shows the number of concepts and the runtime of HCSC, ICFL and CbO. Comparative experiments are performed on MovieLens1 to MovieLens3. The intent threshold α is 2. The results show that HCSC generates significantly fewer concepts than ICFL and CbO.

5.3. Recommendation

In this subsection, we first analyze the impact of the recommendation threshold for CSBR. Then we compare the efficiency of CSBR and LBRA. Finally, we compare the recommendation quality of CSBR and four collaborative filtering methods.

5.3.1. Impact of the recommendation threshold

Fig. 4 shows the impact of the recommendation threshold on the performance of CSBR. The performance indicator is F1, and the recommendation threshold β ranges from 0.1 to 0.7. There is a tradeoff while setting β. In general, CSBR performs best when β is 0.4 or 0.5. Specifically, when $\beta = 0.5$, the best results were obtained in 2 out of the 3 experiments. Therefore, we will let $\beta = 0.5$ in the following experiments.

5.3.2. Comparison with a lattice-based algorithm

The lattice-based recommendation algorithm (LBRA) first constructs a complete concept lattice using ICFL. Then, the user's candidate item set is obtained through the union of the intent of these concepts. Finally, according to the similarity between the items in the candidate item set and the target user, the top k items with highest similarity to the target user are recommended. It should be noted that the concept lattice constructed by LBRA is the same as the CbO algorithm.

Three sampled datasets are employed, including MovieLens1, MovieLens2 and MovieLens3. Three performance indicators are used, including Precision, Recall and F1. According to the above experimental results, the recommended thresholds of the three datasets are 0.4, 0.5 and 0.5 respectively. LBRA uses Jaccard similarity to find neighbors.

Fig. 5(a) shows the precision comparison. For three sampled datasets, the CSBR obtains the best precision on two datasets, MovieLens1 and MovieLens2. Fig. 5(b) shows the recall comparison. CSBR performs well in MovieLens1 and Movie-

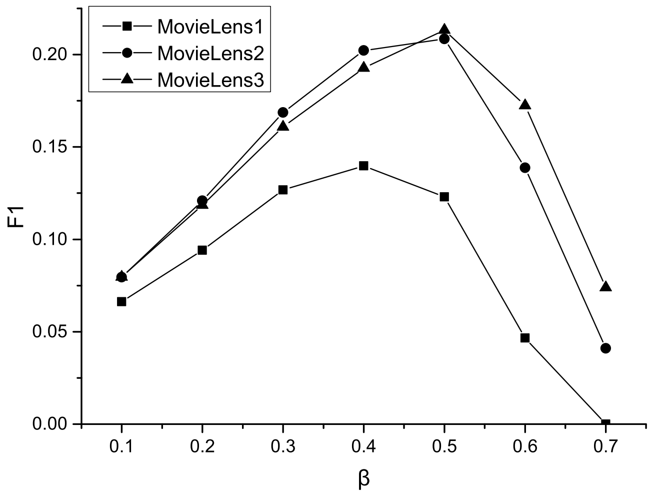
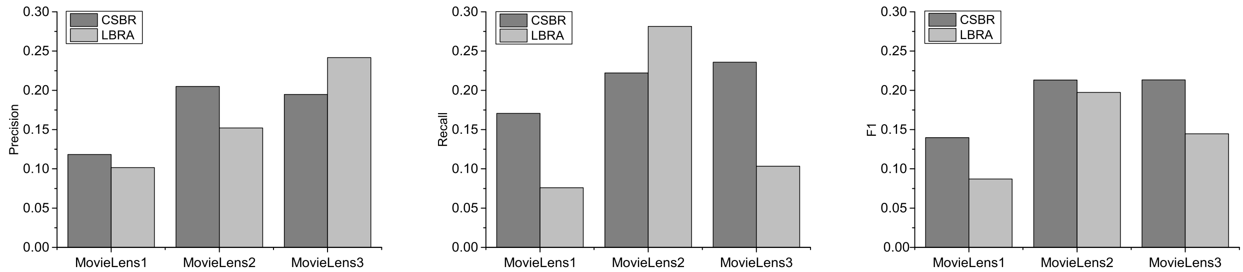


Fig. 4. Impact of the recommendation threshold β.



