IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 14, No. 6, December 2025, pp. 4865~4877
ISSN: 2252-8938, DOI: 10.11591/ijai.v14.i6.pp4865-4877

Solving sparsity and scalability problems for book recommendations on e-commerce

Muhammad Ichsanudin[1], Bevina Desjwiandra Handari[1], Bambang Dwi Wijanarko[2], Gatot Fatwanto Hertono[1]
[1]Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Indonesia, Depok City, Indonesia
[2]Bina Nusantara University, Semarang, Indonesia

Article Info

Article history:

Received Mar 12, 2025
Revised Oct 27, 2025
Accepted Nov 8, 2025

Keywords:

Collaborative filtering
Hierarchical density-based spatial clustering
Scalability
Singular value decomposition
Sparsity

ABSTRACT

This study proposed a hierarchical density-based spatial clustering of applications with noise (HDBSCAN) and randomized singular value decomposition (RSVD) collaborative filtering (CF) method to overcome sparsity and scalability problems for book recommendations on e-commerce. CF is an information retrieval system that assumes a user has the same interest in an object as other users have in the past. When handling large volumes of data, sparsity problems can arise, where finding a similarity relation of user preferences results from a small assessment of an object by users. The scalability is the increased computation of an algorithm caused by increased users or objects, which makes recommendations take longer to form, therefore making them less accurate. HDBSCAN is a density-based clustering method that simplifies the hierarchical arrangement of the most significant clusters for extraction to group users in the same cluster. RSVD is a linear dimension reduction method that breaks a matrix into three sub-matrices by reconstructing the size of that matrix without removing its dominant part, especially for cluster result matrices. The HDBSCAN-RSVD-CF model reduced the root mean squared error (RMSE) by 21.83%, being 3793.73 seconds faster than the CF model. It also performed very well compared to both RSVD-CF and HDBSCAN-CF.

Corresponding Author:

Gatot Fatwanto Hertono
Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Indonesia
Pondok Cina, Beji, Depok City, West Java 16424, Indonesia
Email: gatot-f1@ui.ac.id

1. INTRODUCTION

Currently, companies are competing to improve their shopping experience, especially in e-commerce for books such as Amazon and eBay, to increase the possibility of a buyer buying more books. Nowadays, as demand and volume for users of electronic commerce increases, especially for books, the data that must be processed increases in size, and customer assessments are increasingly arbitrary in variants of book ratings, causing sparsity and scalability [1]. Sparsity is a problem in searching for a relationship of similarity between user preferences due to the user's lack of assessment of an object [2]. Scalability is the increase in the computation of an algorithm caused by an increase in the number of users or objects (books), which lengthens the recommendation formation process [3]. These two problems are quite common in recommendation systems, causing less accurate and more biased recommendation results.

Besides book recommendations, there are many other use cases in industry that use recommendation systems. The use case includes movie recommendation [4]–[14], food recommendations [15], medical purposes such as diabetes diagnosis [16], music recommendations, and market decision for merger and

acquisition company [17]. All these recommendation systems' use cases show promise in helping for decision making, diagnosis and interest preferences. In this research focus, book recommendation use case will be done using open sources data from Kaggle – GoodReads, to show the performance of limited novelty methods performance called hierarchical density-based spatial clustering of applications with noise (HDBSCAN)-randomized singular value decomposition (RSVD)-collaborative filtering (CF) model in coping with sparsity and scalability problem.

A recommendation system is a model that provides a predicted recommendation for a product's preferences to users. In this case, the product is a book, and the users will be the customer buying that book. Some related works such as efficient deep matrix factorization (EDMF) with review feature learning for industrial recommender system [18], confidence-aware recommender model (CARM) via review representation learning and historical rating behavior in the online platforms [19], and multi-perspective social recommendation method with graph representation learning [20] are discussed here.

The recommendation system is categorized into two types of filtering: content-based and CF. Content-based filtering focuses on user profile preferences (e.g., books) and item descriptions to recommend items that are most correlated with other items that users have highly rated in the past. Meanwhile, CF considers variations in criteria such as user preferences, activities, and habits, then recommends an object based on similarity relations with other users [4].

