Efficient Recommender Systems via Co-Clustering-Based Collaborative Filtering

J. Naresh Kumar[1*], Merlin Sheeba G[2]
[1]CSE Dept, Sathyabama Institute of Science & Technology, Chennai, India
[2]Dept of ECE, Jerusalem College of Engineering, Chennai, India
E-mail: naresh5021@gmail.com [2]drmerlinsheebag@gmail.com
*Corresponding author

Keywords – recommender system, co-clustering, collaborative filtering, data mining, co-clustering based collaborative filtering

Received: September 13, 2024

Recommender systems became indispensable for assisting customers, users, and businesses in various domains. Collaborative Filtering (CF) is a widely used technique for generating recommendations considering user and item interactions. Many existing recommenders, such as Single Value Decomposition (SVD) and correlation, to mention a few, are based on the CF technique. These approaches suffer from two significant drawbacks. The first one is that they are computationally expensive, while the second one is the inability to cope with newly arrived user-item interactions. This leads to a situation where users' known preferences do not change over time. However, for all practical purposes in real-time applications, there is a need to update user preferences dynamically. In this paper, we proposed a novel approach known as co-clustering-based CF that performs real-time CF considering newly arrived items, users, and ratings in rapid succession. It systematically clusters rows (users) and columns (items) with an incremental mining model. Specifically, we proposed an Efficient Co-Clustering-Based Product Recommender (ECPR) algorithm for dynamically generating recommendations that reflect the latest state of user-items-ratings dynamics. The framework is evaluated on the benchmark MovieLens dataset comprising 100,000 ratings from 943 users on 1,682 items. Comparative evaluation with existing CF methods, including SVD and Non-Negative Matrix Factorization (NNMF), demonstrates that ECPR achieves up to 3.3% improvement in Mean Absolute Error (MAE) and reduces training time by up to 60%. ECPR outperforms existing CF methods regarding computational cost and accuracy in generating recommendations.

Povzetek: Članek predlaga model ECPR, ki z istočasnim so-grozdovanjem uporabnikov in predmetov omogoča ažurno priporočanje ter presega SVD in NNMF pri natančnosti in hitrosti učenja.

1 Introduction

Recommender systems play a crucial role in advancing sales in e-commerce applications. They help both customers and service providers by assisting customers in making well-informed decisions. Thus, recommender systems save time and effort and improve decision-making accuracy. Not only in the e-commerce domain, recommender systems are helpful in every possible business as they hasten the convergence of decisions and transactions. Collaborative filtering (CF) is an established technique used to generate recommendations. There are many advancements in the recommender systems. The use of artificial intelligence (AI) is increasing in recommender systems. Many researchers have contributed to improving recommender systems.

Among different kinds of approaches used for making recommenders, collaborative filtering is on top as it is widely used, as explored in [1], [4], [7], and [8], to mention a few. CF may be of many kinds: item-based, user-based, model-based, and hybrid. Content-based CF method is investigated in [11], [13], [17], and [20]. Content-based CF is applied for different domains or applications such as multi-attribute networking [11], career development [13], publication recommendations [17], and restaurant survival [20]. Many researchers used CF variants to explore movie recommendations using genre correlation [14], genome tags [16], comparative study [24], and user parameter tuning [25]. Clustering-based approaches in generating recommendations are also found in many prior works, such as [21], [25], [28], and [30]. The literature shows that CF is a widely used technique, and clustering on top of CF could optimize recommender performance. In this paper, we took the clustering process forward with co-clustering integrated with CF to leverage performance in generating recommendations.

A primary goal of the exploratory study is to create an effective, adaptable recommender system that can resolve the shortcomings received from traditional collaborative filtering methods. In particular, it aims to provide an Efficient Co-Clustering Based Product Recommender (ECBR) that identifies latent patterns by concurrently clustering users and items to obtain an accurate prediction. It is also to construct a top-k recommender as an incremental mining model that learns the latest knowledge of the new users, items, or ratings without global retraining. This allows for live flexibility while lowering compute costs.

As a result, the research aims to address the following objectives: How does the simultaneous co-clustering of users and items facilitate more accurate and efficient recommendations than traditional Collaborative Filtering approaches? The incremental mining mechanism is used to study how well it would assimilate newly arrived user-item interactions while preserving the quality of recommendations on these items. Furthermore, how competitive is ECPR regarding computational costs or training speed without sacrificing prediction performance? In detail, answering these questions makes up the study's core, showing the proposed approach's practicality. Our contributions to this paper are as follows.

1. We proposed a novel approach known as co-clustering-based CF that performs real-time CF considering newly arrived items, users, and ratings in rapid succession. It systematically clusters rows (users) and columns (items) with an incremental mining model.
2. We proposed an algorithm known as an Efficient Co-Clustering-Based Product Recommender (ECPR) for dynamically generating recommendations that reflect the latest state of users' item-ratings dynamics.
3. We built an application to evaluate our methodology and an underlying ECPR algorithm in generating recommendations with a case study.

The remainder of the paper is structured as follows: Section 2 reviews the literature on existing methods for generating recommendations. Section 3 discusses the preliminary details required to understand the proposed methodology. Section 4 presents our methodology based on co-clustering on top of collaborative filtering. Section 5 presents the results of experiments, while section 6 concludes our work and provides scope for future work.

2 Related work

This section reviews the literature on existing methods for generating recommendations. Sharma et al. [1] observed that the web's vast options cause information overload. Recommender Systems, especially Collaborative Filtering, alleviate this by suggesting relevant items using Memory-based and Model-based CF techniques. Challenges remain. Hidri et al. [2] introduced a new similarity measure, OS, for collaborative filtering in recommender systems. Its effectiveness is validated through experimental studies. Chang et al. [3] opined that in the era of big data, RS efficiently handles information overload. Hybrid CF models integrate social factors, reducing data dimensions. Qian et al. [4] investigated the DMGCF model for multiple graphs and a dynamic evolution mechanism to mine and utilize side information effectively. Victor et al. [5] focused on the recommender system that aids time-strapped researchers by suggesting relevant research papers based on collaborative filtering and cosine similarity. Jain et al. [6] explored diverse similarity measures for personalized recommendations, highlighting City block distance as superior in high sparsity. Jiang et al. [7] studied collaborative filtering in recommendation systems requiring user participation—the paper models user interactions as a game, proposing behavior rules for satisfactory equilibrium.

Guo et al. [8] focused on the JMP-GCF model that incorporates multi-grained popularity features and high-order connectivity in recommendation systems, improving personalized recommendations. Sharma et al. [9] described a hybrid recommendation system that integrates collaborative and content-based filtering to address limitations and outperform existing models. Zhou et al. [10] proposed the Next Basket Recommendation system, combining blockchain and ensuring secure and private recommendations, enhancing user trust and data protection. Kim et al. [11] proposed that the multi-attribute network-based CBF method outperforms existing methods, addressing over-specialization and sparsity issues. Achhab et al. [12] proposed a hybrid recommender system that outperforms existing methods, integrating Collaborative Filtering, Content-Based, and Self-Organizing Map techniques. Gowda et al. [13] observed that machine learning aids recommendation systems for career choices, considering user preferences, skills, and feedback for better job matches. Venkatesh et al. [14] investigated movie recommendation systems that employ content-based filtering based on genres, aiming to suggest movies similar to users' preferences. Azvy et al. [15] proposed a property recommendation system using content-based filtering to aid prospective buyers in choosing desired properties efficiently.

Gou [16] developed a cloud-computing-enabled transformer-based architecture to improve functional clothing design with consideration of scalability, efficiency, and adaptability through machine learning models. Wang et al. [17] used a Content-based Journals and Conferences Recommender System for computer science to provide prioritized suggestions based on manuscript abstracts. It utilizes web crawling for continuous updates and employs a hybrid model with a softmax regression approach. While achieving 61.37% accuracy, further improvements are expected. Kusumo et al. [18] proposed the CBF-CF-GL method, combining content-based and collaborative filtering, that improves e-learning material recommendation accuracy. Experimentally, it outperforms the CBF-GL method, benefiting from good learners' ratings. Cambria et al. [19] explored personality-aware recommendation systems, addressing challenges like the cold start problem. It discusses design choices, personality modeling methods, and privacy concerns. Gao et al. [20] investigated the impact of customer-generated content on predicting restaurant survival, highlighting the role of aspect-based sentiment analysis. Despite limitations, including data availability and fraudulent reviews, the study suggests future research directions, such as considering additional factors and exploring the effects of external environmental conditions, especially during economic downturns like the COVID-19 pandemic.