(a) Precision (b) Recall (c) F1

Fig. 5. Comparison with LBRA.

Lens3. In Movielens1 and Movielens3, the recall for CSBR is more than twice as high as for LBRA. Fig. 5(c) shows the F1 comparison. CSBR is higher than LBRA in all datasets. Combining the experimental results in Table 5 and Fig. 5, it can be seen that CSBR not only has higher time efficiency but also better recommendation effect than LBRA.

5.3.3. Comparison with collaborative filtering methods

CSBR is compared with four collaborative filtering algorithms, including matrix factorization, kNN, GreConD-kNN and IBCF. Matrix factorization implements recommendation by matching the latent factors of users and items. kNN is a recommendation method for merging the preferences of k nearest neighbors. GreConD-kNN utilizes the user subspace and the similarity between users to implement recommendation. In contract, IBCF takes advantages of the similarity between items. For matrix factorization, we adopt the best settings given in the experiments. For kNN, GreConD-kNN and IBCF, we use Jaccard for user or item distance calculation. Table 6 shows the experimental results of CSBR compared with other algorithms. In the comparative experiment with matrix factorization, CSBR has a better recommendation effect. In the precision comparison, CSBR is better than matrix factorization in four of the five datasets. This phenomenon is particularly obvious in the FilmTrust dataset. In all datasets, the recall and F1 value of CSBR are higher than matrix factorization. There are a few cases in which CSBR is worse than the last three algorithms. For example, the last three algorithms have a better F1 than CSBR in MovieLens-100K and MovieLens-1M datasets. In the remaining datasets, the performance of CSBR is better than them. For FilmTrust, the F1 of CSBR reaches 0.5445, which is the highest value among other algorithms.

Fig. 6 compares the performance of five algorithms under different data sparsity. The experiment is conducted on 8 datasets, these data sets have the same number of users and items, but the data sparsity increases sequentially. In 8 sampled datasets, all five algorithms choose the optimal F1 for comparison. It can be observed from Fig. 6 that as the sparsity of the dataset increases, the F1 values of all algorithms also increase. The growth rate of CSBR is much higher than that of matrix factorization, GreConD-kNN and IBCF, which is similar to that of kNN. It is worth noting that when the data sparsity is low, the F1 of CSBR is higher than that of other algorithms. In the 8 datasets participating in the experiment, CSBR performs best when the sparsity is lower than 0.175. Therefore, compared with other algorithms, CSBR is more suitable for datasets with lower sparsity.

Table 6
Comparison with four algorithms. The best result is highlighted in boldface.

Dataset	Evaluation index	CSBR	MF	*k*NN	GreConD-*k*NN	IBCF
MovieLens-100K	precision	0.2099	0.2254	0.1977	0.1976	**0.2600**
	recall	0.2842	0.2298	**0.3473**	**0.3473**	0.2733
	F1	0.2414	0.2276	0.2520	0.2519	**0.2665**
FilmTrust	precision	0.5590	0.1244	0.3538	0.3953	**0.8730**
	recall	**0.5307**	0.1640	0.3533	0.3502	0.3798
	F1	**0.5445**	0.1415	0.3535	0.3714	0.5294
EachMovie-2ku	precision	0.2552	0.2526	**0.3006**	0.2975	0.2202
	recall	**0.3114**	0.2662	0.1754	0.1766	0.2809
	F1	**0.2805**	0.2593	0.2215	0.2216	0.2790
EachMovie-3ku	precision	0.2564	0.2554	**0.2985**	0.2982	0.2212
	recall	0.3482	0.2793	0.1766	0.1770	**0.4153**
	F1	**0.2953**	0.2668	0.2219	0.2222	0.2286
MovieLens-1M	precision	**0.2076**	0.1227	0.1713	0.1713	0.1517
	recall	0.2282	0.1285	0.3531	0.3531	**0.3941**
	F1	0.2174	0.1256	**0.2307**	**0.2307**	0.2191