CF is divided into two categories: memory-based and model-based. Memory-based is a heuristic approach, such as correlation analysis and vector similarity, that looks for user profiles that resemble active user profiles so that a recommendation can be determined. The model-based approach uses a learning model by utilizing data containing recommendation assessment parameters, which are then applied to provide recommendation predictions [5]. Memory-based and model-based filtering methods can be combined to generate recommendations based on user rating predictions for books. In this case, memory-based filtering calculates the weighting of a correlation value between users using Pearson correlation weighting, and model-based filtering modifies the dataset with the HDBSCAN and RSVD models. This combination will estimate the rating of a book that users are expected to appreciate.

We propose a method of book recommendation system that solves sparsity and scalability problems in CF applied to the electronic book trading datasets and contributes to CF literacy in the form of HDBSCAN and RSVD. The proposed HDBSCAN-RSVD-CF hybrid model is the first approach that integrates HDBSCAN clustering and RSVD within a CF framework to address scalability and sparsity issues in large-scale recommender systems. This research also aims to determine the performance of HDBSCAN and RSVD in CF when compared to HDBSCAN-CF, RSVD-CF, and single CF. The model's performance will be evaluated using density-based cluster validation (DBCV) and root mean squared error (RMSE) to determine the optimal number of clusters from the HDBSCAN method. This research uses a dataset provided by the Kaggle site [21] which consists of book information and ratings from the GoodReads dataset.

The rest of the paper is divided into the following sections: section 2 provides the literature review. Section 3 presents the proposed method. Section 4 presents the results and discussion. Finally, the paper ends with a conclusion is section 5.

2. LITERATURE REVIEW

The research of recommendation system is quite broad. To easily relate to current study, the literature will then be grouped at their focus for overcoming problems in recommendation system. There is research that focuses on overcoming sparsity, such as [8], [9], [15]. Other research focuses on improving CF performance by enhancing the similarity method, such as [10]–[13], [22], [23]. There is also research that focuses on using a clustering or matrix decomposition method to improve CF performance, such as [1], [14], [17]. Finally, there is also some research focused on improving the contribution to CF [18]–[20].

Yet, the research journey to overcome issues of sparsity and scalability using clustering (a grouping method) or matrix decomposition in CF models was undertaken in early 2018. Those studies showed a promising result in overcoming the sparsity and scalability issues regarding the CF recommendation systems model. Initial studies such as the *k*-means cluster and SVD dimension reduction methods are used on a combined memory-model-based CF (hybrid) [4]. In this research, the cosine similarity method calculates data similarity calculations for MovieLens 1M and MovieLens 10M film data. This research has an increased RMSE of ±20% compared to memory-based CF with the *k*-nearest neighbors (*k*-NN) method. The other hybrid CF model in 2018 uses the ontology method approach and singular value decomposition (SVD) dimension reduction, applied to MovieLens and Yahoo! Webscope [5]. The results show that the proposed method overcomes sparsity and scalability problems in film data. The results from applying SVD, expectation maximization, and ontology provide mean absolute error (MAE) ±13% better performance than CF with Pearson nearest. In 2020, both the DBSCAN clustering approach and linear discriminant analysis

(LDA) dimension reduction were used for memory-model-based CF on cloud systems to create a cloud recommender system with the MovieLens dataset [6]. Hybrid CF by combining the clustering method with Slope One algorithm is used to enhance the prediction of item ratings by k-means using the MovieLens 1M dataset [7]. Based on the excellent performance of the hybrid methods from previous research, a method called HDBSCAN-RSVD-CF is introduced to address sparsity and scalability issues, including HDBSCAN, which is inspired by DBSCAN [6] and RSVD that is inspired by SVD in [4], [5].