Chen et al. [21] introduced the Soft K-indicators Alternative Projection (SKAP) algorithm for collaborative filtering, enhancing its subgroup representation. Additionally, integrating item type information improves recommendation accuracy, mitigating common recommender system challenges. Zhou et al. [22] explored the UICDR method that enhances collaborative filtering by detecting user-item communities, improving recommendation performance, and addressing the cold-start problem. Leger et al. [23] proposed sensitive attributes in a co-clustering model to ensure fair recommendations in collaborative filtering. The model maintains user classification independent of the attribute, ensuring ranking fairness. Experiments show a significant reduction in recommendation unfairness. Future research may investigate societal biases with MNAR processes to address the missing data challenge. Uma et al. [24] surveyed various RS approaches, comparing their pros and cons. It proposes a movie recommendation system based on collaborative filtering and SVD++ and tests it against K-NN, SVD, and Co-clustering. The proposed approach shows lower RMSE (0.9201) and MAE (0.7219), effectively addressing cold-start and data sparsity issues. Airen et al. [25] focused on tuning parameters for the Partitioned Weighted Co-Clustering method to enhance personalized movie recommendations. The proposed approach is systematically explained and tested on real-time MovieLens datasets, showing improved accuracy (MAE 0.746) compared to existing methods. However, the study is limited to specific datasets and computational constraints. Future work may combine deep learning approaches and optimize parameters using nature-inspired techniques for better performance. Anwar et al. [26] examined CF methods for improved Recommender System performance, focusing on the cold start and data sparsity challenges. It highlights KNNBaseline's success on MovieTrust datasets and proposes exploring deep learning and resolving scalability, synonymy, privacy, and gray sheep concerns.

Table 1: Comparative summary of prior collaborative filtering methods

Ref.	Methodology	Dataset Used	Accuracy/MAE	Computational Efficiency	Addressed Issues
[1]	Memory-based CF	MovieLens	MAE: ~0.81	High memory usage	Data sparsity, scalability
[4]	Multi-graph CF (DMGCF)	MovieLens, Others	Not specified	Moderate	Side information utilization
[9]	Hybrid CF + Content-Based Filtering	MovieLens, Yelp	Not specified	Moderate	Cold-start, sparsity
[24]	CF with SVD++	MovieLens	MAE: 0.92, RMSE: 0.72	High computation	Cold-start, sparsity
[25]	Co-Clustering Weighted Approach	MovieLens	MAE: 0.74	Moderate	Parameter tuning, accuracy improvement
This Paper	Co-Clustering-Based CF (ECPR)	MovieLens (100K)	MAE: 0.80 (up to 3.3% better)	60% faster	Cold-start, computational overhead, dynamic updates

Honda et al. [27] highlighted the efficacy of fuzzy co-clustering for collaborative filtering, emphasizing the significance of selecting appropriate partition models based on specific application requirements. Atasu et al. [28] observed that the Recommender systems benefit from distributed training, boosting performance. The OCuLaR algorithm optimizes multi-core CPUs, GPUs, and mixed clusters, with future goals focused on scalability and GPU efficiency. Chen et al. [29] focused on a relay-aided massive MIMO cellular network, deriving closed-form expressions for spectral and energy efficiency. It explores the trade-off between them, optimizing power control for efficiency. Two optimization methods, 1-D searching and alternate optimization, are proposed and validated through simulations. Notsu et al. [30] introduced a modified three-mode fuzzy co-clustering (3FCCM) algorithm, enhancing parameter tuning and practical applicability. The new 3FCCMP algorithm leverages a probabilistic criterion, improving fuzzification and guideline tuning. Future research may focus on extending the algorithm to handle interdependent situations among elements. On the other hand, Meesala et al. [43] proposed a simultaneous clustering and feature selection methodology based on Social Group Optimization and dynamic thresholding to analyze and perform dimension reduction over a microarray dataset. As presented in Table 1, although memory-based, [44] model-based, and hybrid collaborative filtering approaches have been significantly studied, they still have a few limitations. Memory-based approaches require processing the whole user-item rating matrix, so they face scalability and high computational cost challenges. Hybrid models address challenges like data sparsity but tend to add complexity and restrict real-time dynamism. Currently, there are co-clustering methods

[25]. However, they are designed for a static dataset as they spend most of their time in parameter tuning stages and do not offer an updating strategy to adapt these parameters for a dynamic environment. Besides, the previous works do not efficiently address the cold-start problem and the computational complexity when considering large-scale systems. Instead, the proposed ECPR model addresses these issues and fills these gaps using incremental co-clustering to achieve real-time updates of new users, items, and ratings with eased computational efficiency.

3 Preliminaries

This section provides essential details that help understand the proposed method in Section 4. It also sheds light on different kinds of recommenders, collaborative filtering (CF), and CF methods. Our novel approach in this paper is based on co-clustering, as highlighted in Figure 1.

3.1 Recommender systems

A recommender system is a system that enables users to make desired recommendations. Such systems play a crucial role in every field. Recommendations help users expedite their work and help businesses. For instance, in e-commerce, recommender systems help customers to find more suitable items quickly, and for companies, they help in faster convergence of sales.

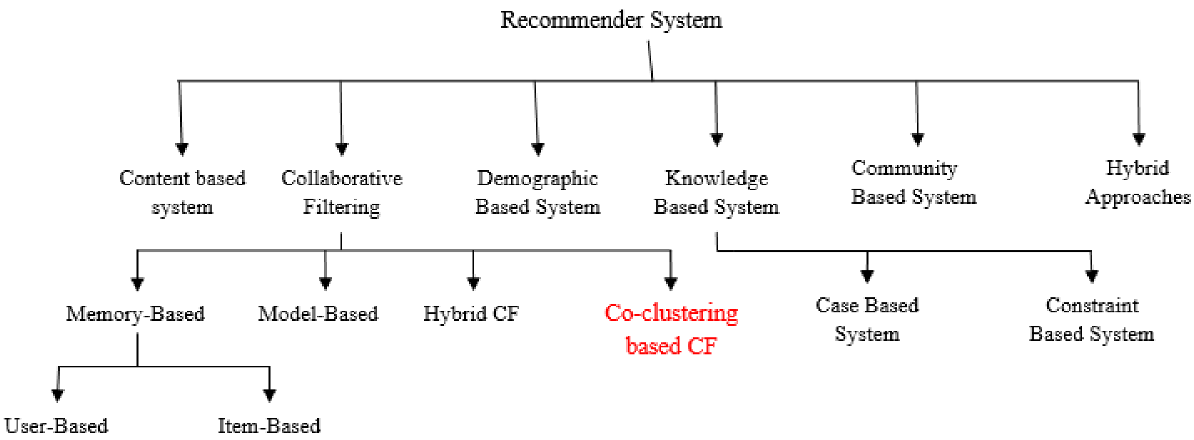

Figure 1: Overview of different kinds of recommender systems

There are many kinds of recommender systems. They are content-based, CF, knowledge-based, demographic-based, community-based, and hybrid approaches. In our research in this paper, we exploit collaborative filtering. Therefore, we explore CF approaches in Section 3.2.

3.2 Collaborative filtering

As the name implies, CF is the technique that filters choices in the system depending on the prior decisions of other users. Considering the number of users represented as, $U = [u_1, u_2, u_3, \ldots\ldots u_m]$, and the number of items denoted as, $I = [i_1, i_2, i_3, \ldots . i_n]$, it is possible to have a matrix of rating R of m x n size. Each user in the system has their opinion on I in the form of a rating score. Thus, the rating can determine the number of similar users.

