User-Specific Bicluster-based Collaborative Filtering: Handling Preference Locality, Sparsity and Subjectivity

MIGUEL G. SILVA, LASIGE and Faculdade de Ciências, Universidade de Lisboa, Portugal
RUI HENRIQUES, INESC-ID and Instituto Superior Técnico, Universidade de Lisboa, Portugal
SARA C. MADEIRA, LASIGE and Faculdade de Ciências, Universidade de Lisboa, Portugal

Collaborative Filtering (CF), the most common approach to build Recommender Systems, became pervasive in our daily lives as consumers of products and services. However, challenges limit the effectiveness of Collaborative Filtering approaches when dealing with recommendation data, mainly due to the diversity and locality of user preferences, structural sparsity of user-item ratings, subjectivity of rating scales, and increasingly high item dimensionality and user bases. To answer some of these challenges, some authors proposed successful approaches combining CF with Biclustering techniques.

This work assesses the effectiveness of Biclustering approaches for CF, comparing the impact of algorithmic choices, and identifies principles for superior Biclustering-based CF. As a result, we propose USBFC, a Biclustering-based CF approach that creates user-specific models from strongly coherent and statistically significant rating patterns, corresponding to subspaces of shared preferences across users. Evaluation on real-world data reveals that USBFC achieves competitive predictive accuracy against state-of-the-art CF methods. Moreover, USBFC successfully suppresses the main shortcomings of the previously proposed state-of-the-art biclustering-based CF by increasing coverage, and co-clustering-based CF by strengthening subspace homogeneity.

CCS Concepts: • Information systems → Collaborative filtering; Clustering; Recommender systems.

Additional Key Words and Phrases: Collaborative Filtering, Biclustering, Pattern Discovery, Subspace Clustering, Recommender Systems

1 INTRODUCTION

Decision making plays a vital role in everyone’s lives. As an attempt to cope with massive range of options, there has been large academic and industry interest in automatically recommending items to individuals since last century. Spotify, Amazon, Netflix, and Facebook are some popular platforms that actively use recommender systems [13]. From e-commerce to online advertisement, these systems are unavoidable in our daily online journeys to suggest items in a personalized way. Collaborative Filtering (CF) approaches, firstly proposed by [19], are currently seen as the widest implemented and most mature of the technologies to build recommender systems. Given a set of observed item ratings, CF aims at estimating unknown preferences based on the assumption that users with similar preferences in the past will yield similar preferences in the future. Despite the role of Collaborative Filtering, significant challenges limit its effectiveness, including the diversity and locality of user preferences, the structural sparsity of user-item ratings, the subjectivity of rating scales, and the increasingly large user and item bases [13, 49].

To address the diversity of user profiles, reduce the dimensionality and minimize rating sparsity, matrix factorization and clustering approaches have been combined within CF approaches for two decades [13]. However, traditional clustering techniques are typically applied to either group users or items separately. In real-world CF scenarios, the preferences of a subset of users is frequently only significantly correlated on a subset of the overall items, and vice versa [47]. Biclustering performs clustering in two dimensions simultaneously, being able to find these local preference patterns that...
correspond to data subspaces (biclusters) [40]. Although biclustering has been originally proposed in biomedical domains [5, 6, 21], it increasingly shows promising results in the recommendation domain [15, 18, 48]. In this context, [18] introduced co-clustering as a tool to scale Collaborative Filtering. Despite efficient, their approach can incur in predictive errors associated with the difficulty in guaranteeing a strong homogeneity in the found sub-spaces (co-clusters). More recently, [48] proposed a novel biclustering-based CF approach that outperformed state-of-the-art rating prediction approaches, however, yielding a small coverage of unknown rating estimates.

To address these challenges, this work proposes a novel CF approach, referred as User-specific Bicluster-based Collaborative Filtering (USBCF), that identifies subspaces of shared preferences to create user-specific small and denser user-item matrices, which are then used to guide the training of traditional CF models. USBCF yields three major contributions of interest:

– state of the art biclustering searches for the discovery of high-coverage solutions of rating patterns with statistical significance guarantees;
– superior matching of user preferences against the found rating subspaces by taking into account both the matching extent and fitness of preferences;
– higher predictive coverage without hampering predictive accuracy by placing primary rating estimates from flexible biclustering structures, followed by secondary rating estimates from checkboard coclustering structures.

The proposed approach is assessed against baseline CF methods and peer biclustering-based CF approaches using reference real-world rating data. The gathered results show that the proposed approach successfully surpasses the limited coverage of state-of-the-art biclustering-based approaches and improves the rating prediction accuracy of traditional memory-based approaches.

The manuscript is organized as follows. Section 2 provides essential background on CF and Biclustering. Section 3 surveys relevant work on biclustering-based CF. Section 4 proposes USBCF, introducing the principles to address current shortcomings. Section 5 assesses USBCF against state-of-the-art peer approaches in real data, discussing its behavior. Finally, concluding remarks and future directions are drawn.

2 BACKGROUND

2.1 Collaborative Filtering

Collaborative Filtering (CF) aims at recommending items to users from available user-item rating data. Direct ratings are submitted by users explicitly, while indirect ratings can be inferred from their interactions with the system. User ratings can be further classified according to its domain – unary (positive only), binary, ordinal or numerical [52]. The set of ratings forms a sparse matrix, often referred to as U-I rating interaction matrix, with its sparsity being defined as the prevalence of unknown ratings.

**Definition 2.1.** Considering a set of $n$ users $\mathcal{U} = \{u_1, ..., u_n\}$, a collection of $m$ items $\mathcal{I} = \{i_1, ..., i_m\}$, user-item rating data $\mathcal{D}$ is a set of preferences $r_{ui}$, each corresponding to a unary, binary, ordinal or numerical rating placed by user $u$ on item $i$.

Given $\mathcal{D}$, collaborative filtering (CF) aims at estimating unknown preferences, $\hat{r}_{ui}$. Collaborative filtering can be reduced to specific users or items of interest, referred as active users or active items.

CF gives rise to Predictive and Recommendation stances. In the Predictive stance, unknown ratings are estimated for a given user and item based on available ratings [29]. As for the recommendation tasks, the system usually generates and provides an ordered list of $n$ items, known as Top-N recommendation list, which is a list including the most relevant/useful items for the user. This
subsequent Recommendation task, often involves dissimilarity criteria among the returned items or between the known and top items [29].

Collaborative Filtering methods are usually divided into two main classes, the memory-based and the model-based [49]. The memory-based algorithms use the user-item system ratings directly to predict ratings for new items. In this context, user-based (item-based) systems predict the rating of an item by a particular user using a simple estimator over the set of ratings of similar users (items). Some challenges limit memory-based CF effectiveness when dealing with recommendation data, mainly due to the vast amounts of data and their sparse nature. Model-based approaches try to overcome the limitations of memory-based CF by learning predictive models from the available data and later use them to predict users’ ratings for new items.

2.2 Clustering

Clustering has been largely considered in Collaborative Filtering [13]. In the context of memory-based CF, clustering has been applied to precompute groups of users (items) in user-based (item-based) approaches, promoting interpretability and testing time efficiency by assessing the similarity of a new (user) against the centroids of the precomputed clusters. Alternatively, clustering has been largely applied to model-based CF, where the task of learning a predictive model from the complete rating data space is reduced to each cluster of users (items) produced from clustering, facilitating the learning task in accordance with group-wise preferences.

Irrespective of the approach, clustering relies on similarity functions to estimate proximity between users (items). Popular examples robust to missing rates include cosine vector similarity and Pearson correlation [13].

Despite the relevance of the clustering-based CF, clustering mislay valuable information because they can only be applied either to the user or item dimension separately, disregarding the rich structure of the other dimension and thus preventing the identification of shared local preferences.

2.3 Biclustering

In real-world applications with large item collections, shared preferences between a group of users (items) are only correlated on a subset of items (users). In this context, computing similarity against the whole dimension neglects the diversity of user preferences along item classes and therefore misses groups of users with correlated preferences on specific subsets of items. Opposed to one-way clustering, biclustering is thus a technique able to cluster users and items simultaneously, producing subspaces (biclusters) with coherent preferences as illustrated in Fig.1.

**Definition 2.2.** Given user-item rating data $\mathcal{D}$, a bicluster $B = (U, I)$ is a subspace given by a subset of users, $U \subseteq \mathcal{U}$, and a subset of items, $I \subseteq \mathcal{I}$. The biclustering task aims to find a set of biclusters $\mathcal{B} = \{B_1, \ldots, B_q\}$, such that each bicluster $B_k$ satisfies specific criteria of homogeneity [24], dissimilarity and statistical significance [28].