3. METHOD

The proposed method, which combines HDBSCAN and RSVD methods, will provide book recommendations to buyers by first applying the HDBSCAN clustering technique to group data with parameter optimization by DBCV, followed by the RSVD method to reduce the large data into smaller forms due to fusing its dominant terms of features. The Pearson correlation method is applied to the reduced matrix. The results will reveal a level of similarity between buyers as a basis for recommending a book to buyers. The model workflow is shown in Figure 1.

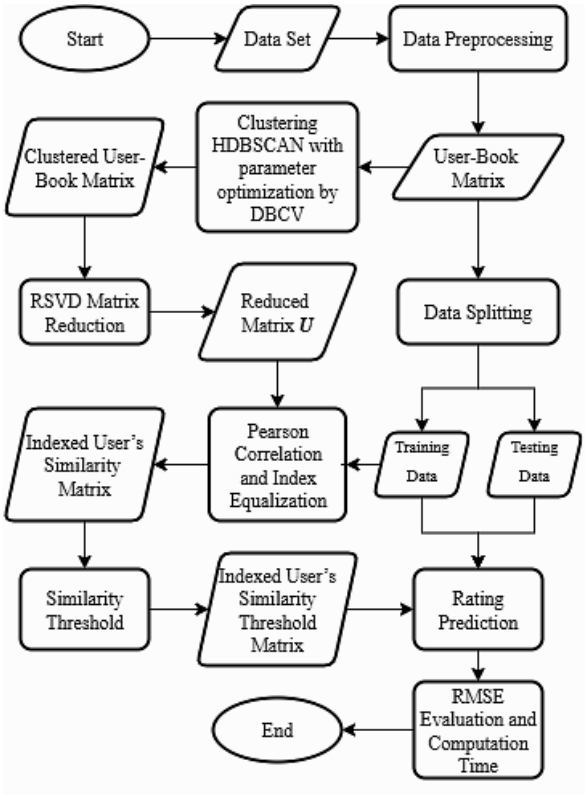

Figure 1. HDBSCAN-RSVD-CF recommendation system model workflow

3.1. Data set

The dataset is from Kaggle for electronic book trading [21]. Data collection is from 2008 to 2023 and divided into two parts: the dataset with the details of a book which contains 19 columns (id, name, authors, CountsOfReview, description, ISBN, language, PublishDay, PublishMonth, PublishYear, Publisher, Rating, RatingDist1, RatingDist2, RatingDist3, RatingDist4, RatingDist5, RatingDistTotal, and pagesNumber) that contains ±5,000,000 entries, and a second set containing detailed book ratings by users (id, name, and rating) that contains ±1,100 users with user ratings of more than 360,000 books. From these two datasets, data preprocessing is applied, such as dropping unnecessary values (e.g., null data and unknown symbols), harmonize upper- and lower-case letters, and adjusting table columns with id and name columns as the key, results in a user-books matrix with the user ID as the index, columns for the book names, and rating as the contents of the matrix. The matrix has 3336 users as rows and 7591 columns. The data with the value not a number (NaN) is replaced by zero as a neutral rating number.

3.2. Hierarchical density-based spatial clustering of applications with noise

The clustering of the user-book matrix used HDBSCAN, where the most optimal clustering parameters were evaluated by hierarchical hyperparameter tuning method, where parameters were optimized in sequence using DBCV evaluation. The HDBSCAN model workflow is in Figure 2. According to the HDBSCAN documentation by McInnes *et al*. [24], the two primary parameters that significantly affect the optimization of HDBSCAN are min_cluster_size (the minimum number for the formation of a cluster) and min_samples (the minimum point in the core point environment) [24]. The min_cluster_size helps to set a minimum number as the minimum number of a group to be created. Because the model works in a density approach (neighborhood), choosing the right min_cluster_size will define how well the group is to be formed, especially when facing sparse data like an e-commerce book. If the min_cluster_size is set to be a higher value, the outcome will be a smaller number of groups, which in sparse data might lead to bias grouping. The min_samples value is then used to calculate the distance between a point (data) to its nearest neighbor. In other terms, min_samples tells how conservative the clustering will be. The higher the value of min_sample, the more conservative the clustering, which can result in an increasing number of outlier points (data) that are flagged as -1 in the HDBSCAN result. Therefore, it is very necessary to pay attention to the selection of these two hyperparameters to avoid a bias and a very conservative result of the HDBSCAN result.