Table 2: A sample matrix reflecting ratings

	i_1	i_2	i_3	i_4	i_5
u_1	4	?	5	3	1
u_2	5	2	5	?	?
u_3	2	4	?	1	3

Each row in Table 2 reflects the user's rating distribution across me. In contrast, each column reflects the rating pattern for a given item across the U. Based on the CF technique, a recommender system can recommend the most preferable items to the current user. This process of generating recommendations is done using two steps. In step 1, there is a prediction activity. It considers either user similarity or item similarity. In other words, over an item i_2, user 1 has some interest. This interest can be mined by finding other users' similarities to u sub one or other items that share similarities with i_2. With the help of data mining or machine learning (ML) algorithms, it is possible to predict and generate recommendations. The recommendation is step 2, in which the system recommends the most interesting items to u_1. From the above discussion, it is observed that CF methods could model user-item interactions depending on ratings. Sometimes, the performance of CF deteriorates if the user ratings are not genuine. In other words, such ratings lead CF to exhibit poor quality in recommendations. A baseline prediction process [32] used in CF is expressed in Eq. 1.

$$b_{ui} = \bar{r} + b_u + b_i$$
(1)

Where item and user biases are denoted as b_i and b_u Respectively, the average rating is denoted by $\bar{r}$. The prediction of this baseline algorithm is often subtracted while r_{ui} Predictions are needed to deal with user and item biases, if any.

3.2 Types of collaborative filtering

The CF technique recommends items based on like-minded users' interests or items similar to those the target user likes. As explored in [33], many statistical methods are used to determine similarity measures for CF. Here are many kinds of CF approaches.

A) Memory-based approaches

As discussed in [34], these CF algorithms memorize the matrix associated with user-item ratings and then exploit the available rating databases to compute user or item similarity. While these methods can be implemented more efficiently, they cause memory issues as they need ample space to cope with the complete rating matrix. Moreover, these approaches are lazy learners, which causes scalability issues. Memory-based approaches are of two kinds: user-based and item-based. In the user-based CF, the recommender system exploits users with similar ratings [35]. It produces a user vector that reflects everyday items to validate the rating score. It could predict the prediction probability of a user u_2 on an item i_4 by computing similarity measures between u_1 and u_2 over the items i_1 and i_3. In the same fashion, it computes similarity scores with all users concerned. u_2CF is item-based, considering the similarity of items, as discussed in [36]. It focuses on rating patterns of a specific item as opposed to user-based CF. Rating across the number of users is considered to know its preference for a given item.

Predictions are based on similarity measures that help find the nearest neighbors, leading to the CF process. These measures also help find top recommendations by finding similarities between specific users and others. Pearson correlation is one such measure, which is given in Eq. 2.

$$C_{uv} = \frac{\sum_{i=1}^{n}(r_{ui}-\bar{r}_u)(r_{vi}-\bar{r}_v)}{\sqrt{\sum_{i=1}^{n}(r_{ui}-\bar{r}_u)^2(r_{vi}-\bar{r}_v)^2}} \quad (2)$$

Given two users u and v, the average rating of these users is denoted as $\bar{r}_u$ and $\bar{r}_v$ Respectively. In the same fashion, item similarity is computed for all users. Cosine similarity is another prevalent measure for similarity computation. Sometimes, a weighted sum of ratings must be computed for prediction purposes. This can be calculated as in Eq. 3.

$$P_{ui} = \bar{r}_u + \frac{\sum_{v=1}^{m}(r_{vi}-\bar{r}_v).C_{uv}}{\sum_{v=1}^{m}|C_{uv}|} \quad (3)$$

Here for all items, P_{ui} Is computed. Once a ranked list of items is obtained, generating the top recommendations that are most relevant to a given user is possible.

B) Model-based approaches

Model-based alternatives are used to address the scalability issues of the memory-based methods. These methods are fundamentally different from that of memory-based ones. They build a model to learn from user-item interactions by representing them with low-dimensional feature vectors. They are also known as latent factor models that exploit Matrix Factorization (MF) or its variant named Singular Value Decomposition (SVD), as discussed in [37]. MF best characterizes users and items with feature vector representation reflecting rating patterns. Considering the number of users U and items I, R is the rating matrix with the size |U|x|I|. Then, to discover latent features from two matrices denoted as P and Q, computation expressed in Eq. 4 is required.

$$\hat{R} = P \times Q^T \approx R \quad (4)$$

It is observed that P (|U|×L) and Q (|I|×L) are matrices from which R is factorized. Each factor for matrix P finds users' interest in different items. There is a close relationship between users and items, leading to the generation of recommendations. For a given item i by user u, the rating prediction is as in Eq. 5.

$$\hat{r}_{ui} = p_u q_i^T \quad (5)$$

Here, feature vectors associated with the user and item are denoted as p_u and q_iFeature vectors are learned by reducing the difference between predicted and real values. Therefore, Eq. 6 is derived from Eq. 5.

$$min_{p,q} \sum_{u,i}(r_{ui} - \hat{r}_{ui})^2 + \lambda(\|p_u\|^2 + \|q_i\|^2) \quad (6)$$

Here, a regularization term λ is used to eliminate the overfitting problem. If the rating data is in continuous form, MF and SVD models are best. Otherwise, clustering approaches, as discussed in [41] and [42], probabilistic models, as explored in [39] and [40], and Bayesian methods, as studied in [38], are used.

C) Hybrid collaborative filtering

Combining model- and memory-based approaches to address scalability and sparsity issues is possible. From the literature, several methods are based on the CF technique. Table 1 shows a summary of essential literature findings. Recommender systems are found in literature about books, songs, movies, and other domains. CF approaches exploit works offline to get ready and then use online more to generate recommendations. They strive to reduce the gap between estimated values and actual values. Less difference indicates better performance. Table 3 shows the notations used in the proposed system.

Table 3: Notations used

Symbol	Description
A	The user-item rating matrix is where rows represent users and columns represent items.
W	Binary matrix indicating presence (non-zero) ratings in AA
m	Total number of users
n	Total number of items
k	Number of item (column) clusters
l	Number of user (row) clusters
$\rho(i)$	Cluster assignment function mapping user i to a row cluster
$\gamma(j)$	Cluster assignment function mapping item j to a column cluster
C_r	Set of rows (user) clusters
C_c	Set of columns (item) clusters
A^{COC}	Co-cluster average rating matrix (average ratings per user-item cluster pair)
A^{RC}	Row-cluster average matrix (average ratings per user cluster)
A^{CC}	Column-cluster average matrix (average ratings per item cluster)
A^{R}	Row-wise (user) average ratings
A^{C}	Column-wise (item) average ratings
R	Predicted ratings matrix after applying co-clustering
$Incremental\ Update$	Dynamic adjustment to clusters and summary statistics when new users/items/ratings are added
Summary Statistics	Aggregated co-cluster, row, and column averages used for prediction
MAE	Mean Absolute Error, used to evaluate the prediction accuracy

4 Proposed co-clustering based recommender system

Clustering is an unsupervised method that uses distance measures for grouping data instances. It has plenty of real-world applications, such as credit card fraud detection. However, it considers grouping of cases based on rows only. This means that it misses some helpful information in the clustering process. To overcome this, many co-clustering methods came into existence. Such methods simultaneously perform clusters on rows and columns. Recommender systems help generate recommendations in applications like e-commerce to promote business by assisting customers in making decisions faster.

4.1 Our framework

We proposed a co-clustering-based framework with CF to generate recommendations. The recommendation generation problem is considered to be a prediction problem meant to predict unknown ratings. A matrix representation of user and item interactions transforms the situation into a weighted matrix approximation and solves it using a co-clustering-based CF approach.