Biclusters capture local rating patterns corresponding to correlated preferences for a subset of users and items. The biclustering homogeneity criteria determines the structure, coherence and quality of a biclustering solution. The structure is described by the number, size, shape and position of biclusters. Flexible structures of biclusters are characterized by an arbitrary number of (possibly overlapping) biclusters. The coherence of a bicluster is given by the observed correlation of ratings (coherence assumption) and, in the presence of numerical preferences, the allowed deviation from expectations (coherence strength). Finally, the quality of a bicluster is defined by the type and amount of tolerated noise.
**Definition 2.3.** Given a real-valued dataset, let the elements in a bicluster $r_{ui} \in (U, I)$ have **coherence** across observations given by $r_{ui} = k_i + y_u + \eta_{ui}$, where $k_i$ is the expected rating for item $i$, $y_u$ is the adjustment for user $u$, and $\eta_{ui}$ is the noise factor. A bicluster satisfying a specific coherence strength, $\delta \in \mathbb{R}^+$, has values described by $r_{ui} = k_i + y_u + \eta_{ui}$ and $\eta_{ui} \in [-\delta/2, \delta/2]$.

**Definition 2.4.** Given a bicluster $B=(U,I)$ with coherence in accordance with Def. 2.3, the $y_u$ factors define the coherence assumption: **constant** pattern on rows when $y_u = 0$ and **additive** pattern on rows when $y_u \neq 0$.

The bicluster **pattern** $\varphi_B$ is the set of expected values in the absence of adjustments and noise, $\varphi_B = \{k_i \mid i \in I\}$.

**Fig. 1.** Clustering (a-b) vs. biclustering (c) solutions over ordinal user-item ratings.

**Fig. 2.** Illustrative forms of coherence [40]: a) constant value, b) constant-values on rows, c) constant-values on columns, d) coherent values (additive model), e) coherent values (multiplicative model), f) overall coherent evolution, g) coherent evolution on the rows, h) coherent evolution on the columns.
Commonly pursued forms of coherence are constant values across the subspace, users or items. When considering forms of numerical and ordinal feedback, biclusters can further accommodate additive, multiplicative or order-preserving factors. Fig. 2 illustrates these different types of biclusters.

A bicluster is statistically significant if its probability to occur deviates from expectations (i.e. is unexpectedly low against a null data model).

Let $\mathcal{B}$ be the set of biclusters that satisfy a given homogeneity and statistical significance criteria, $(U, I) \in \mathcal{B}$ is a maximal bicluster iff there is no other bicluster $(U', I')$ such that $U \subseteq U' \land I \subseteq I'$ satisfying the given criteria. Although an optimal biclustering solution is one containing all maximal biclusters, the biclustering formulation can further include dissimilarity criteria to reduce the size of outputs and minimize redundancies.

Even when considering unary rating data, biclustering is a well-established NP-hard task. As a result, many biclustering algorithms use heuristic mechanisms (producing sub-optimal solutions) [40], and place restrictions on the allowed homogeneity criteria.

2.4 Contrasting biclustering with coclustering and dimensionality reduction

The biclustering task is not equivalent to the coclustering task, also termed block clustering. Coclustering is a restrictive form of biclustering requiring that all users and items belong to a subspace (exhaustive condition) and to a single subspace only (exclusive condition), producing a checkboard structure as a result. Although coclustering limits the inherent flexibility of the biclustering task [34], it guarantees that all user-item pairs are included in a single subspace.

Definition 2.5. Given user-item rating data $\mathcal{D}$, coclustering aims to partition users and items, $(\mathcal{U} = \{U_1, ..., U_r\}, \mathcal{I} = \{I_1, ..., I_s\})$, so that subspaces resulting from the intersecting partitions, $\mathcal{U} \times \mathcal{I}$ optimize some homogeneity criteria.

Dimensionality reduction of rating data became popularized after playing a significant role in the Netflix Prize competition [4]. Dimensionality reduction procedures are now pervasive in CF [13] as they offer a means of diminishing the sparsity of user-item rating data by placing data projections into denser spaces, and tackling generalization difficulties of model-based CF approaches caused by the arbitrarily-high dimensionality of item collections. Note, however, that most common forms of dimensionality reduction in CF are based on matrix factorization approaches, whose behavior resembles a restricted variant of coclustering approaches. In this context, they do not yield the inherent flexibility of biclustering stances in reducing the original space into an arbitrarily-high number of statistically significant subspaces capturing coherent preferences.

3 RELATED WORK

This section covers the most notorious contributions in the field of Collaborative Filtering incorporating subspace clustering techniques; from Predictive (section 3.1) to Recommendation (section 3.2) stances.

3.1 Rating prediction approaches

In 2005, George and Merugu first introduced co-clustering as a tool to improve Collaborative Filtering [18]. In this work, they used weighted Bregman co-clustering [3] to group users and items along a checkerboard subspace structure, as firstly introduced by [23]. Their co-clustering CF approach is based on the idea that the input data’s missing ratings can be predicted using a suitable low parameter approximation of the input rating matrix, similar to CF approaches that rely on dimensionality reduction procedures such as singular value decomposition (SVD). The Bregman co-clustering returns non-overlapping subspace clusters covering the whole rating data space, as well as summary
statistics derived from the co-clustering solution, to construct a matrix approximation for the input data matrix. For this particular application, the summary statistics are the rating averages of the users, items, and co-clusters. According to the authors, this permits a more reasonable rating approximation than just considering the average value of the co-cluster corresponding to the value we want to predict, which is plausible since it takes into account the biases of individual users and items. After computing the reconstructed approximate matrix, its values are considered for the rating prediction task. The authors designed incremental and parallel versions of the original co-clustering algorithm to build an efficient real-time CF mechanism, capable of updating the co-clusters as new users and ratings enter the system. The results of the work show that their approach achieves satisfactory accuracy compared with baseline matrix factorization models but at a lower computational cost.

In 2007, [9] suggested a biclustering CF methodology, using a heuristic immune-inspired biclustering technique, denoted BIC-aiNet, that finds biclusters with coherent values. Their methodology is based on two main stages: the biclusters’ generation, and the similarity computation between the users and the resulting biclusters. After their artificial immune-inspired network generates the biclusters, they are used to predict how the active user would rate a specific item. The rating prediction is performed by searching for the biclusters that include the active user and item, and then, calculating the residue of each bicluster through mean-squared residue (MSR) [7],

$$\text{MSR}(B) = \frac{1}{|U||I|} \sum_{u \in U} \sum_{i \in J} (r_{ui} - r_{ui} - r_{ul} - r_{d})^2,$$

(1)

where $|U|$ is the number of users in the bicluster, $|I|$ is the number of items in the bicluster, $r_{ui}$ is the rating of user $u$ assigned to item $i$, $r_{ui}$ and $r_{ul}$ represent the mean value of user $u$ and item $i$, respectively, and $r_{d}$ is the mean value of the entire bicluster. Finally, the bicluster with the smaller residue value is selected, and the average of its movies’ ratings is used as the prediction. This methodology was evaluated in both rating prediction and Top-N recommendation scenarios, achieving better scores than the methods used for comparison, being the constant version of BCF [50], one of those.

[10] also used an immune inspired biclustering algorithm, MOM-aiNET [8] to obtain biclusters used to predict missing ratings for recommendation purposes. Their algorithm generates biclusters with a controlled percentage of missing values. Then, they use a quadratic programming approach to predict those values by trying to minimize the mean-squared residue of the bicluster. This approach is viewed as an improvement of the one developed by [9] (described above), as the latter simply replaces the missing values in a bicluster by the average of the bicluster’s values, not reflecting the trends obtained by coherent biclusters. Whereas, in this approach, the missing values are viewed as variables that should be obtained to minimize the residue.

A more recent work in the direction of biclustering CF was published in 2014 by [38], who proposed a CF algorithm utilizing information-theoretic co-clustering (ITCC) [11]. Their approach (ITCCCF) consists in performing two-way clustering to discover a set of user clusters and a set of item clusters. After that, they compute the preference of a user $u$ for the cocluster containing an item $i$, $B = (U, I, (P_{u,i}))$, for each pair user-item per cluster,

$$P_{u,i} = \frac{|I_u \cap I_i|}{|I_u|},$$

(2)

where $I_u$ are the items that the user $u$ rated and $I_i$ are the items from the cocluster containing item $i$. A matrix of user-cluster preferences is constructed and its values are used to obtain the cosine similarity between each pair of users, $sim_p(u_1, u_2)$. This approach then combines the previous
clustering preference similarity with a rating similarity determined through Person’s correlation coefficient to find the $k$ most similar users. The $k$-nearest users are selected and the User-based CF is used to generate the rating prediction. Moreover, the whole process is also repeated for the item clusters and the Item-based CF is used to generate a new rating prediction. The final rating prediction is a linear combination that fuses the ratings from the user-based and item-based approaches. In order to evaluate their approach, using real-world datasets, the authors compared it with 5 state-of-the-art methods, some already mentioned in this work (UBCF [43], IBCF [46], CBCF [45], BCC [18]), being able to surpass all of them regarding prediction accuracy. Figure 3 provides an overview of how this fusion rating technique works.