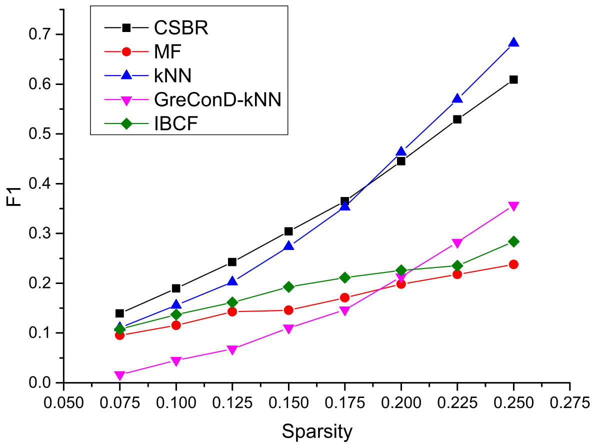


Fig. 6. Performance comparison under different data sparsity.

5.4. Discussions

Now we can answer the questions proposed at the beginning of this section.

1) HCSC reduces the number of concepts and shortens the time for concept construction. This is validated by Table 5. Through analysis, the number of concepts constructed by the HCSC algorithm does not exceed the number of users.
2) There is no optimal setting for β that is valid for any dataset. It is related to the size and sparsity of the dataset. This is validated by Fig. 3. The optimal setting for the threshold can be obtained by experiment or empirical value.
3) The concept set is applicable to recommendation. The three recommendation methods based on HCSC are close to, or even better than, the recommendation based on the concept lattice. This is validated by Table 5 and Fig. 5. Our algorithms are more efficient and accurate than LBRA. This demonstrates that it is not necessary to create the concept lattice for recommender systems.
4) Our approach can improve the quality of recommendation. Compared with other collaborative recommendation algorithms, CSBR has better performance on some datasets, especially in datasets with lower sparsity. This is validated by Table 6 and Fig. 6.

6. Conclusions and further works

In our study, we have proposed a heuristic concept set construction for recommendation. The heuristic approach is designed to construct a high-quality concept set from the formal context. We combine the local popularity and similarity of the item to improve the recommendation effect. In our study, we have proposed a heuristic concept set construction for recommendation, which contains concept construction and recommendation. A heuristic approach is designed to construct a concept set to improve the efficiency. For the recommendation, the local popularity of an item and item similarity are both combined to further improve the recommendation effect. With a collaborative filtering and matrix factorization baseline, the proposed method takes less time and performs better in recommendation. Experimental results show that the proposed method is more accurate than kNN and matrix factorization baseline. This study is innovative for the application of FCA in the field of recommender systems.

The following research topics deserve further investigation:

1) The rating information should be considered when constructing a concept set. Currently, the algorithm only considers whether the user has rated the item. It loses a lot of specific information in the dataset.
2) The intent of the concept should be applied to the recommendation. It is an important part of the concept, it represents the common preferences of users in the extent. Its application can promote personalized recommendations.
3) The heuristic information should be enriched. In current algorithm, the heuristic information is used to mine high-quality concepts applied to the recommendation. According to different application scenarios, different heuristic information can be set to obtain the desired concept.

To sum up, this paper proposes a comprehensive algorithmic framework that can be enriched in the future. We hope this work opens up new doors for applications of formal context analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the Central Government Funds of Guiding Local Scientific and Technological Development (No. 2021ZYD0003), the National Natural Science Foundation of China (Nos. 62006200, 61976245), the Sichuan Province Youth Science and Technology Innovation Team (No. 2019JDTD0017), and the Fundamental Research Funds for the Central Universities (No. XDJK2019B029).