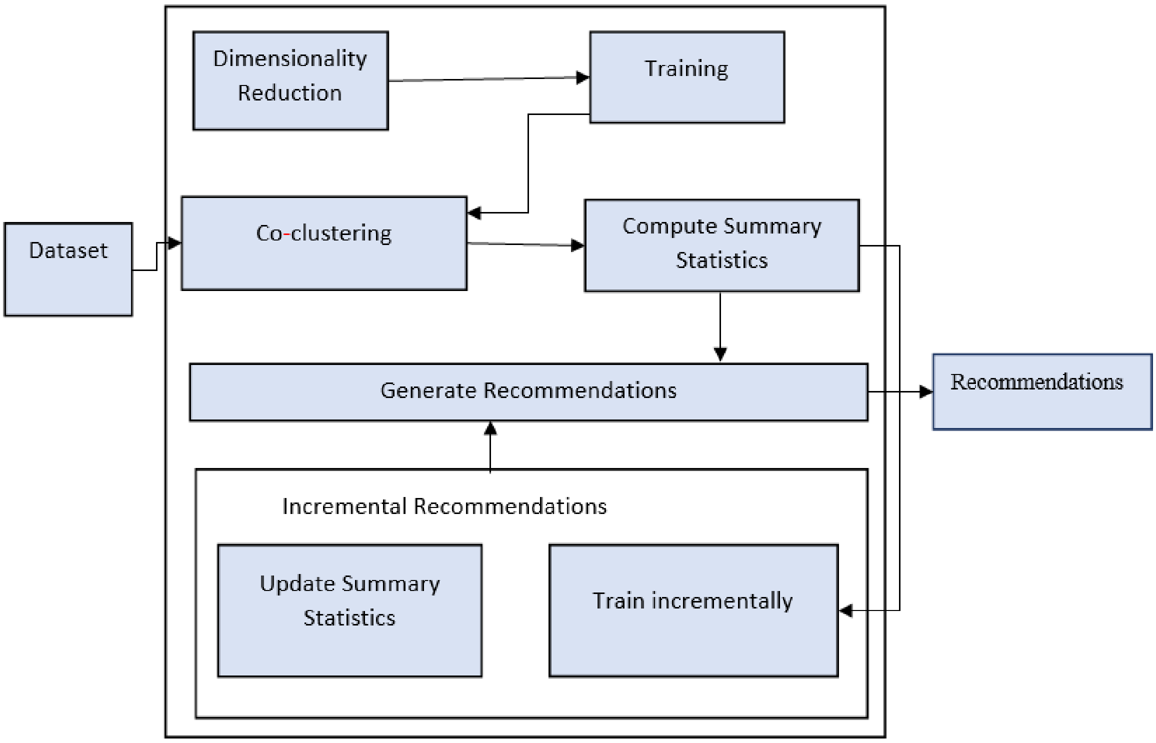

Figure 2: Proposed framework based on co-clustering for generating recommendations

As presented in Figure 2, the given data containing user and item interactions are subjected to dimensionality reduction, improving the data quality by removing rarely rated and rarely rated items. The framework supports static and dynamic approaches in training and the generation of recommendations (prediction). Static training involves a fixed set of users, items, and ratings, while dynamic training includes newly arriving user and item interactions. The interactions, in either case, are represented as matrix forms with latent factor models, and thus, it is possible to have model-based approaches. This is the reason dimensionality reduction is used in the proposed system. The predicted ratings are used to have top-n recommendations. In the training process, co-clustering is employed, and summary statistics are computed. The summary statistics are then used to generate recommendations. In a dynamic approach, when new user and item interactions arrive at runtime, the generated summary statistics are subjected to incremental updates, and the model again achieves the generation of recommendations that include dynamically added interactions.

4.2 Algorithm design

The Efficient co-clustering-based Product Recommendation (ECPR) is proposed to improve recommendation accuracy and computing efficiency by performing user and item co-clustering. It essentially has two phases: a training phase where user-item clusters are built and an incremental update phase responsive to new user-item interactions. This technique solves scalability, cold-start problems, and real-time learning in the recommender system.

Algorithm: Efficient Co-clustering-based Product Recommender (ECPR)
Inputs:
A (rating matrix)
W (non-zeros matrix)
k (column clusters)
l (row clusters
Outputs:
Co-clustering results
Summary statistics A^{COC}, A^{RC}, A^{CC}, A^{R} and A^{C}
Recommendations R

1. Begin

   Initial Training (Static)
2. Initialize (ρ, γ)

3. Repeat
4. Compute summary statistics
5. Update row clusters

$$\rho(i) = argmin_{1\le g\le k}\sum_{j=1}^{n} w_{ij}(A_{ij} - A^{COC}_{g\gamma(j)} - A^{R}_{i} + A^{RC}_{g} - A^{C}_{j} + A^{CC}_{\gamma(j)})^2, 1 \le i \le m$$

6. Update column clusters

$$\gamma(j) = argmin_{1\le h\le l}\sum_{i=1}^{m} w_{ij}(A_{ij} - A^{COC}_{\rho(i)h} - A^{R}_{i} + A^{RC}_{\rho(i)} - A^{C}_{j} + A^{CC}_{h})^2, 1 \le j \le n$$

7. Until convergence

Train Incrementally

8. Take new users and item interactions
9. User-item cluster identification
10. Consider old user-old item case

g$\leftarrow \rho(i), h \leftarrow \gamma(j)$

11. Consider old user-new item case

g$\leftarrow \rho(i), r \leftarrow 0, \gamma(j) \leftarrow 0, \mathcal{P} \leftarrow \mathcal{P} \cup \{\mathcal{P}_i\}$

12. Consider new user-old item case

g$\leftarrow 0, h \leftarrow \gamma(j), \rho(i) \leftarrow 0, \mathcal{U} \leftarrow \mathcal{U} \cup \{\mathcal{U}_i\}$

13. Consider new user-new item case

G$\leftarrow 0, h \leftarrow 0, \rho(i) \leftarrow 0, \gamma(j) \leftarrow 0$

$$\mathcal{U} \leftarrow \mathcal{U} \cup \{\mathcal{U}_i\}, \mathcal{P} \leftarrow \mathcal{P} \cup \{\mathcal{P}_i\}$$

14. Update summary statistics
15. Update non-zero counts
16. Consider old user-old item case

g$\leftarrow \rho(i), h \leftarrow \gamma(j)$

r$\leftarrow A^{R}_{i} + A^{C}_{j} - A^{RC}_{g} - A^{CC}_{h} + A^{CoC}_{gh}$

17. Consider old user-new item case

g$\leftarrow \rho(i), r \leftarrow A^{R}_{i}$

18. Consider new user-old item case

h$\leftarrow \gamma(j), r \leftarrow A^{C}_{j}$

19. Consider new user-new item case

R$\leftarrow A^{glob}$

20. Return R

Algorithm 1: Efficient co-clustering-based product recommender (ECPR)

As presented in Algorithm 1, it takes rating matrix, non-zeros matrix, column clusters, and row clusters as input and generates recommendations through a model-based learning approach incrementally. The ECPR algorithm works in two main stages: the Training Stage, where the algorithm is tuned, and the Incremental Update Stage, used to produce recommendations. The fundamental operation of the algorithm is to allow for the simultaneous clustering of users and items; thus, the systems can readily uncover hidden patterns of user-item interactions. This co-clustering methodology helps capture user preferences and item characteristics in a unified setting, consequently improving the quality of recommendations.

During the early training, the algorithm processes the dataset; a conventional representation consists of a matrix with actors, film rows, and columns, respectively, and the ratings or interactions as entry keys. Simultaneously, it partitions users into separate user clusters and items into item clusters using co-clustering algorithms. In this approach, the algorithm progressively improves these clusters by minimizing the gap between actual ratings and its cluster-based predicted ratings. Once convergence is reached, the algorithm calculates summary statistics that capture the relationship between user and item clusters, forming the basis for producing recommendations.

After the initial clustering, the algorithm moves to the incremental update phase, which is what makes ECPR different from many classical approaches. New users, items, and ratings are constantly added to real-world applications, and retraining the whole model from the beginning can require a lot of computation resources—ECPR progresses incrementally by updating its model to handle the issue. When new user-item interactions have arrived, the algorithm checks whether the new user item is similar to any clustered users and items and belongs to one of the already-formed clusters. It then adjusts the related summary statistics to reflect the new interactions. New clusters may be formed as needed to learn new patterns in the data. By incrementally maintaining and updating summary statistics without reprocessing the entire dataset, ECPR is well-suited for real-time applications. This is a significant algorithm improvement over traditional collaborative filtering methods, which must be wholly retrained to deal with new incoming data.