In 2017, [33] also proposed a fusion-based approach called Nearest Biclusters Collaborative Filtering with Fusion (NBCFu) to address rating prediction using biclustering. In this approach, the xMOTIFs algorithm [42] is used to generate the biclusters that are viewed as quality neighbors of users and items. After the biclusters’ generation, the authors use the CjacMD measure [41] (Equation 3) to obtain the K-nearest biclusters of the users and items. Considering a subspace of users $U \subseteq \mathcal{U}$, this similarity measure combines Mean Measure of Divergence (MMD) [31], Jaccard similarity, and cosine similarity,

$$\text{sim}(u, U)_{\text{JacMD}} = \text{sim}(u, U)_{\text{cos}} + \text{sim}(u, U)_{\text{Jaccard}} + \text{sim}(u, U)_{\text{MMD}}.$$  

After this step, User-based CF and Item-based CF using cosine similarity are applied to the users/items found in most similar biclusters. Finally, the rating prediction of both approaches is combined through a weighted sum, optimized using gradient descent, to generate a final and more accurate rating prediction. The authors claim that their approach has better accuracy results than some state-of-the-art CF methods, including SVD++ [35]. Figure 4 shows an overview of NBCFu.
Fig. 5. Outline of BBCF rating prediction approach proposed by [48].

weighted sum of the ratings that the remaining users gave to the item in the same bicluster, weighted by their correlation with the active user. The similarity score between an active user \( u \) and another user \( u' \), \( \text{sim}(u, u') \), belonging to the same bicluster is given by the Pearson correlation coefficient of their ratings in the bicluster,

\[
\hat{r}_{ui} = \bar{r}_{ui} + \frac{\sum_{u' \in U} (r_{u'i} - \bar{r}_{u'i}) \cdot \text{sim}(u, u')}{\sum_{u' \in U} |\text{sim}(u, u')|}.
\]  

(4)

This formula can be seen an adaptation of the rating estimates used in User-based CF but restricted to a subspace. Using this prediction approach, the users with the highest positive correlation with the active user, have the most impact on the prediction. The algorithm’s performance was compared against other collaborating filtering techniques, and it performed similarly to the previously mentioned approach, BIC-aiNet [9].

Recently, [48] introduced a new biclustering-based CF technique, BBCF. In this work, the authors use biclustering as a preprocessing step to scale CF approaches. Once the biclustering algorithm is executed, the users are compared with the found biclusters based on the items they have in common,

\[
\text{sim}(u, B_k) = \frac{|I_u \cap I_k|}{|I_k|}.
\]  

(5)

Then, similarities between users and biclusters are weighted with the number of users in the bicluster so that biclusters with more users are privileged,

\[
\text{WF}(u, B_k) = \text{sim}(u, B_k) \times |U_k|.
\]  

(6)

After this step, the K-nearest biclusters per user are merged, creating a larger subspace referred as Nearest Neighbor Setup (NSS). The NSS of each user is viewed as a denser subspace of the U-I matrix, including a subset of users similar to the respective user along a subset of overall items. Finally, a classic CF approach, such as the Item-based, is applied over the personalized NSS of each user to predict the missing ratings. According to the authors, this approach tackles the scalability and sparsity problems that deteriorate the performance of memory-based CF methods, through the usage of the personalized NSSs instead of the entire U-I matrix. The authors evaluated BBCF using QUBIC and Item-based CF as the underlying biclustering and CF approaches, respectively, yielding better scores in most scenarios. However, they also highlight a clear drawback of their methodology. They point out that their approach cannot generate a prediction/recommendation for as many users/items as traditional methods. This handicap occurs because the NSS subspaces used within the CF algorithm may not fully cover the original user-item rating data space.
3.2 Top-N Recommendation Approaches

[51] proposed a neighborhood-based CF algorithm (NBCF) that uses biclustering to improve scalability and accuracy of CF. They also created a similarity measure, defined in Equation 5, to identify biclusters that better reflect users’ preferences. Their approach can be combined with biclustering algorithms that find subspaces with either constant or coherent values. The authors used Bimax and xMOTIFs, respectively. Upon selecting biclusters with overall constant values, we can only discover sets of users and items with identical rating values. Whereas, a coherent pattern allows to find users and items correlated with more complex behaviors, such as users that, for a subset of items, exhibit coherent ratings. After the discovery of biclusters, they measure the similarity of the active user with each bicluster, finding the k-nearest-biclusters to the user. Then, to capture the influence that each item belonging to a bicluster has over the active user, they use Weighted Frequency (WF), as in Equation 6. Finally, Weighted Frequencies scores are summed per item. The final top-N recommended items are the $N$ items with the highest sum of WF. Figure 6 summarizes the process of this top-N recommendation approach. The authors reported that both proposed approaches – constant and coherent biclustering assumptions – achieved better results in terms of accuracy and efficiency than classic CF algorithms such as User-based CF [43], Item-based CF [46], and clustering-based CF (CBCF) [55]. They also concluded that despite being slightly less efficient, the recommendation accuracy when considering coherent biclusters outperforms the baseline approach with constant biclusters.

Still in the context of top-N recommendation tasks, [2] also proposed a biclustering neighborhood-based CF method (BCN). In this work, the authors use properties from the field of Formal Concept Analysis (FCA) [17]. FCA is a specialized form biclustering aiming at discovering dense subspaces from rating data produced from unary or binary preferences [32]. FCA algorithms can be adapted to enumerate and order subspaces, which can then be used to identify neighborhoods of closely related subspaces. Alqadah et al. took advantage of these FCA properties to build neighborhoods for active users. Their approach finds the “smallest” subspace containing the active user $u$, which is the subspace containing the fewest number of users and the greatest number of items. Then, by exploring the subspace neighborhood through FCA properties, it finds similar users to $u$, and appends the items in the neighborhood to a candidate set of items. Finally, the candidate items are ranked by combining a global and a bicluster neighborhood similarity, and the top $N$ items are returned as recommendations. The global distance between a user $u$ and an item $i$, $g(u, i)$, measures the similarity between the user and the item in the entire matrix,

$$g(u, i) = \frac{\sum_{i_k \in \mathcal{L}} \text{Jac}(U_i, U_k)}{|I'|},$$

(7)

where $\text{Jac}(U_i, U_k)$ is the Jaccard index defined over the set of all users who interact with item $i$ and the set of those who interact with item $i_k$, and $I'$ is the set of items belonging to all found subspaces.
In contrast, the local/neighborhood similarity, \( l(u, i) \), captures similarities between locally similar users and items, considering the bicluster similarity [1] using subspaces in which \( i \) occurs. The authors claim that their approach is superior to those computing the biclusters offline since, in BCN, they map a user to a bicluster on demand. However, the BCN framework was designed to work on implicit feedback recommendations, and thus it can only be applied to binary data. Nevertheless, to surpass this limitation, it could be an interesting research stream to study the potential of an FCA adaptation capable of dealing with numerical rating data. For instance, in a very recent thesis, [32] proposed and FCA extension to deal with numerical matrices and discover more flexible types of biclusters which could be potentially relevant for recommendation purposes.

4 USER-SPECIFIC BICLUSTERING-BASED COLLABORATIVE FILTERING

Grounded on the previously surveyed contributions and further empirical evidence, this section introduces a new approach, User-Specific Biclustering-based Collaborative Filtering (USBCF).

The proposed USBCF approach is divided into four major steps. First, biclusters are mined from rating data using biclustering searches that guarantee the exhaustive discovery of flexible structures of statistically significant, and well-defined rating patterns (section 4.1). Second, and inspired by the Singh and Mehotra contributions [48], the found biclusters are used to create larger subspaces personalized for each user (section 4.2). These subspaces are denser and show stronger preference correlation than the original rating data. Third, we introduce a novel matching mechanism between an active user and the enlarged subspaces that considers both the coverage and correlation between the user and the biclusters (section 4.3). By integrating item coverage and correlation profile, USBCF guarantees that the selected subspaces per user are relevant, showing reduced propensity to contain divergent preferences. Fourth, and once the personalized subspace is created for a particular user, a well-established CF algorithm is used to originate a unique predictive model for each system user (section 4.4). Finally, secondary rating estimates from checkboard coclustering structures are further collected to account for missing ratings that may fall outside each personalized subspace.