References

[1] R. Belohlavek, Similarity relations in concept lattices, J. Log. Comput. 10 (6) (2000) 823–845, https://doi.org/10.1093/logcom/10.6.823.
[2] R. Bělohlávek, V. Vychodil, Reducing the size of fuzzy concept lattices by hedges, in: The 14th IEEE International Conference on Fuzzy Systems, IEEE, 2005.
[3] R. Belohlavek, V. Vychodil, Discovery of optimal factors in binary data via a novel method of matrix decomposition, J. Comput. Syst. Sci. 76 (1) (2010) 3–20, https://doi.org/10.1016/j.jcss.2009.05.002.
[4] J. Besson, C. Robardet, J. Boulicaut, S. Rome, Constraint-based bi-set mining for biologically relevant pattern discovery in microarray data, Intell. Data Anal. J. 9 (1) (2005) 59–82, https://doi.org/10.3233/IDA-2005-9105.
[5] H.-W. Chen, L.-M. Wang, Z. Zhang, Top-N recommendation algorithm based on conceptual neighborhood, J. Chin. Comput. Syst. 38 (11) (2017) 2553–2559, https://doi.org/10.3969/j.issn.1000-1220.2017.11.025.
[6] P. du Boucher-Ryan, D. Bridge, Collaborative recommending using formal concept analysis, Knowl.-Based Syst. 19 (5) (2006) 309–315, https://doi.org/10.1016/j.knosys.2005.11.017.
[7] R. Godin, R. Missaoui, H. Alaoui, Incremental concept formation algorithms based on Galois (concept) lattices, Comput. Intell. 11 (2) (1995) 246–267, https://doi.org/10.1111/j.1467-8640.1995.tb00031.x.
[8] P.A. Grigoriev, S.A. Yevtushenko, QuDa: applying formal concept analysis in a data mining environment, in: Concept Lattices, vol. 2961, 2004.
[9] L.-H. Hu, J.-F. Zhang, S.-L. Zhang, A pruning based incremental construction of horizontal partitioned concept lattice, in: Computational Intelligence, vol. 4114, 2006.
[10] Z.-Y. Hu, M.-W. Shao, H. Liu, J.-S. Mi, Cognitive computing and rule extraction in generalized one-sided formal contexts, Cogn. Comput. (2021) 1–21, https://doi.org/10.1007/s12559-021-09868-z.
[11] D.I. Ignatov, S.O. Kuznetsov, Concept-based recommendations for internet advertisement, Comput. Sci. (2009) 157–166, http://arxiv.org/abs/0906.4982v1.
[12] D.I. Ignatov, E. Nenova, N. Konstantinova, A.V. Konstantinov, Boolean matrix factorisation for collaborative filtering: an FCA-based approach, in: International Conference on Artificial Intelligence: Methodology, Systems, and Applications, Springer, 2014.
[13] D.I. Ignatov, A. Yakovleva, On suboptimality of grecond for Boolean matrix factorisation of contranominal scales, in: FCA4AI, 2021.
[14] R. Kannan, M. Ishteva, H. Park, Bounded matrix factorization for recommender system, Knowl. Inf. Syst. 39 (3) (2014) 491–511, https://doi.org/10.1007/s10115-013-0710-2.
[15] M. Kaytoue, S.O. Kuznetsov, A. Napoli, S. Duplessis, Mining gene expression data with pattern structures in formal concept analysis, Inf. Sci. 181 (10) (2011) 1989–2001, https://doi.org/10.1016/j.ins.2010.07.007.
[16] J.F.D. Kengue, P. Valtchev, C.T. Djamegni, A parallel algorithm for lattice construction, in: Formal Concept Analysis, vol. 3403, 2005.
[17] A.C. Kumar, S. Srinivas, Concept lattice reduction using fuzzy k-means clustering, Expert Syst. Appl. 37 (3) (2010) 2696–2704, https://doi.org/10.1016/j.eswa.2009.09.026.
[18] S. Kuznetsov, Machine learning on the basis of formal concept analysis, Autom. Remote Control 62 (10) (2001) 1543–1564, https://doi.org/10.1023/A:1012435612567.
[19] S.O. Kuznetsov, A fast algorithm for computing all intersections of objects from an arbitrary semilattice, Nauchn.-Tekh. Inf. Ser. 2-Inf. Protsessy Sist. (1) (1993) 17–20, https://www.researchgate.net/publication/273759395.
[20] L. Kwuida, R.S. Kuitché, R.E. Temgoua, On the size of ∃-generalized concept lattices, Discrete Appl. Math. 273 (2020) 205–216, https://doi.org/10.1016/j.dam.2019.02.035.