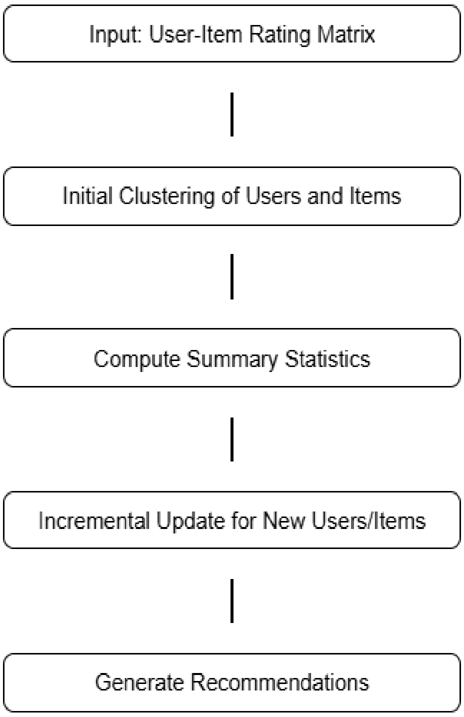

Figure 3: Flowchart of the proposed ECPR framework illustrating initial clustering, summary computation, incremental updates, and recommendation generation.

As presented in Figure 3, the proposed framework generates an initial set of user and item clusters using a bi-clustering method capable of mutually clustering rows (users) and columns (items) of the sparse user-item rating matrix iteratively. The formulation for clustering is based on minimizing the squared differences between the actual ratings and the predicted ratings derived from the cluster assignments. More specifically, the algorithm starts with either some random initial cluster memberships or using heuristics. Then, it updates the memberships so that it optimally minimizes the intra-cluster variance for each of the users and the items. It selects clusters based on an objective function that balances row and column clustering objectives, matching user preferences with item characteristics. When convergence is attained, summary statistics are calculated to reflect co-cluster means, row-cluster, and column-cluster averages. In the incremental update scheme, assigning newly sampled users or items to existing clusters is based on the minimum distance between the freshly sampled users or items and the already constructed statistics (the existing clusters), thus avoiding a rerun of complete clustering. This approach guarantees real-time adaptability and scalability with cluster assignments and summary statistics being executed dynamically with every new addition to the cluster.

4.3 Case study

Our algorithm is designed to work for different datasets. It is meant to generate recommendations faster based on co-clustering-based CF. Though the algorithm is intended to be generic, its implementation uses a specific case study called "Movie Recommendations." This case study assumes significance due to increased online OTT platforms rendering movie services. Of late, there has been a tremendous increase in the number of viewers of OTT platforms. A recommender system helps movie viewers with personalized recommendations. Such recommendations consider individual preferences and different users' historical ratings and preferences. The movie recommender system is evaluated with the Movie Recommendations case study, which found that the co-clustering-based approach outperforms existing recommenders.

5 Experimental results

This section presents experimental results in terms of exploratory data analysis and the actual performance of the proposed recommender system. The results regarding prediction accuracy against several parameters and the chosen CF algorithm are provided. Observations are also made on prediction time against a given CF algorithm and training time against the data size, besides training time against several processors. Our research is based on collaborative filtering. In other words, our Efficient Co-Clustering Product Recommender (ECPR) algorithm is designed to exploit co-clustering on top of the CF approach. Experimental results are evaluated and compared with state-of-the-art recommender systems such as SVD and NNMF.

The data split was performed without stratification, with users and items evenly represented across subsets. To emphasize the efficiency of the proposed method in a standard CPU-based setup, all experiments were performed in a computational environment with 32GB RAM, Intel Core i7 CPU (3.40GHz), and no GPU. Compared to other methodologies where we added traditional baselines like Singular Value Decomposition (SVD) and Non-Negative Matrix Factorization (NNMF), we used a hybrid collaborative filtering approach that combines SVD with the aforementioned content-based filtering methods. This combined baseline was implemented through item metadata that simulated content-based features and collaborative filtering predictions. We compare the proposed ECPR algorithm with those models regarding prediction accuracy (Mean Absolute Error) and computational efficiency (training time).

All experiments were performed with the MovieLens 100K dataset, which includes 100,000 ratings from 943 users on 1,682 items, to allow for replication. We filtered out users and items with little interaction and normalized the rating values. Data was split uniformly randomly into 70% training, 15% validation, and 15% test datasets. Peer co-clustering hyperparameters, or the number of row and column clusters, were optimized through a grid search in {2, 4, 6, 8, 10}, with the best one according to validation MAE being selected. Implemented in Python with Scikit-learn and Pandas, the experiments were conducted on an Intel Core i7 CPU (3.40GHz) machine with 32GB RAM without GPU acceleration. After each iteration, new synthetic user-item interactions were injected into the existing ones to validate the incremental update mechanism; then, the update efficiency was reported. Each experiment was performed five times using different random seeds, and the mean and 95% CIs were reported to demonstrate significance.

5.1 Exploratory data analysis

This section explores the MovieLens dataset collected from [31]. This dataset is a widely used benchmark for building recommender systems. It is a ratings dataset that helps in developing algorithms to generate recommendations.

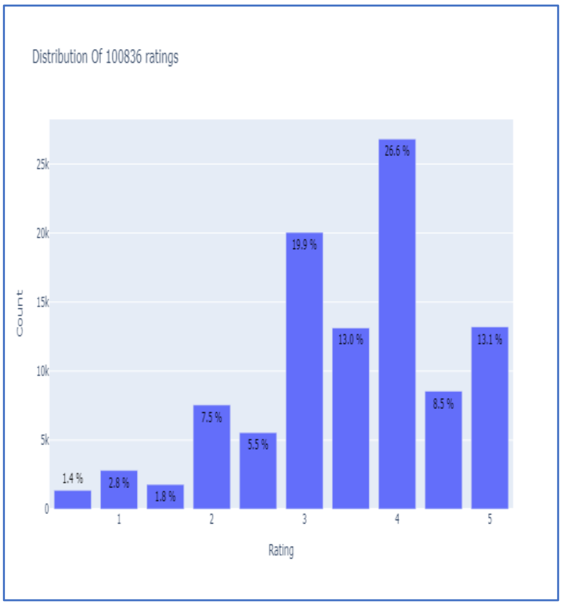

Figure 4: Distribution of ratings in the dataset

As presented in Figure 4, the dataset has different ratings. The count of instances in the dataset against each rating is visualized.

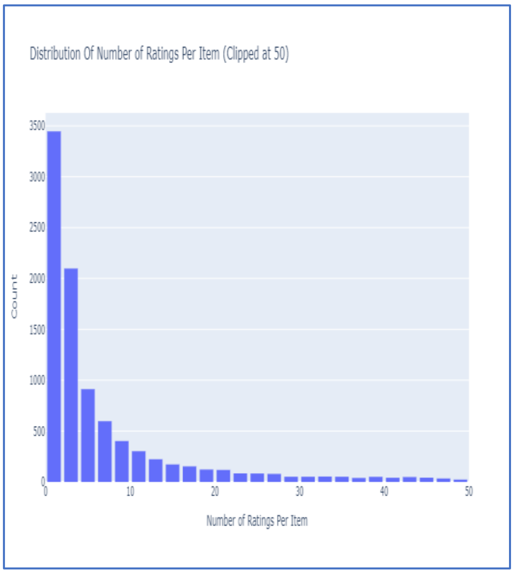

Figure 5: Distribution of the number of ratings per item in the dataset

As presented in Figure 5, several items are in the given dataset. Each item has several ratings as visualized.

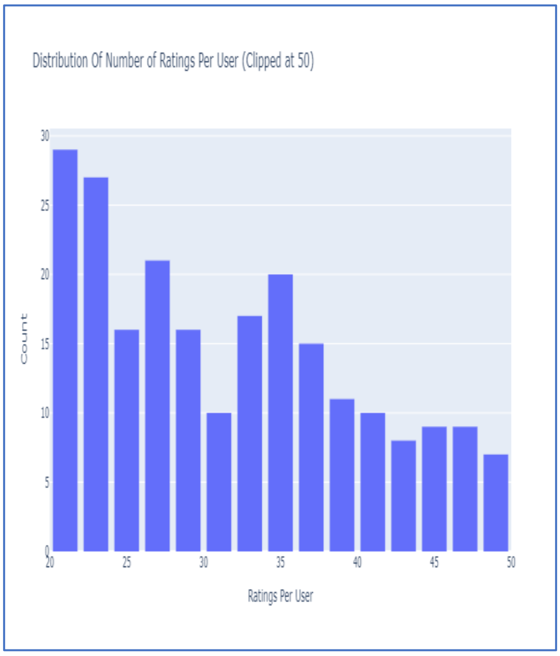

Figure 6: Distribution of the number of ratings per user in the dataset

As presented in Figure 6, several users are in the given dataset. Each user has several ratings as visualized.