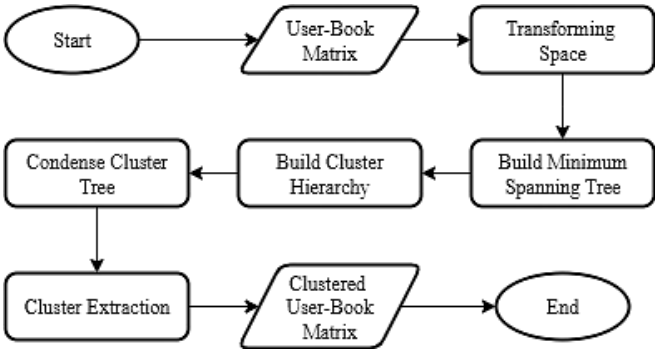

Figure 2. HDBSCAN model workflow

In HDBSCAN, clusters are divided into two groups: noise and clusters based on density from the user-books matrix. A cluster with a value of -1 represents a cluster with members that are considered noise [24]. The cluster of value -1 can be called a group of users that has low similarity of interest from one to another. However, in this research, the data from the noise cluster is still considered when weighting the predicted rating of a book recommended to active users since the cluster contains half more data. It is also a valuable source to see how well the HDBSCAN-RSVD-CF helps the HDBSCAN model to correlate users with low similarity of interest in the kind of books.

The clustering process replaces empty values with a zero value as a neutral value from the data distribution. This step is important since the HDBSCAN algorithm cannot weight clusters with gaps in the data, especially in the Python program with the HDBSCAN library [25], and regarding HDBSCAN [24]. The DBCV weighs the parameters based on the density of the HDBSCAN model and gives it a score from -1 to 1, which is appropriate to the goodness of the cluster result. The higher the DBCV score, the better the clustering result. Clustering results in groups of users with similar book interests, allowing the recommendation system to reduce computation by considering only users within the same cluster instead of all users. More details on DBCV can be found in [26].

3.3. Randomized singular value decomposition

The purpose of RSVD is for matrix reduction, where a comparison will be carried out to select the best *n*-component hyperparameters. This *n*-component will affect the accuracy of RMSE and run time of the model. The RSVD model workflow is in Figure 3. The process is as follows:

i) RSVD is a randomized method to reconstruct high-dimensional matrices into smaller matrices, followed by the SVD process, which splits a matrix into three matrix components. The result is three matrix components from the main matrix $A_{m \times n}$, so that $A_{m \times n} = U_{m \times k} \times \Sigma_{k \times k} \times V^{T}_{k \times n}$, where U and V are orthonormal matrices, and Σ is a diagonal and non-negative matrix.

ii) The RSVD process can be applied if matrix A has a low-rank structure, and it is a very efficient matrix decomposition algorithm based on random sampling theory. It is also called the randomized numerical method [27]. The k values for Σ, indicates the eigenvalues from matrix A.

iii) The U matrix builds a recommendation system, and the matrix dimensions are reduced according to the number of n-component values (singular values) selected from the k values of Σ, so the recommendation system does not use too much work in progress [28].

iv) Consider the matrix $A_{m\times n} = U_{m\times k} \times \Sigma_{k\times k} \times V^T{}_{k\times n}$. Then, for a selection of the number of n-component values represented as t in $\Sigma_{k\times k}$, where $t < k$ and k represents the rank value of the matrix $A_{m\times n}$. Therefore, the n-component (t) value is between 0—rank of the matrix, or the smallest row/column size in the matrix $A_{m\times n}$. The rank of matrix A is the number of linearly independent column spaces. The matrix $U_{m\times k}$ will experience a dimension reduction, from $U_{m\times k}$ becomes the reduced matrix $U_{m\times t}$, see Figure 4.

v) In the reduced matrix U, the rows represent users within the same cluster, while the columns represent an aggregation of books, focusing only on the most dominant parts. The element $u_{p,t}$ represents the book rating by user p for the t-th aggregated books.