The methodology can be easily parallelized since the entire process is independent for each user. Figure 7 presents and overview of the USBCF methodology applied over an illustrative sample of user-item ratings and the corresponding user-specific CF model.

4.1 Bicluster discovery in user-item rating data

The first step of the approach consists of mining biclusters within user-item rating data, such that three major properties of interest are observed: i) nearly-exhaustive biclustering searches are employed to guarantee that all potentially relevant biclusters are identified, thus maximizing the coverage of the original rating dataset and, as a result, supporting the prediction of missing ratings; ii) the targeted type of biclusters (Fig. 2) should guarantee that meaningful and well-delineate preference patterns are retrieved, allowing coherent preference variations to take place on different items of a subspace; and iii) the retrieved biclusters should pass statistically significance tests, thus minimizing the susceptibility to use false positive preference patterns (retrieved preference patterns that occur by chance) in the prediction process.

Existing biclustering-based CF approaches generally fail to satisfy some of the properties due to the underlying biclustering algorithmic choices. In this work, we select QUBIC2\(^1\) [53] as the biclustering algorithm. QUBIC2 is an efficient nearly-exhaustive deterministic biclustering algorithm that identifies statistically significant scaling-patterned biclusters with constant-values on the rows.

When applied over an U-I Matrix, QUBIC2 identifies groups of users who have the same preference for the bicluster items. The tolerated noise per bicluster can be further parameterized in QUBIC2.

\(^1\)https://github.com/OSU-BMBL/QUBIC2
Following empirical evidence, we restrict the biclustering search to discover non-noisy maximal biclusters as noise and preference divergences can be accommodated in the subsequent subspace enlarging step (section 4.2). The minimum size of the biclusters is controlled by a parameter $minCols$. Different values of $minCols$ may lead to different biclustering solutions. Instead of setting the $minCols$ parameter to a fixed value, in this work, we created an enriched biclustering solution that aggregates the solutions from running QUBIC2 with different $minCols$ setups. A detailed description of the algorithm can be found in QUBIC’s original works [37, 53].

### 4.2 Biclustering neighborhood formation

The goal of this second step is to create, for each user, a new data space with meaningful and personalized information. To do so, we take advantage of the biclusters found by the biclustering algorithm in the previous module. After the biclusters are generated, we can expect that some of those biclusters represent a user’s preferences better than others. From this perspective, we try
to find a “neighborhood” of biclusters. In other words, a subset of overall found biclusters such that each bicluster in this subset, \( B \in \mathcal{B} \), has a preference pattern, \( \varphi_B \), correlated with the observed preferences for the active user. Once this neighborhood of biclusters is identified, we followed the principles proposed by [48] to create a personalized user-specific dataset.

The procedure to obtain the personalized dataset consists in merging all the users and items of the bicluster neighborhood, corresponding to a full outer join of the subspaces in the neighborhood. For example, considering \( B_3 = (U_3 = \{u_4, u_5\}, I_3 = \{i_3, i_5\}) \) and \( B_6 = (U_6 = \{u_6, u_8\}, I_6 = \{i_4, i_6\}) \) in Figure 7, the union set of users and items of this neighborhood is \( (U_N = \{u_1, u_2, u_6, u_7, u_8\}, I_N = \{i_4, i_5, i_6\}) \). Figure 8 illustrates the process.

Finally, the active user is added to the personalized rating dataset when he is not amongst the set of users of his own space. This step is crucial to accommodate available preferences from the active user and to guarantee that the subsequent application of Collaborative Filtering approaches can generate recommendations for his omissive preferences.

### 4.3 Similarity between individual and subspace preferences

To identify neighborhoods for personalized data creation, the subset of biclusters with preference patterns correlated with the preferences of an active user need to be identified. [51] introduced the concept of similarity between a bicluster and a user. However, their similarity is solely based on the rated items’ interception, thus being unable to guarantee that the preferences of the active user are correlated with the preferences of the users in the biclusters. Understandably, this is an undesirable condition in the presence of binary, ordinal or numeric rating data.

To address this limitation, we propose a new user-bicluster similarity that, besides the rated items’ interception, also considers the values of ratings. Following empirical evidence, this similarity is calculated through the product of two distinct scores which we refer to as **items’ match** \( \text{sim}_{\text{match}} \) and **pattern fit** \( \text{sim}_{\text{fit}} \), respectively.

The **items’ match score** captures the portion of bicluster items which were rated by the user, \( \text{sim}_{\text{match}}(u, B_k) = \frac{|I_u \cap I_k|}{|I_k|} \), for \( u \) and \( B_k \) (Equation 8), where \( I_u \) are the items rated by the user \( u \) and \( I_k \) are the items in bicluster \( B_k \). Its values range between [0,1], with 1 representing the maximum similarity score between a user and bicluster. Understandably, a similarity entirely based on rated items’ is insufficient as it disregards the values of the ratings. For instance, in Figure 9, we have an example of bicluster and an active user that rating all the items in the bicluster. Using similarity criteria solely based on Equation 8, we would yield a maximum similarity. However, the preferences between the active user and the ones from
the given bicluster are divergent, thus the inclusion of such bicluster in the neighborhood would hamper the subsequent CF task.

Given this, we introduce the **pattern fit score**, which is responsible for measuring the resemblance between the user rating pattern and the bicluster preference pattern (Def. 2.4). Given a user \( u \) and a bicluster \( B_k = (U_k, I_k) \),

\[
\text{sim}_{\text{fit}}(u, B_k) = 1 - \frac{\text{RMSE}(\varphi_{B_k}, \varphi_{(u, I_u \cap I_k)})}{r_{\max} - r_{\min}},
\]

where \( \varphi_{B_k} \) is the bicluster pattern (Def. 2.4), \( \varphi_{(u, I_u \cap I_k)} \) is the user preferences along the items \( I_k \) in the bicluster, \( r_{\max} \) \( r_{\min} \) correspond to the maximum (minimum) observed rating, and RMSE corresponds to the root mean squared error. In sum, the pattern fit scores corresponds to the normalized root-mean-squared differences between the preferences defined by the subspace pattern and the ratings of the user.

The similarity score defined in Equation 9 considers a rating error applied to biclusters with a constant coherent pattern. However, this similarity could be easily adapted to different coherence types, such as order-preserving, by adopting a Spearman’s rank correlation coefficient or a Kendall rank correlation coefficient, instead of the root mean squared error, to take into account order preservation in the patterns.

The final similarity between a user and a biclusters is the simple product between the scores previously defined,

\[
\text{sim}(u, B_k) = \text{sim}_{\text{match}} \times \text{sim}_{\text{fit}},
\]

as revealed by empirical evidence from the experimental comparison of weighted sums, \( \sum_i \alpha_i x_i \), and weighted products, \( \prod_i x_i^{\alpha_i} \).

Both **items’ match score** and **pattern fit score** values range between [0,1], so its product will also be in the same range. The product operation further establishes the relevance for both components to yield good levels for a bicluster to be seen as relevant for an active user. In other words, a poor value in one of the components will result in a low similarity score.

A \( \text{minSim} \) threshold is finally considered to filter the relevant biclusters for an active user. Only the biclusters with similarity with the user above the threshold are selected to produce its personalized dataset.

### 4.4 Learning User-Specific Recommendation Models

Once the personalized datasets are created, a Collaborative Filtering algorithm is subsequently trained to create a unique recommendation model for each individual. Although the USBCF approach can be parameterized with any Collaborative Filtering approach, Item-based CF approaches are a particularly suitable option due to the inherent properties of the personalized data spaces.
Despite their inherent simplicity, Item-based CF approaches are one of the form CF approaches and still recognized for producing state-of-the-art rating predictions. USBCF generates smaller and denser U-I matrices which, when allied with these approaches, allow the models to perform the recommendation tasks more effectively.

It is possible that the personalized data spaces do not cover the entire recommendation space. For instance, in Figure 7, the personalized dataset does not contain items \( i_1 \) and \( i_2 \), originating a user-model with limited coverage capability, i.e. missing preferences for the items not included in the personalized data space cannot be estimated using the personalized. These situations occur when there are no shared local preferences containing those items, which poses a particularly difficult predictive setting irrespective of the selected CF approach.

To address this challenge, it is possible to find subspaces with lower homogeneity, specifically an exhaustive partitioning of the original space into a set of subspaces that cover the entire itemset. Following the principles introduced by [18], USBCF relies on the previously proposed coclustering approach to estimate the typical small fraction of items falling into this condition, resorting to its default behavior for items included in the personalized data spaces.