5.2 Prediction accuracy

Mean Absolute Error (MAE) is the metric used to determine prediction accuracy. The MAE of the proposed algorithm ECPR is compared against SVD and NNMF. The number of prediction parameters and clusters are used to observe MAE.

Table 4: Shows MAE performance of recommender systems against k value

K Value	**MAE**		
	SVD	**NNMF**	**Proposed** (ECPR)
0	0.814	0.833	0.806
2	0.816	0.83	0.809
4	0.81	0.828	0.804
6	0.823	0.854	0.824
8	0.824	0.855	0.83
10	0.824	0.856	0.834

As presented in Table 4, the performance of the proposed recommender, known as ECPR, is compared against existing methods in terms of MAE against a given k value.

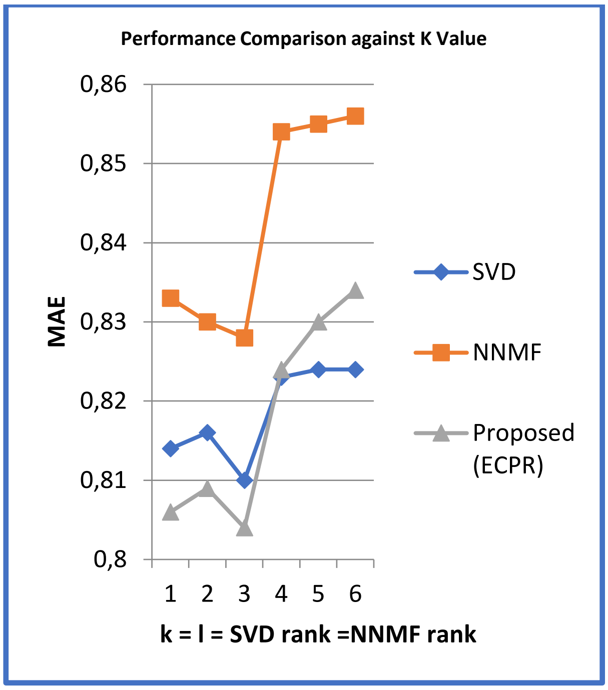

Figure 7: Performance of recommender systems in terms of MAE against k value

As presented in Figure 7, variations of MAE are shown by different recommenders and are provided against the k value. Each method exhibited a different MAE value. Less MAE value indicates better performance. When the k value is 0, SVD showed 0.814 MAE, NNMF 0.833, and proposed ECPR 0.806. When the k value is 10, SVD showed 0.824 MAE, NNMF 0.856, and proposed ECPR 0.834. The results show that NNMF showed the lowest performance as it exhibited the highest MAE against all k values. SVD is found to be far better than that of NNMF. ECPR exhibits the highest performance with initial k values up to 3. Afterward, the MAE of the proposed method is increased.

Table 5: Shows MAE of recommenders against several prediction parameters

# Prediction Parameters	MAE		
	SVD	NNMF	Proposed (ECPR)
0	0.813	0.833	0.805
2000	0.815	0.83	0.805
4000	0.81	0.828	0.808
6000	0.823	0.854	0.815
8000	0.824	0.855	0.819
10000	0.825	0.856	0.820

Table 5 presents the MAE of recommenders against several prediction parameters, reflecting the accuracy of recommendations. A lower MAE indicates better performance.

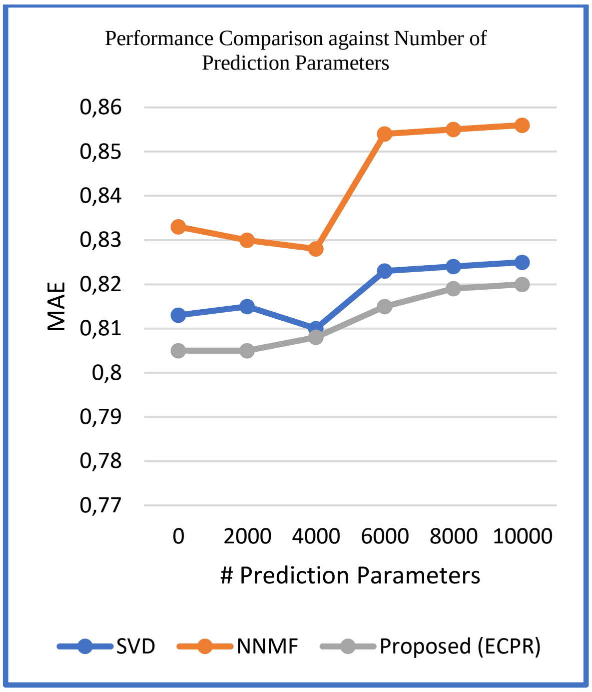

Figure 8: Performance of recommender systems in terms of MAE against k value

As presented in Figure 8, variations of MAE are shown by different recommenders and are provided against several prediction parameters. Several prediction parameters indicate additional storage requirements beyond the raw rating matrix. Concerning the proposed co-clustering approach, various statistics, such as values of (m+n+kl-k-l), are known as prediction parameters. In the case of existing methods, prediction parameters are values of ((m+n)(k+l). Each method exhibited a different MAE value. Less MAE value indicates better performance. When the number of prediction parameters is 0, SVD showed 0.813 MAE, NNMF 0.833, and proposed ECPR 0.805. When the k value is 10000, SVD showed 0.825 MAE, NNMF 0.856, and proposed ECPR 0.820. The results show that NNMF showed the lowest performance as it exhibited the highest MAE against all prediction parameters. SVD is found to be far better than that of NNMF. ECPR exhibits the highest performance with all number of prediction parameters.

Table 6: shows the training time of recommenders against several known ratings.

# Known Ratings	Training Time (seconds)	
	SVD	Proposed (ECPR)
10000	0.3	0.2
20000	0.9	0.6
30000	5	0.8
40000	7	0.9
50000	20	1

60000	30	2
70000	50	4
80000	70	6
90000	80	8

As presented in Table 6, recommenders' training time against the number of known ratings reflects efficiency in generating recommendations. Lower training time indicates better performance.