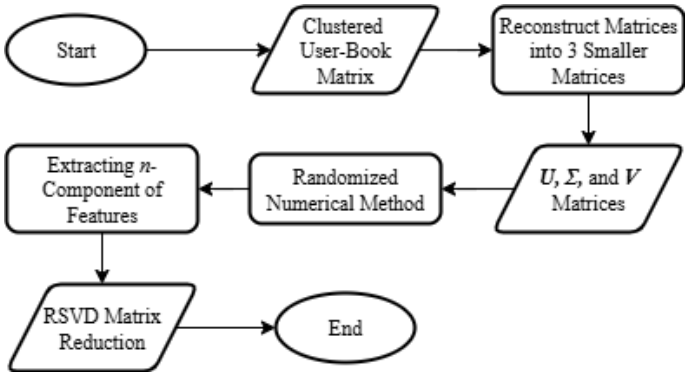

Figure 3. RSVD model workflow

$$\begin{pmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,k} \\ u_{2,1} & u_{2,2} & \cdots & u_{2,k} \\ \cdots & \cdots & \cdots & \cdots \\ u_{m,1} & u_{m,2} & \cdots & u_{m,k} \end{pmatrix} \rightarrow \begin{pmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,t} \\ u_{2,1} & u_{2,2} & \cdots & u_{2,t} \\ \cdots & \cdots & \cdots & \cdots \\ u_{m,1} & u_{m,2} & \cdots & u_{m,t} \end{pmatrix}$$

Figure 4. Illustration of the reduced matrix U

The determination of parameters involves the n-component, which is set to 50%, 60% to 100% of the rank from each clustered user-book matrix. This corresponds to the active user's cluster. The n-component retrieval is applied only when there are at least 10 users in a cluster; otherwise, all users in the cluster are included.

3.4. Splitting data

The training and testing data division was carried out with an 80:20 division sequentially on the user-books matrix. Users in the testing data are called active users, as they will receive recommendations from the model. Meanwhile, the training data consists of inactive users who generate predicted book ratings. The process was carried out five times sequentially, or 5-fold cross-validation.

3.5. Pearson correlation coefficient and index equalization

The Pearson correlation coefficient (PCC) method is applied to users in the index (row) of the matrix U to obtain the users' similarity matrix. Based on this matrix, the Pearson correlation method is also applied between active users (users from testing data) and inactive users (training data) to generate the indexed similarity matrix. Finally, active users can be identified from this matrix to have book

recommendations later that align with their interests through the predicted book ratings. The initial process of establishing the user's similarity matrix uses the reduced matrices U on a given cluster created by the RSVD process. Use PCC defined in (1) between active user (u) and another user (u') from the same cluster as follows:

$$PCC(u,u')=\frac{\sum_{i\in I}(r_{u,i}-\bar{r}_u)(r_{u',i}-\bar{r}_{u'})}{\sqrt{\sum_{i\in I}(r_{u,i}-\bar{r}_u)^2}\times\sqrt{\sum_{i\in I}(r_{u',i}-\bar{r}_{u'})^2}} \quad (1)$$

Where $r_{u,i}$ and $r_{u',i}$ The book rating given by two users u and u' to the i-th book, correspondingly. The $\bar{r}_u$ and $\bar{r}_{u'}$ denotes the average book rating by users u and u', and $I = I_u \cap I_{u'}$ denotes the books rated by user u and u', respectively. The value $PCC(u,u') \in [-1,1]$ shows the similarity value between users u and u' correspondingly. When the value of PCC is close to 1 or -1, the users show a high positive or negative correlation.

After obtaining the users' similarity matrix, the index equalization process equates the user's "ID" index from the user similarity matrix with the training data with the same index (the user's "ID"). This process removes rows from the user "IDs" not in the training data and will not appear in the results. It is important not to delete the user "ID" in the matrix column because this column, which still contains the user "ID" from the testing data, will be used as a reference for inactive users with similarity to active users that will receive recommendations.