5 EXPERIMENTS AND RESULTS

This section experimentally assesses the proposed approach against state-of-the-art peers. In particular, we aim to answer the following research questions:

- **RQ1.** How does USBCF compare against baseline CF approaches in terms of quality of predictions and coverage capability?
- **RQ2.** How does the biclustering search and minimum user-bicluster similarity threshold impact USBCF performance?
- **RQ3.** Can USBCF surpass the main limitations presented by the state of the art Biclustering-based CF (BBCF) [48]?

5.1 Experimental Setting

5.1.1 Experimental Platform and Software. The models were trained on a machine with Intel Xeon Silver 4216 CPU @ 2.40GHz having sixty-four cores and 64GBs of RAM. Regarding the implementation of the approaches, the USBCF and the BBCF were implemented in Python. The remaining approaches consider the available implementation in Surprise [30], a Python scikit library for building and analyzing Recommender Systems.

5.1.2 Datasets. Experiments were run on two benchmark datasets with different volumes (MovieLens-100k and MovieLens-1M).

The MovieLens-100k\(^2\) dataset [22], one of the most popular publicly available benchmark datasets to serve as domain testing for new CF approaches. This version of the MovieLens’ dataset contains 100,000 ordinal ratings (1-5 scale) from 943 users on 1,682 movies. The data is in the form of triplets \(<user, item, rating>\). Each user has a minimum of 20 ratings made and a maximum of 737, but the average is 106 ratings per user with a standard deviation of 100 ratings. The sparsity of the MovieLens-100k is 93.6%, in other words, approximately 6.4% of the entries are filled.

The MovieLens-1M\(^3\) dataset [22] is a bigger version of the MovieLens-100k dataset that contains 1,000,209 ratings of 3,707 movies made by 6,040 users. The sparsity of the MovieLens-1M is 95.54%.

5.1.3 Experimental Protocol. We trained and evaluated all the approaches on the canonical u.base/u.test splits of the MovieLens-100k dataset, used to perform a 5-fold cross-validation with 80% of the
ratings as training data and the remaining 20% for testing per fold. The same procedure was used for the MovieLens-1M dataset.

To measure the rating prediction accuracy of the recommendation algorithms, we used two standard metrics, namely Mean Absolute Error (MAE),

\[
MAE = \frac{1}{n} \sum_{j=1}^{n} |\hat{r}_{u_0,ij} - r_{u_0,ij}|,
\]

and Root Mean Squared Error (RMSE),

\[
RMSE = \sqrt{\frac{1}{n} \sum_{j=1}^{n} (\hat{r}_{u_0,ij} - r_{u_0,ij})^2}.
\]

In this work, beyond the accuracy dimension, we also consider coverage measures for our evaluation, as some approaches provide high quality predictions but for only a small part of the user-item pairs.

- **Prediction Coverage.** The percentage of user-item pairs for which a prediction can be made
- **Item Coverage.** The number of items for which predictions can be formed as a percentage of the total number of items

The optimal hyper-parameters for each model have been estimated using cross-validation on the available training data. Details on the selected hyper-parameters and the experiments reproducibility are discussed in section 5.5.

5.2 Performance comparison with popular CF approaches (RQ1)

In this experiment, we compare the performance of the USBCF against baselines and state-of-the-art rating prediction methods on the two benchmark datasets. The results reported for the USBCF approach use a combination of QUBIC2 biclustering solutions with varying minimum of columns, \( minCols \in \{3, 5, 7, 10, 15, 20\} \), a minimum user-bicluster similarity threshold of 0.25, and an Item-based CF with a neighborhood of 20 items as the CF algorithmic choice. We refer to this model as USBCF-IB.

5.2.1 Methods.

- **Baseline Rating Predictor (Bias)** [36]: user-item bias rating prediction algorithm, denoted by: \( \hat{r}_{i,j} = \mu + b_i + b_j \), where \( \mu \) is the global mean rating, \( b_j \) is the item bias, and \( b_i \) is the user bias.
- **User-based CF (USCF)** [43]: user-based CF model with mean-centered cosine similarity as similarity function and weighted-average - taking into account the mean ratings of each user - as aggregation function.
- **Item-based CF (IBCF)** [46]: item-based CF model with mean-centered cosine similarity as similarity function and weighted-average - taking into account the mean ratings of each user - as aggregation function.
- **SVD Model by Simon Funk (FunkSVD)** [16]: the famous SVD-inspired approach popularized by Simon Funk during the Netflix Prize contest. It uses regularized stochastic gradient descent to train the user-feature and the item-feature matrix.
- **Singular Value Decomposition with Implicit Feedback (SVD++)** [35]: an extension of the classic SVD approach that incorporates implicit feedback into the SVD model.
- **Non-negative Matrix Factorization (NMF)** [39]: non-negative matrix factorization model for rating prediction.
- **Weighted Bregman Co-Clustering (CoCCF)** [18]: scalable Collaborative Filtering framework that uses weighted Bregman co-clustering and averages to generate predictions.
- **Biclustering-based CF (BBCF)** [48]: state-of-the-art biclustering-based methodology that uses biclustering as a preprocessing step to scale CF approaches.

### 5.2.2 Results

Results in Table 1 show that USBCF-IB outperforms the Bias, UBCF, IBCF, and CoCCF models in rating prediction accuracy with statistical significance (at $\alpha=1E^{-3}$), falling behind the BBCF-IB model in terms of accuracy yet significantly surpassing this later model in user and item coverage. Generally, the USBCF-IB approach achieves a rating prediction performance comparable to the matrix factorization-based approaches on the targeted data domain. It is important to note that UBCF, IBCF, BBCF-IB, and USBCF-IB accuracy results could possibly be further improved using a different similarity or aggregation method. For instance, subtracting the item’s mean rating instead of the user’s mean rating tends to be more effective [12].

In terms of predictive coverage capability, the compared approaches were able to output a prediction for nearly 100% of the test pairs, while the USBCF-IB model returns predictions for 80.03% of the pairs and considers 65.14% of the overall items, in the absence of the principles introduced along section 4.4, nearly doubling the predictive coverage of the BBCF model. Focusing on the USBCF-IB and IBCF models on the MovieLens-100k dataset, the results suggest the USBCF-IB approach can improve the IBCF accuracy results with a slight decrease in the coverage capability.

To investigate if the USBCF-IB model produced better predictions for the test pairs it made a prediction, we complemented the USBCF-IB with predictions of alternative approaches for the test pairs the USBCF-IB could not output a prediction. The results in Table 2 confirm the USBCF is improving the classic IBCF (k=20) model, and show how the performance of the hybrid approaches vary according to the minimum user-bicluster similarity (minSim) USBCF parameter. Combining USBCF-UB with a CoCCF model also improves the quality of the predictions considerably.

Results on the MovieLens-1M dataset also indicate a competitive predictive accuracy of the proposed approach against the benchmarks methods. Table 3 shows that in this data, the USBCF-IB model, with a minimum similarity of 0.25, retrieves an average mean absolute error of 0.686, covering 90.45% of the user-item pairs. Approaches based on matrix factorization strategies such as NMF were not included as they were not able to converge in due time for adequate parameterization settings considering the size of the dataset. When the complementing USBCF-IB model with the remaining 9.55% of the predictions, the results in Table 4 support the hypothesis that the USBCF-IB model improves the classic IBCF.

| Model       | MAE      | RMSE     | Coverage (%) | Item Coverage (%) |
|-------------|----------|----------|--------------|-------------------|
| Bias        | 0.750 ± 0.007 | 0.946 ± 0.009 | 100.00       | 100.00            |
| UBCF        | 0.752 ± 0.005 | 0.959 ± 0.007 | 99.83        | 97.85             |
| IBCF        | 0.746 ± 0.004 | 0.952 ± 0.005 | 99.83        | 97.85             |
| FunkSVD     | 0.720 ± 0.006 | 0.912 ± 0.006 | 100.00       | 100.00            |
| SVD++       | 0.719 ± 0.006 | 0.911 ± 0.007 | 100.00       | 100.00            |
| NMF         | 0.719 ± 0.005 | 0.921 ± 0.006 | 99.83        | 97.85             |
| CoCCF       | 0.757 ± 0.008 | 0.967 ± 0.010 | 100.00       | 100.00            |
| BBCF-IB     | 0.690 ± 0.007 | 0.886 ± 0.006 | 42.52        | 42.65             |
| USBCF-IB    | 0.716 ± 0.003 | 0.913 ± 0.004 | 80.03        | 65.14             |

Table 1. Average predictive results on the MovieLens-100k dataset.
Table 2. Predictive accuracy and coverage of USBCF when complemented with other CF approaches on the MovieLens-100k dataset.