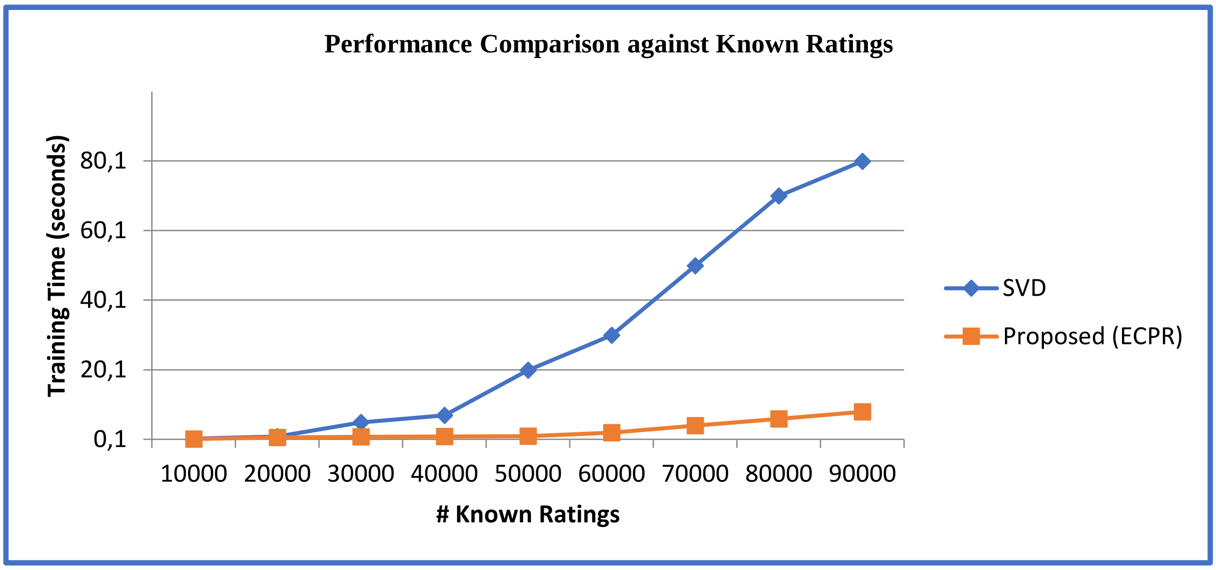

Figure 9: Training time of recommender systems against several known ratings

As presented in Figure 9, variations of training time are shown by different recommenders and are provided against several known ratings. The number of known ratings indicates the ratings of items in the dataset. Each method required a different training time. Less training time indicates better performance. As the number of known ratings increases, training time also increases. In this case, the proposed recommender system ECPR is compared against SVD. It is observed that the proposed algorithm outperforms SVD due to its modus operandi. ECPR is based on co-clustering, which simultaneously performs row and column clustering to expedite the training process under a given number of known ratings.
Hyperparameters like numbers of row and column clusters were tuned carefully (using grid search over a range of cluster numbers) to get performance tuning. The data were split into training (70%), validation (15%), and test (15%) using a 70-15-15 ratio, and the configuration with the lowest Mean Absolute Error (MAE) on the validation set was selected. Moreover, different random splits were used to conduct all the experiments five times to ensure the consistency and robustness of the results, where the average MAE values and their 95% confidence intervals (95% CI) are reported. As an illustration, ECPR yielded an MAE value reaching 0.804 ± 0.006 when $k = 4$. Although this study used the popular MovieLens dataset, evaluation was done on other datasets like Last. FM and Amazon Reviews, which we plan to augment as part of our future work to evaluate the proposed approach's generalizability and scalability.

6 Discussion

The experimental results described in Section 5 confirm that the proposed model of Efficient Co-Clustering-Based Product Recommender (ECPR) achieves better prediction accuracy and computational efficiency than some traditional collaborative filtering algorithms, including Singular Value Decomposition (SVD) and Non-Negative Matrix Factorization (NNMF). More specifically, ECPR can achieve a 3.3% reduction in MAE over SVD and is particularly effective in reducing training time when the number of ratings is increased.
Such discrepancy in MAE values can be explained by the design of ECPR. In contrast to SVD and NNMF, which are built on matrix factorization that is exploitative in dealing with sparsity and static data assumption, ECPR is based on a co-clustering approach by clustering both the user and item collaboratively. Using two-level clustering enables us to better capture latent patterns in user-item interactions, thereby enhancing prediction accuracy. Furthermore, unlike FLASH, which needs multiple retraining iterations to integrate new user-item interactions smoothly, the ECPR dynamic incremental update mechanism enables the immediate incorporation of new interactions, maximizing performance sustainability.
ECPR is also easier to scale. Traditional CF approaches, such as SVD, are computationally expensive, and as they operate on large-scale datasets, they lead to significant time and memory complexities. ECPR solves this problem by applying dimensionality reduction and represents user-

item interactions with summary statistics that capture the summary of each user-item experience, making ECPR a candidate for real-time environments in which data is never stationary. Furthermore, unlike existing methods where clustering is performed statically and does not have an online updating capacity, the key novelty of the ECPR algorithm is an in-situ co-clustering update method where rows and columns are added incrementally through an online update. Because of this design, ECPR is especially suitable for new recommendation scenarios like streaming platforms and e-commerce applications, where user preferences and supply items keep evolving.

6 Conclusion and future work

In this paper, we proposed a novel co-clustering-based CF that rapidly performs real-time CF considering newly arrived items, users, and ratings. It systematically clusters rows (users) and columns (items) with an incremental mining model. We proposed an Efficient Co-Clustering Product Recommender (ECPR) algorithm for dynamically generating recommendations reflecting the latest user-items-ratings dynamics state. We used a benchmark dataset to evaluate the proposed algorithm. ECPR outperforms existing CF methods regarding computational cost and accuracy in generating recommendations. Experimental results revealed that the proposed co-clustering-based approach to the recommender system could improve prediction accuracy and time taken for training. ECPR is based on co-clustering, which simultaneously performs row and column clustering to expedite the training process under a given number of known ratings. We will further utilize our recommender system in some potential applications, such as large-scale e-commerce platforms to recommend products for personalized, streaming services to recommend interesting content, and the advertising system to do targeted advertisements in our future work. We will also extend the underlying algorithm to allow for parallel co-clustering for better scalability and to reduce computation time even more. Alternatively, another fruitful avenue is integrating neural network architectures with co-clustering so the system can learn complex nonlinear user interactions. Additionally, graph-based models like Graph Convolutional Networks (GCNs) to represent the higher-order relationships and social connectivity between users and items may be added to improve the quality of recommendations. This enhances the applicability and performance of the proposed framework in dynamic and large environments.

References

[1] Sharma, Ritu; Gopalani, Dinesh and Meena, Yogesh (2017). 3rd International Conference on Computational Intelligence & Communication Technology (CICT) - Collaborative filtering-based recommender system: Approaches and research challenges, 1–6. http://doi:10.1109/CIACT.2017.7977363.

[2] Gazdar, Achraf and Hidri, Lotfi (2019). A new similarity measure for collaborative filtering-based recommender systems. Knowledge-Based Systems, 105058–. http://doi:10.1016/j.knosys.2019.105058.

[3] Chen, Rui; Hua, Qingyi; Chang, Yan-Shuo; Wang, Bo; Zhang, Lei and Kong, Xiangjie (2018). A Survey of Collaborative Filtering-Based Recommender Systems: from Traditional Methods to Hybrid Methods Based on Social Networks. IEEE Access, 1–1. http://doi:10.1109/ACCESS.2018.2877208.

[4] Tang, H., Zhao, G., Bu, X., & Qian, X. (2021). Dynamic evolution of multi-graph-based collaborative filtering for recommendation systems. Knowledge-Based Systems, 228, 107251. http://doi:10.1016/j.knosys.2021.107251.

[5] Murali, M Viswa; Vishnu, T G; Victor, Nancy (2019). 5th International Conference on Advanced Computing & Communication Systems (ICACCS) - A Collaborative Filtering-based Recommender System for Suggesting New Trends in Any Domain of Research. 550–553. http://doi:10.1109/ICACCS.2019.8728409.

[6] Gaurav Jain, Tripti Mahara, and Kuldeep Narayan Tripathi. (2020). A Survey of Similarity Measures for Collaborative Filtering-Based Recommender System. Springer, pp.343-352. https://doi.org/10.1007/978-981-15-0751-9_32.

[7] Xu, Lei; Jiang, Chunxiao; Chen, Yan; Ren, Yong and Liu, K. J. Ray (2018). User Participation in Collaborative Filtering-Based Recommendation Systems: A Game Theoretic Approach. IEEE Transactions on Cybernetics, 1–14. http://doi:10.1109/TCYB.2018.2800731.

[8] Kang Liu, Feng Xue, Xiangnan He, Dan Guo and Richang Hong. (2022). Joint Multi-Grained Popularity-Aware Graph Convolution Collaborative Filtering for Recommendation. IEEE. 14(8), pp.1-13.

[9] Sunny Sharma; Vijay Rana and Manisha Malhotra; (2021). Automatic recommendation system based on hybrid filtering algorithm. Education and Information Technologies. http://doi:10.1007/s10639-021-10643-8.

[10] Tao Hai, Jincheng Zhou, Ye Lu, Dayang N.A. Jawawi, Anurag Sinha, Yash Bhatnagar and Noble Anumbe. (2023). Posterior probability and collaborative filtering based Heterogeneous Recommendations model for user/item Application in use case of IoVT. Elsevier, pp.1-27.

[11] Son, Jieun and Kim, Seoung Bum (2017). Content-Based Filtering for Recommendation Systems Using Multiattribute Networks. Expert Systems with Applications, S0957417417305468–. http://doi:10.1016/j.eswa.2017.08.008.

[12] Yassine Afoudi; Mohamed Lazaar and Mohammed Al Achhab; (2021). Hybrid recommendation system combined content-based filtering and collaborative prediction using artificial neural network. Simulation Modelling Practice and Theory. http://doi:10.1016/j.simpat.2021.102375.

[13] Yadalam, Tanya V; Gowda, Vaishnavi M; Kumar, Vanditha Shiva; Girish, Disha and M, Namratha (2020). 5th International Conference on Communication and Electronics Systems (ICCES) - Career Recommendation Systems using Content based Filtering. 660–665. http://doi:10.1109/ICCES48766.2020.9137992.