[21] C.-P. Li, J.-H. Li, M. He, Concept lattice compression in incomplete contexts based on K-medoids clustering, Int. J. Mach. Learn. Cybern. 7 (4) (2016) 539–552, https://doi.org/10.1007/s13042-014-0288-3.
[22] J.-H. Li, C.-L. Mei, Y.-J. Lv, Incomplete decision contexts: approximate concept construction, rule acquisition and knowledge reduction, Int. J. Approx. Reason. 54 (1) (2013) 149–165, https://doi.org/10.1016/j.ijar.2012.07.005.
[23] J.-H. Li, C.-L. Mei, W.-H. Xu, Y.-H. Qian, Concept learning via granular computing: a cognitive viewpoint, Inf. Sci. 298 (2015) 447–467, https://doi.org/10.1016/j.ins.2014.12.010.
[24] J.-L. Li, Z.-Y. He, Q.-L. Zhu, An entropy-based weighted concept lattice for merging multi-source geo-ontologies, Entropy 6 (2013) 2303–2318, https://doi.org/10.3390/e15062303.
[25] D. Liu, X. Ye, A matrix factorization based dynamic granularity recommendation with three-way decisions, Knowl.-Based Syst. 191 (2020) 105243, https://doi.org/10.1016/j.knosys.2019.105243.
[26] N. Meddouri, H. Khoufi, M. Maddouri, Parallel learning and classification for rules based on formal concepts, Proc. Comput. Sci. 35 (2014) 358–367, https://doi.org/10.1016/j.procs.2014.08.116.
[27] E. Nenova, D.I. Ignatov, A.V. Konstantinov, An FCA-based Boolean matrix factorisation for collaborative filtering, pp. 1–17, arXiv preprint, https://arxiv.org/abs/1310.4366, 2013.
[28] C.H. Pak, J.H. Kim, M.G. Jong, Describing hierarchy of concept lattice by using matrix, Inf. Sci. 542 (2021) 58–70, https://doi.org/10.1016/j.ins.2020.05.020.
[29] F.-L. Qian, Q.-Q. Min, S. Zhao, X.-Y. Wang, Y.-P. Zhang, Three-way decision collaborative recommendation algorithm based on user reputation, in: Rough Sets (IJCRS'2019), 2019.
[30] P.K. Singh, A. Gani, Fuzzy concept lattice reduction using Shannon entropy and Huffman coding, J. Appl. Non-Class. Log. 25 (2015) 101–119, https://doi.org/10.1080/11663081.2015.1039857.
[31] P.K. Singh, C.A. Kumar, Concept lattice reduction using different subset of attributes as information granules, Granul. Comput. 2 (2017) 159–173, https://doi.org/10.1007/s41066-016-0036-z.
[32] F. Škopljanac-Mačina, B. Blašković, Formal concept analysis - overview and applications, Proc. Eng. 69 (2014) 1258–1267, https://doi.org/10.1016/j.proeng.2014.03.117.
[33] G. Stumme, R. Taouil, Y. Bastide, N. Pasquier, L. Lakhal, Computing iceberg concept lattices with Titanic, Data Knowl. Eng. 42 (2) (2002) 189–222, https://doi.org/10.1016/S0169-023X(02)00057-5.
[34] P. Valtchev, R. Missaoui, R. Godin, Formal concept analysis for knowledge discovery and data mining: the new challenges, in: Concept Lattices, vol. 2961, 2004.
[35] L. Wei, J.-J. Qi, Relation between concept lattice reduction and rough set reduction, Knowl.-Based Syst. 23 (8) (2010) 934–938, https://doi.org/10.1016/j.knosys.2010.07.001.
[36] R. Wille, Restructuring lattice theory: an approach based on hierarchies of concepts, in: Ordered Sets, vol. 83, 1982.
[37] W.-Z. Wu, Y. Leung, J.-S. Mi, Granular computing and knowledge reduction in formal contexts, IEEE Trans. Knowl. Data Eng. 21 (10) (2008) 1461–1474, https://doi.org/10.1109/TKDE.2008.223.