| Model                        | MAE        | RMSE       | Coverage (%) | Item Coverage (%) |
|------------------------------|------------|------------|--------------|-------------------|
| USBCF-IB+IBCF (minSim=0.1)  | 0.742 ± 0.005 | 0.947 ± 0.005 | 99.83        | 97.85             |
| USBCF-IB+IBCF (minSim=0.2)  | 0.741 ± 0.004 | 0.946 ± 0.005 | 99.83        | 97.85             |
| USBCF-IB+IBCF (minSim=0.3)  | 0.740 ± 0.004 | 0.945 ± 0.004 | 99.83        | 97.85             |
| USBCF-IB+IBCF (minSim=0.4)  | 0.739 ± 0.004 | 0.945 ± 0.005 | 99.83        | 97.85             |
| USBCF-IB+IBCF (minSim=0.5)  | 0.740 ± 0.004 | 0.947 ± 0.005 | 99.83        | 97.85             |
| USBCF-IB+CoCCF (minSim=0.1) | 0.742 ± 0.004 | 0.947 ± 0.005 | 100.00       | 100.00            |
| USBCF-IB+CoCCF (minSim=0.2) | 0.742 ± 0.004 | 0.947 ± 0.005 | 100.00       | 100.00            |
| USBCF-IB+CoCCF (minSim=0.3) | 0.743 ± 0.005 | 0.950 ± 0.006 | 100.00       | 100.00            |
| USBCF-IB+CoCCF (minSim=0.4) | 0.745 ± 0.005 | 0.954 ± 0.008 | 100.00       | 100.00            |
| USBCF-IB+CoCCF (minSim=0.5) | 0.750 ± 0.007 | 0.960 ± 0.010 | 100.00       | 100.00            |

Table 3. Average predictive results of baseline and subspace clustering-based approaches on the MovieLens-1M dataset.

| Model                        | MAE        | RMSE       | Coverage (%) | Item Coverage (%) |
|------------------------------|------------|------------|--------------|-------------------|
| Bias                         | 0.719 ± 0.001 | 0.909 ± 0.001 | 100.00        | 100.00            |
| UBCF                         | 0.735 ± 0.001 | 0.932 ± 0.001 | 99.98        | 99.22             |
| IBCF                         | 0.704 ± 0.001 | 0.899 ± 0.001 | 99.98        | 99.22             |
| CoCCF                        | 0.717 ± 0.001 | 0.915 ± 0.002 | 100.00       | 100.00            |
| BBCF                         | 0.676 ± 0.001 | 0.866 ± 0.001 | 71.08        | 61.30             |
| USBCF-IB (minSim=0.25)       | 0.686 ± 0.001 | 0.876 ± 0.001 | 90.45        | 77.21             |

Table 4. Predictive accuracy and coverage of USBCF complemented with other CF approaches on the MovieLens-1M dataset.

The testing time of USBCF-IB is computationally efficient, bounded by the lightweighted computational complexity of the subsequent item-based CF step. USBCF-IB training time is also efficient under heuristic biclustering searches and the creation of the personalized dataset (linear on the number of biclusters and their pattern length). Finally, once biclusters are discovered on referenced rating data, USBCF-IB can be applied in an online manner to predict preferences for unseen users.

5.3 Biclustering and user-bicluster similarity in USBCF (RQ2)

In the second experiment, we study the impact of biclustering search on the USBCF performance using different QUBIC2 parameterizations. We also assess how the user-bicluster similarity affects the size of the personalized datasets produced by the approach, directly impacting the accuracy and coverage results. Accordingly, we varied the parameter of QUBIC2 that controls the shape of
biclusters and the USBCF user-bicluster similarity threshold, both affecting the number, size and format of the biclusters produced and therefore the properties of the personalized data spaces.

Figure 10 shows that, for the MovieLens-100k dataset, the average quality of the predictions tends to be worse for biclustering solutions with a larger minimum of columns per bicluster. The coverage capability peaks for biclustering solutions with a minimum of 10 columns per bicluster.

![Figure 10](image.png)

Fig. 10. Effect of the minimum columns per bicluster parameter in the performance.

We also tested the behavior of the approach when using biclusters from multiple biclustering solutions. We ran the biclustering algorithm with the minimum columns parameter in \{3, 5, 7, 10, 15, 20\}, combining the solutions and filtering non-maximal biclusters. The gathered results in Figure 11 against Figure 10 show that the combination of biclustering solutions with distinct properties produces personalized CF models with higher coverage capability and lower RMSE.

![Figure 11](image.png)

Fig. 11. Sensitivity of USBCF-IB to the user-bicluster similarity threshold parameter.

We finally examined the average size of the personalized datasets when varying the user-bicluster similarity threshold. Figure 12 shows that the USBCF minimum similarity parameter affects the dimensions of the personalized datasets, with lower threshold values creating larger datasets.
5.4 Performance comparison with Biclustering-based CF (BBCF) (RQ3)

The goal of this third experiment is to compare USBCF against the state-of-the-art biclustering-based methodology, BBCF, introduced by [48]. The reported results for BBCF were gathered using an hyperparameterized QUBIC2 biclustering solution, which yielded optimal behavior with a minimum of 15 columns per bicluster. Likewise the BBCF original work, we vary the number of biclusters per neighborhood parameter ($NNBics$).

Results in Figure 13 show that smaller sets of biclusters per neighborhood are more favorable for the quality of the rating predictions. However, there is a clear RMSE-coverage trade-off, as the model’s predictive coverage capability tends to degrade as the number of bicluster per neighborhood decreases. Figure 14 shows that the size of the personalized datasets increase with larger values for the $NNBics$ parameter. These results highlight that, as expected, larger values of $NNBics$ lead to more complex personalized models, impacting the scalability of the approach.
Comparing these results with the ones from the previous experiment (subsection 5.3), we conclude that for models with similar coverage, USBCF model surpasses BBCF in prediction error. Moreover, the USBCF approach obtains higher coverage results than BBCF while using smaller personalized datasets.

5.5 Reproducibility of the Experiments

The source code of USBCF is available at our GitHub repository \(^4\). The code to reproduce the experiments is also available in the same repository, as well as the obtained results. Regarding the hyper-parameters, we used the following defaults during the experiments:

- UBCF and IBCF: varied maximum number of users/items in the neighborhood (k). We present results for k = 20 as the results show it is an adequate value for the neighborhood size regarding predictive accuracy on both datasets.
- FunkSVD, SVD++, and NMF: grid search over the three main hyper-parameters for the optimization algorithm: number of features (n_factors), number of iterations (n_epochs) and regularization term (reg). For the MovieLens-100k data, the best hyper-parameters were n_factors = 100, n_epochs = 100, reg = 0.1 for SVD; n_factors = 30, n_epochs = 50, reg = 0.1 for SVD++; and n_factors = 200, n_epochs = 100, reg = 0.05 for NMF. The reported results use these optimal hyper-parameters estimates. As for the MovieLens-1M dataset, the best hyper-parameters were not attained as the approaches did not converge in due time.
- BBCF-IB: Varied the hyper-parameter number of nearest biclusters (nnbrs) from 50 to 300 for the Movielens-100k dataset and from 50 to 1000 for the Movielens-1M dataset, both using an Item-Based approach (k = 20) as CF algorithm. We also tested the minimum number of columns per bicluster (minCols) QUBIC2 parameter from 3 to 20. The best hyper-parameters considering predictive and coverage results were nnbrs = 50 and minCols = 15 for the Movielens-100k dataset; and nnbrs = 500 and minCols = 15 for the Movielens-1M dataset.
- USBCF-IB: Varied the minimum user-bicluster similarity threshold (minSim) parameter from 0.1 to 0.5, and used an Item-Based approach (k = 20) as CF algorithm. The best minSim considering predictive and coverage results were minSim = 0.25 for both datasets.

\(^4\)https://git.lastige.di.fc.ul.pt/mmsilva/usbcf-paper
6 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a user-specific CF approach that uses biclustering as a sub-step to improve traditional CF methods. Principles for superior CF were introduced, including superior matching of active user preferences against preference patterns to account for preference diversity; formation of user-personalized data spaces to account for preference locality and sparsity; and secondary rating estimates from coclustering structures to yield high predictive coverage.

Experimental results on the popular MovieLens-100k and MovieLens-1M benchmark datasets against state-of-the-art CF methods show that USBCF yields competitive rating prediction errors, significantly improving classic memory-based CF approaches by alleviating the sparsity of the datasets. Moreover, the proposed approach outperforms the state-of-the-art biclustering-based approach (BBCF) in prediction quality while successfully overcoming BBCF’s major limitation, the predictive coverage capability.