[14] Satapathy, Suresh Chandra; Bhateja, Vikrant and Das, Swagatam (2019). Smart Intelligent Computing and Applications Volume 105 (Proceedings of the Second International Conference on SCI 2018, Volume 2) || Content-Based Movie Recommendation System Using Genre Correlation., 10.1007/978-981-13-1927-3(Chapter 42), 391–397. http://doi:10.1007/978-981-13-1927-3_42.

[15] Badriyah, Tessy; Azvy, Sefryan; Yuwono, Wiratmoko and Syarif, Iwan (2018). International Conference on Information and Communications Technology (ICOIACT) - Recommendation system for property search using content-based filtering method, 25–29. http://doi:10.1109/ICOIACT.2018.8350801.

[16] Gou, A. (2025). Cloud-Computing-Enabled Transformer Architecture for the Design of Functional Clothing Structures. Informatica, 49(11), 207–218. https://doi.org/10.31449/inf.v49i11.6763.

[17] Wang, Donghui; Liang, Yanchun; Xu, Dong; Feng, Xiaoyue and Guan, Renchu (2018). A Content-Based Recommender System for Computer Science Publications. Knowledge-Based Systems, S0950705118302107–. http://doi:10.1016/j.knosys.2018.05.001.

[18] Turnip, Rudolf; Nurjanah, Dade and Kusumo, Dana Sulistyo (2017). IEEE Conference on e-Learning, e-Management and e-Services (IC3e) - Hybrid recommender system for learning material using content-based filtering and collaborative filtering with good learners' rating. 61–66. http://doi:10.1109/IC3e.2017.8409239.

[19] Sahraoui Dhelim, Nyothiri Aung, Mohammed Amine Bouras, Huansheng Ning and Erik Cambria. (2022). A Survey on Personality-Aware Recommendation Systems. Springer, pp.1-17.

[20] Hengyun Lia Bruce and Yub Gang Lic Huicai Gao. (2023). Restaurant survival prediction using customer-generated content: An aspect-based sentiment analysis of online reviews. elsevier, pp.1-33. https://doi.org/10.1016/j.tourman.2022.104707.

[21] Li, Man; Wen, Luosheng and Chen, Feiyu (2021). A novel Collaborative Filtering recommendation approach based on Soft Co-Clustering. Physica A: Statistical Mechanics and its Applications, 561, 125140–. http://doi:10.1016/j.physa.2020.125140.

[22] Feng, Liang; Zhao, Qianchuan and Zhou, Cangqi (2019). Improving Performances of Top-N Recommendations with Co-clustering Method. Expert Systems with Applications, 113078–. http://doi:10.1016/j.eswa.2019.113078.

[23] Gabriel Frisch, Jean-Benoist Legern and Yves Grandvalet. (2022). Co-clustering for fair recommendation. Springer, pp.1-32.

[24] Taushif Anwar and V. Uma; (2021). Comparative study of recommender system approaches and movie recommendation using collaborative filtering. International Journal of System Assurance Engineering and Management. http://doi:10.1007/s13198-021-01087-x.

[25] Sonu Airena and Jitendra Agrawal. (2023). Movie Recommender System Using Parameter Tuning of User and Movie Neighbourhood via Co-Clustering. Elsevier, pp.1176-1183. https://creativecommons.org/licenses/by-nc-nd/4.0.

[26] Taushif Anwar1 · V. Uma1 · Md. Imran Hussain1 · Muralidhar Pantula. (2022). Collaborative filtering and kNN based recommendation to overcome cold start and sparsity issues: A comparative analysis. Springer. (.), pp.1-22. [Online]. Available at: https://doi.org/10.1007/s11042-021-11883-z

[27] Honda, Katsuhiro; Ubukata, Seiki and Notsu, Akira (2018). A comparative study on effects of some exclusive conditions in fuzzy co-clustering for collaborative filtering. Journal of Ambient

Intelligence and Humanized Computing. http://doi:10.1007/s12652-018-0789-0.

[28] Atasu, Kubilay; Parnell, Thomas; Dunner, Celestine; Vlachos, Michail and Pozidis, Haralampos (2017). 46th International Conference on Parallel Processing (ICPP) - High-Performance Recommender System Training Using Co-Clustering on CPU/GPU Clusters, 372–381. http://doi:10.1109/ICPP.2017.46.

[29] Das, Joydeep; Banerjee, Madhurima; Mali, Kalyani and Majumder, Subhashis (2019). International Conference on Information Technology (ICIT) - Scalable Recommendations Using Clustering Based Collaborative Filtering, 279–284. http://doi:10.1109/ICIT48102.2019.00056.

[30] Honda, Katsuhiro; Hayashi, Issei; Ubukata, Seiki; Notsu, Akira (2020). International Symposium on Community-centric Systems (CcS) - A Comparative Study on Three-mode Fuzzy Co-clustering Based on Co-occurrence Aggregation Criteria, 1–6. http://doi:10.1109/ccs49175.2020.9231431.

[31] MivieLens Dataset. Retrieved from https://grouplens.org/datasets/movielens/

[32] Ricci, Francesco; Rokach, Lior; Shapira, Bracha and Kantor, Paul B. (2011). Recommender Systems Handbook || Introduction to Recommender Systems Handbook., 10.1007/978-0-387-85820-3(Chapter 1), 1–35. http://doi:10.1007/978-0-387-85820-3_1.

[33] L. Ungar and D. P. Foster, "A formal statistical approach to collaborative filtering," in Confernce on Automated Learning and Discovery Pittsburgh, PA, 1998

[34] Delgado, Joaquin and Ishii, and Naohiro, "Memorybased weighted majority prediction for Recommender Systems," in SIGIR Workshop Recomm. Syst. Citeseer: Citeseer, 1999.

[35] R. Paul, I. Neophytos, S. Mitesh, B. Peter, and R. John, "GroupLens: an open architecture for collaborative filtering of netnews," in ACM conference on Computer supported cooperative work Chapel Hill, North Carolina, USA: ACM, 1994.

[36] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, "Item-based collaborative filtering recommendation algorithms," in 10th international conference on World Wide Web Hong Kong, Hong Kong: ACM, 2001. http://doi:10.1145/371920.372071.

[37] X. Luo, M. Zhou, S. Li, Y. Xia, Z. You, Q. Zhu, and H. Leung, "An Efficient Second-Order Approach to Factorize Sparse Matrices in Recommender Systems," IEEE Transactions on Industrial Informatics, vol. 11, pp. 946-956, 2015. http://doi:10.1109/TII.2015.2443723.

[38] X. Su and T. M. Khoshgoftaar, "Collaborative filtering for multi-class data using belief nets algorithms," in 18th IEEE International Conference on Tools with Artificial Intelligence: IEEE, 2006, pp. 497-504. http://doi:10.1109/ICTAI.2006.41.

[39] T. Hofmann, "Collaborative filtering via gaussian probabilistic latent semantic analysis," in 26th annual international ACM SIGIR conference on Research and development in informaion retrieval Toronto, Canada: ACM, 2003. http://doi:10.1145/860435.860483.

[40] B. M. Marlin, "Modeling user rating profiles for collaborative filtering," in Advances in neural information processing systems, 2003, p. None.

[41] P. S. Chakraborty, "A scalable collaborative filtering-based recommender system using incremental clustering," in International Conference on Advance Computing IEEE, 2009, pp. 1526-1529. http://doi:10.1109/iadcc.2009.4809245.

[42] D. Zhang, C. H. Hsu, M. Chen, Q. Chen, N. Xiong, and J. Lloret, "Cold-Start Recommendation Using BiClustering and Fusion for Large-Scale Social Recommender Systems," IEEE Transactions on Emerging Topics in Computing, vol. 2, pp. 239-250, 2013.

[43] Meesala, Y. V. N., Parida, A. K., & Naik, A. (2024). Simultaneous Clustering and Feature Selection Using Social Group Optimization with Dynamic Threshold Setting for Microarray Data. *Informatica*, 48(23), 199–218. https://doi.org/10.31449/inf.v48i23.7019

[44] Yongjian Dong. (2024). Particle Swarm Optimization in Gene Expression Profile Clustering. Informatica. 48, p.77–88, https://doi.org/10.31449/inf.v48i16.6360