[38] W.-H. Xu, W.-T. Li, Granular computing approach to two-way learning based on formal concept analysis in fuzzy datasets, IEEE Trans. Cybern. 46 (2) (2016) 366–379, https://doi.org/10.1109/TCYB.2014.2361772.
[39] S.-C. Yang, Y.-N. Lu, X.-Y. Jia, W.-W. Li, Constructing three-way concept lattice based on the composite of classical lattices, Int. J. Approx. Reason. 121 (2020) 174–186, https://doi.org/10.1016/j.ijar.2020.03.007.
[40] Y.Y. Yao, Interpreting concept learning in cognitive informatics and granular computing, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 39 (4) (2009) 855–866, https://doi.org/10.1109/TSMCB.2009.2013334.
[41] Y.Y. Yao, Granular computing and sequential three-way decisions, in: Rough Sets and Knowledge Technology (RSKT'2013), 2013.
[42] Y.Y. Yao, Three-way granular computing, rough sets, and formal concept analysis, Int. J. Approx. Reason. 116 (2020) 106–125, https://doi.org/10.1016/j.ijar.2019.11.002.
[43] Y.Y. Yao, X.-F. Deng, Sequential three-way decisions with probabilistic rough sets, in: IEEE 10th International Conference on Cognitive Informatics and Cognitive Computing (ICCI-CC'2011), 2011.
[44] X.-Q. Ye, D. Liu, An interpretable sequential three-way recommendation based on collaborative topic regression, Expert Syst. Appl. 168 (2021) 114454, https://doi.org/10.1016/j.eswa.2020.114454.
[45] X.-Q. Ye, D. Liu, D.-C. Liang, Three-way granular recommendation algorithm based on collaborative filtering, Comput. Sci. 45 (2018) 90–96, http://qikan.cqvip.com/Qikan/Article/Detail?id=674437695.
[46] H. Yu, G.-Y. Wang, Y.Y. Yao, Current research and future perspectives on decision-theoretic rough sets, Jisuanji Xuebao/Chinese J. Comput. 38 (2015) 1628–1639, https://doi.org/10.11897/SP.J.1016.2015.01628.
[47] H. Yu, X.-M. Yang, Industrial big data applications based on three-way decisions, J. Northwest Univ. (Nat. Sci. Ed.) 51 (2021) 505–515, http://qikan.cqvip.com/Qikan/Article/Detail?id=7105095275.
[48] H.-R. Zhang, F. Min, Three-way recommender systems based on random forests, Knowl.-Based Syst. 91 (2016) 275–286, https://doi.org/10.1016/j.knosys.2015.06.019.
[49] X.-X. Zhang, L. Chen, Y. Wang, G.-Y. Wang, Improving incremental nonnegative matrix factorization method for recommendations based on three-way decision making, Cogn. Comput. (2021), https://doi.org/10.1007/s12559-021-09897-8.
[50] Z.-H. Zhang, F. Min, S.-P. Shen, Z.-C. Wen, X.-B. Zhou, Tri-partition state alphabet-based sequential pattern for multivariate time series, Cogn. Comput. (2021) 1–19, https://doi.org/10.1007/s12559-021-09871-4.
[51] H.-L. Zhi, J.-H. Li, Granule description of incomplete data: a cognitive viewpoint, Cogn. Comput. (2021) 1–12, https://link.springer.com/article/10.1007/s12559-021-09918-6.
[52] C.-F. Zou, D.-Q. Zhang, J.-F. Wan, M.M. Hassan, J. Lloret, Using concept lattice for personalized recommendation system design, IEEE Syst. J. 11 (1) (2015) 305–314, https://arxiv.org/abs/0906.4982v1.
[53] L. Zou, K. Pang, X.-Y. Song, N. Kang, X. Liu, A knowledge reduction approach for linguistic concept formal context, Inf. Sci. 524 (2020) 165–183, https://doi.org/10.1016/j.ins.2020.03.002.