This work displays the potentialities of the biclustering technique for recommendation purposes. Accordingly, we highlight potentially relevant research avenues for future work:

- explore the impact that different coherence assumptions, degrees of noise tolerance, and biclustering searches yield on the USBCF approach. As USBCF can be parameterized with any biclustering algorithm with minor modifications to the overall methodology, we aim at assessing the role of order-preserving preference patterns in CF recurring to BicPAMS [25], quality relaxations (percentage of noisy and missing elements) [26, 53], and iterative search-and-masking procedures [25] to guarantee a better coverage of the original rating data space;
- incorporate available background knowledge associated with the profile of users, as well as metadata related to the composition and characteristics of items, to guide the USBCF approach using knowledge-enriched biclustering searches [27];
- extend USBCF to online settings. USBCF is as-is readily applicable to new users. Still, to handle preference drift along time, the underlying biclustering solutions used to create the personalized data spaces must be updated with newly arriving user-item ratings. To prevent the need to continuously retrain USBCF, we aim at developing an incremental version of USBCF based on updatable biclustering searches where new items and users can be added/remove to the subspaces in order to explore the inherently temporal context of user-item ratings.

ACKNOWLEDGMENTS

This work was supported by the Fundação para a Ciência e a Tecnologia (FCT) under LASIGE strategic project (UIDB/00408/2020 and UIDP/00408/2020), iCare4U (PTDC/EME-SIS/31474/2017), WISDOM (DSAIPA/DS/0089/2018), ILU (DSAIPA/DS/0111/2018) and INESC-ID (UIDB/50021/2020).

REFERENCES

[1] Faris Alqadah and Raj Bhatnagar. 2011. Similarity measures in formal concept analysis. *Ann. Math. Artif. Intell.* 61, 3 (2011), 245–256. https://doi.org/10.1007/s10472-011-9257-7

[2] Faris Alqadah, Chandan K. Reddy, Junling Hu, and Hatim F. Alqadah. 2015. Biclustering neighborhood-based collaborative filtering method for top-n recommender systems. *Knowl. Inf. Syst.* 44, 2 (2015), 475–491. https://doi.org/10.1007/s10115-014-0771-x

[3] Arindam Banerjee, Inderjit S. Dhillon, Joydeep Ghosh, Srujana Merugu, and Dharmendra S. Modha. 2007. A Generalized Maximum Entropy Approach to Bregman Co-clustering and Matrix Approximation. *J. Mach. Learn. Res.* 8 (2007), 1919–1986. http://dl.acm.org/citation.cfm?id=1314563

[4] J. Bennett and S. Lanning. 2007. The Netflix Prize. In *Proceedings of the KDD Cup Workshop 2007*. ACM, New York, 3–6.
http://www.cs.uic.edu/~liub/KDD-cup-2007/NetflixPrize-description.pdf
[5] Sara C. Madeira, Miguel Cacho Teixeira, Isabel Sá-Correia, and Arlindo Oliveira. 2010. Identification of Regulatory Modules in Time Series Gene Expression Data Using a Linear Time Biclustering Algorithm. IEEE/ACM transactions on computational biology and bioinformatics. ACM 7 (04 2010), 153–65. https://doi.org/10.1109/TCBB.2008.34

[6] Yizong Cheng and George M. Church. 2000. Biclustering of Expression Data. In Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, August 19-23, 2000, La Jolla / San Diego, CA, USA, Philip E. Bourne, Michael Gribskov, Russ B. Altman, Nancy Jensen, Debra A. Hope, Thomas Lengauer, Julie C. Mitchell, Eric D. Scheef, Chris Smith, Shawn Strande, and Helge Weissig (Eds.). AAAI, 93–103. http://www.aaai.org/Library/ISMB/2000/ismb00-010.php

[7] Yizong Cheng and George M. Church. 2000. Biclustering of expression data.. In Ismb. Vol. 8. 93–103.

[8] Guilherme Palermo Coelho, Fabricio Olivetti de França, and Fernando J. Von Zuben. 2008. A Multi-Objective Multi-population Approach for Biclustering. In Artificial Immune Systems, 7th International Conference, ICARIS 2008, Phuket, Thailand, August 10-13, 2008. Proceedings (Lecture Notes in Computer Science, Vol. 5132), Peter J. Bentley, Doheon Lee, and Sungwon Jung (Eds.). Springer, 71–82. https://doi.org/10.1007/978-3-540-85072-4_7

[9] Pablo A. D. de Castro, Fabricio Olivetti de França, Hamilton M. Ferreira, and Fernando J. Von Zuben. 2007. Applying Biclustering to Perform Collaborative Filtering. In Seventh International Conference on Intelligent Systems Design and Applications, ISDA 2007, Rio de Janeiro, Brazil, October 20-24, 2007, Luiza de Macedo Mourelle, Nadia Nedjah, Janusz Kacprzyk, and Ajith Abraham (Eds.). IEEE Computer Society, 421–426. https://doi.org/10.1109/ISDA.2007.91

[10] Fabricio Olivetti De França, Guilherme Palermo Coelho, and Fernando J Von Zuben. 2009. Coherent recommendations using biclustering. In Proc. of the XXX Congresso Ibero-Latino-Americano de Métodos Computacionais em Engenharia (CILAMCE). 1–15.

[11] Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. 2003. Information-theoretic co-clustering. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 24 - 27, 2003, Lise Getoor, Ted E. Senator, Pedro M. Domingos, and Christos Faloutsos (Eds.). ACM, 89–98. https://doi.org/10.1145/956750.956764

[12] Michael D. Ekstrand, Michael Ludwig, Joseph A. Konstan, and John Riedl. 2011. Rethinking the recommender research ecosystem: reproducibility, openness, and LensKit. In Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011, Bamshad Mobasher, Robin D. Burke, Dietmar Jannach, and Gediminas Adomavicius (Eds.). ACM, 133–140. https://doi.org/10.1145/2043932.2043958

[13] Michael D. Ekstrand, John Riedl, and Joseph A. Konstan. 2011. Collaborative Filtering Recommender Systems. Found. Trends Hum. Comput. Interact. 4, 2 (2011), 175–243. https://doi.org/10.1561/1100000009

[14] Islam El-Nabarawy, Donald C. Wunsch II, and Ashraf M. Abdelbar. 2016. Biclustering ARTMAP collaborative filtering recommender system. In 2016 International Joint Conference on Neural Networks, IJCNN 2016, Vancouver, BC, Canada, July 24-29, 2016. IEEE, 2986–2991. https://doi.org/10.1109/IJCNN.2016.7727578

[15] Liang Feng, Qianchuan Zhao, and Cangqi Zhou. 2020. Improving performances of Top-N recommendations with co-clustering method. Expert Syst. Appl. 143 (2020). https://doi.org/10.1016/j.eswa.2019.113078

[16] Simon Funk. 2006. Netflix Update: Try This At Home. Retrieved from https://sifter.org/simon/journal/20061211.html. Accessed: February 11, 2021.

[17] Bernhard Ganter and Rudolf Wille. 1999. Formal Concept Analysis - Mathematical Foundations. Springer. https://doi.org/10.1007/978-3-642-59830-2

[18] Thomas George and Srujana Merugu. 2005. A Scalable Collaborative Filtering Framework Based on Co-Clustering. In Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005), 27-30 November 2005, Houston, Texas, USA, IEEE Computer Society, 625–628. https://doi.org/10.1109/ICDM.2005.14

[19] David Goldberg, David A. Nichols, Brian M. Oki, and Douglas B. Terry. 1992. Using Collaborative Filtering to Weave an Information Tapestry. Commun. ACM 35, 12 (1992), 61–70. https://doi.org/10.1145/138859.138867

[20] Stephen Grossberg. 2013. Adaptive Resonance Theory: How a brain learns to consciously attend, learn, and recognize a changing world. Neural Networks 37 (2013), 1–47. https://doi.org/10.1016/j.neunet.2012.09.017

[21] Neelima Gupta and Seema Aggarwal. 2010. MIb: Using mutual information for biclustering gene expression data. Pattern Recognit. 43, 8 (2010), 2692–2697. https://doi.org/10.1016/j.patcog.2010.03.002

[22] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. ACM Trans. Interact. Intell. Syst. 5, 4 (2016), 19:1–19:19. https://doi.org/10.1145/2827872

[23] J. A. Hartigan. 1972. Direct Clustering of a Data Matrix. J. Amer. Statist. Assoc. 67, 337 (1972), 123–129. https://doi.org/10.1080/01621459.1972.10481214

[24] Rui Henriques, Claudia Antunes, and Sara C. Madeira. 2015. A Structured View on Pattern Mining-based Biclustering. Pattern Recognition 4, 12 (2015), 3941–3958.

[25] Rui Henriques, Francisco L. Ferreira, and Sara C. Madeira. 2017. BiPAMS: software for biological data analysis with pattern-based biclustering. BMC Bioinform. 18, 1 (2017), 82:1–82:16. https://doi.org/10.1186/s12859-017-1493-3
[26] Rui Henriques and Sara C Madeira. 2004. BicPAM: Pattern-based biclustering for biomedical data analysis. *Algorithms for Molecular Biology* 9, 1 (2014), 1–30.

[27] Rui Henriques and Sara C. Madeira. 2016. BiC2PAM: constraint-guided biclustering for biological data analysis with domain knowledge. *Algorithms Mol. Biol.* 11 (2016), 23. https://doi.org/10.1186/s13015-016-0085-5

[28] Rui Henriques and Sara C. Madeira. 2018. BSig: evaluating the statistical significance of biclustering solutions. *Data Mining and Knowledge Discovery* 32, 1 (2018), 124–161.

[29] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* 22, 1 (2004), 5–53. https://doi.org/10.1145/963770.963772

[30] Nicolas Hug. 2020. Surprise: A Python library for recommender systems. *J. Open Source Softw.* 5, 52 (2020), 2174. https://doi.org/10.21105/joss.02174

[31] Joel D Irish. 2010. The mean measure of divergence: Its utility in model-free and model-bound analyses relative to the Mahalanobis D2 distance for nonmetric traits. *American Journal of Human Biology* 22, 3 (2010), 378–395.

[32] Nyoman Juniarta. 2019. *Mining complex data and biclustering using formal concept analysis. (Fouille de données complexes et biclustering avec l’analyse formelle de concepts).* Ph.D. Dissertation. University of Lorraine, Nancy, France. https://tel.archives-ouvertes.fr/tel-02426034

[33] Surya Kant and Tripti Mahara. 2018. Nearest biclusters collaborative filtering framework with fusion. *J. Comput. Sci.* 25 (2018), 204–212. https://doi.org/10.1016/j.jocs.2017.03.018

[34] Yuval Kluger, Ronen Basri, Joseph T Chang, and Mark Gerstein. 2003. Spectral biclustering of microarray data: co-clustering genes and conditions. *Genome research* 13, 4 (2003), 703–716.

[35] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Las Vegas, Nevada, USA, August 24–27, 2008, Ying Li, Bing Liu, and Sunita Sarawagi (Eds.). ACM, 426–434. https://doi.org/10.1145/1401890.1401944

[36] Yehuda Koren. 2010. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Trans. Knowl. Discov. Data* 4, 1 (2010), 1:1–1:24. https://doi.org/10.1145/1644873.1644874

[37] Guojun Li, Qin Ma, Haibao Tang, Andrew H. Paterson, and Ying Xu. 2009. QUBIC: a qualitative biclustering algorithm for analyses of gene expression data. *Nucleic Acids Research* 37, 15 (06 2009), e101–e101. https://doi.org/10.1093/nar/gkp491

[38] Changyong Liang and YaJun Leng. 2014. Collaborative filtering based on information-theoretic co-clustering. *Int. J. Syst. Sci.* 45, 3 (2014), 589–597. https://doi.org/10.1080/00207721.2012.724109

[39] Xin Luo, Mengchu Zhou, Yunni Xia, and Qingsheng Zhu. 2014. An Efficient Non-Negative Matrix-Factorization-Based Approach to Collaborative Filtering for Recommender Systems. *IEEE Trans. Ind. Informatics* 10, 2 (2014), 1273–1284. https://doi.org/10.1109/TII.2014.2308433

[40] Sara C. Madeira and Arlindo L. Oliveira. 2004. Biclustering Algorithms for Biological Data Analysis: A Survey. *IEEE ACM Trans. Comput. Biol. Bioinform.* 1, 1 (2004), 24–45. https://doi.org/10.1109/TCBB.2004.2

[41] Tripti Mahara et al. 2016. A new similarity measure based on mean measure of divergence for collaborative filtering in sparse environment. *Procedia Computer Science* 89 (2016), 450–456.

[42] T. M. Murali and Simon Kasif. 2003. Extracting Conserved Gene Expression Motifs from Gene Expression Data. In *Proceedings of the 8th Pacific Symposium on Biocomputing, PSB 2003*, Lihue, Hawaii, USA, January 3-7, 2003, Russ B. Altman, A. Keith Dunker, Lawrence Hunter, and Teri E. Klein (Eds.). 77–88. http://psb.stanford.edu/psb-online/proceedings/psb03/murali.pdf

[43] Paolo Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *CSCW ’94, Proceedings of the Conference on Computer Supported Cooperative Work*, Chapel Hill, NC, USA, October 22-26, 1994, John B. Smith, F. Donelson Smith, and Thomas W. Malone (Eds.). ACM, 175–186. https://doi.org/10.1145/192844.192905

[44] Francesco Ricci, Lior Rokach, and Bracha Shapira (Eds.). 2015. *Recommender Systems Handbook*. Springer. https://doi.org/10.1007/978-1-4899-7637-6

[45] Pavan Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *CSCW ’94, Proceedings of the Conference on Computer Supported Cooperative Work*, Chapel Hill, NC, USA, October 22-26, 1994, John B. Smith, F. Donelson Smith, and Thomas W. Malone (Eds.). ACM, 175–186. https://doi.org/10.1145/192844.192905

[46] Badrul Munir Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2002. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the fifth international conference on computer and information technology*, Vol. 1. Citeseer, 291–324.

[47] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10*, Hong Kong, China, May 1-5, 2001, Vincent Y. Shen, Nobuo Saito, Michael R. Lyu, and Mary Ellen Zurko (Eds.). ACM, 285–295. https://doi.org/10.1145/371920.372071

[48] Kelvin Sim, Vivekanand Gopalkrishnan, Arthur Zimek, and Gao Cong. 2013. A survey on enhanced subspace clustering. *Data Min. Knowl. Discov.* 26, 2 (2013), 332–397. https://doi.org/10.1007/s10618-012-0258-x

[49] Monika Singh and Monica Mehrotra. 2018. Impact of biclustering on the performance of Biclustering based Collaborative Filtering. *Expert Syst. Appl.* 113 (2018), 443–456. https://doi.org/10.1016/j.eswa.2018.06.001
[49] Xiaoyuan Su and Taghi M. Khoshgoftaar. 2009. A Survey of Collaborative Filtering Techniques. *Adv. Artif. Intell.* 2009 (2009), 421425:1–421425:19. https://doi.org/10.1155/2009/421425

[50] Panagiotis Symeonidis, Alexandros Nanopoulos, Apostolos Papadopoulos, and Yannis Manolopoulos. 2006. Nearest-Biclusters Collaborative Filtering with Constant Values. In *Advances in Web Mining and Web Usage Analysis, 8th International Workshop on Knowledge Discovery on the Web, WebKDD 2006, Philadelphia, PA, USA, August 20, 2006, Revised Papers (Lecture Notes in Computer Science, Vol. 4811)*, Olfa Nasraoui, Myra Spiliopoulou, Jaideep Srivastava, Bamshad Mobasher, and Brij M. Masand (Eds.). Springer, 36–55. https://doi.org/10.1007/978-3-540-77485-3_3

[51] Panagiotis Symeonidis, Alexandros Nanopoulos, Apostolos N. Papadopoulos, and Yannis Manolopoulos. 2008. Nearest-biclusters collaborative filtering based on constant and coherent values. *Inf. Retr.* 11, 1 (2008), 51–75. https://doi.org/10.1007/s10791-007-9038-4

[52] João Vinagre, Alípio Mário Jorge, and João Gama. 2015. An overview on the exploitation of time in collaborative filtering. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* 5, 5 (2015), 195–215. https://doi.org/10.1002/widm.1160

[53] Juan Xie, Anjun Ma, Yu Zhang, Bingqiang Liu, Sha Cao, Cankun Wang, Jennifer Xu, Chi Zhang, and Qin Ma. 2020. QUBIC2: a novel and robust biclustering algorithm for analyses and interpretation of large-scale RNA-Seq data. *Bioinform.* 36, 4 (2020), 1143–1149. https://doi.org/10.1093/bioinformatics/btz692

[54] Rui Xu and Donald C. Wunsch II. 2011. BARTMAP: A viable structure for biclustering. *Neural Networks* 24, 7 (2011), 709–716. https://doi.org/10.1016/j.neunet.2011.03.020

[55] Gui-Rong Xue, Chenxi Lin, Qiang Yang, Wensi Xi, Hua-Jun Zeng, Yong Yu, and Zheng Chen. 2005. Scalable collaborative filtering using cluster-based smoothing. In *SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, August 15-19, 2005*, Ricardo A. Baeza-Yates, Nivio Ziviani, Gary Marchionini, Alistair Moffat, and John Tait (Eds.). ACM, 114–121. https://doi.org/10.1145/1076034.1076056