3.6. Similarity threshold

The similarity threshold will provide a list of people with similar interests or not. After getting the indexed users' similarity matrix, the PCC value will be set with a threshold to determine if the user has a value above the limit, in which case it matches active users. The matrix is called the indexed users' similarity threshold matrix. Establishing a similarity value threshold is important in increasing prediction accuracy for an active user. The higher the restriction value, the higher the accuracy [29].

3.7. Book rating prediction

The predictions for active users utilize three main matrices: the user similarity, the active user-book, and the inactive user-book matrix. The prediction process is carried out iteratively, one by one, for each active user on testing data using (2).

$$\hat{r}_{a,i}=\bar{r}_a+\frac{\sum_u^n(PCC_{a,u}\times|r_{u,i}-\bar{r}_u|)}{\sum_u^n PCC_{a,u}} \quad (2)$$

Where $\hat{r}_{a,i}$ is the predicted rating of the i-th book by the a-th active user, $\bar{r}_a$ is the average rating of the book rated by the a-th active user, $PCC_{a,u}$ is the Pearson correlation assessment of the a-th active user to the u-th inactive user, $r_{u,i}$ is the assessment by the u-th inactive user of the i-th book, and $\bar{r}_u$ is the average rating of the book from u-th inactive users [30]. After obtaining the predicted values, the average values will be calculated to determine the RMSE of the model.

4. RESULTS AND DISCUSSION

We present the results and discussion of the implementation of the HDBSCAN-RSVD-CF model. The RMSE values are the average of five quintile results from the model's rating predictions for active users, compared to the actual book ratings they have rated. The model records the running time values after forming the user-books matrix and dividing the dataset into five quintiles.

The implementation of HDBSCAN selects several important parameters in the form of min_cluster_size and min_samples to obtain clustered data [24]. The process involves hierarchical hyperparameter tuning method, where parameters were optimized sequentially rather than simultaneously, to determine the best min_cluster_size parameter value see Table 1 and the best min_samples parameter value shows in Table 2. The quality of clustering results is assessed by DBCV.

According to the DBCV, the value closer to 1 is better. The HDBSCAN parameter is set with a value of min_cluster_size of 2 and min_samples of 1. In the process, there is a cluster with the value of -1, indicating data considered as noise, which is still included in predicting rating values for active users. The result of this stage is a clustered user-book matrix.

Min_cluster_size is the minimum number for the formation of a cluster, where the value is an integer in the interval of [2, inf). While the process of fine tuning the min_cluster_size, the min_samples is

set to default, which, in the documentation, will be the same value as the min_cluster_size [24]. The result of Table 1 shows that the best hyperparameters for min_cluster_size is 2, where the number of clusters is 82, and has the largest DBCV score (0.0550). The larger the min_cluster_size, the lower the DBCV score and the decay of cluster result. This result of Table 1 justified the effect of choosing higher min_cluster_size. This will result in smaller clusters which indicate biased results of grouping the data. On the other hand, choosing the smallest value for min_cluster_size still results in a low DBCV score (the nature DBCV value is in [-1,1]), which indicates the data is very sparse.

The min_cluster_size of 2 will then be carried to find the best min_samples by the hierarchical hyperparameter tuning method that sequentially optimized each parameter, results in min_samples is 1. Min_samples is the minimum point in the core point of some environment (group), where the value is an integer number in [1, inf). From Table 2, the iteration process shows that the higher the min_samples, the more the result of DBCV score and number of clusters will decay. This result shows that choosing more core points for the model will result in higher distances calculated by the nearest neighbor algorithm between points, which indicates the increasing number of outliers in some groups of clusters.

The results of Tables 1 and 2 show that the model facing the data is too sparse to be grouped. The best DBCV evaluation shows a low score (0.0851), which might indicate insufficient performance when handling the sparsity and scalability problem. To address this issue, RSVD is applied to each cluster to suppress noise and facilitate the selection of the most representative component for modeling.

Table 1. Parameter iteration table of min_cluster_size

Iteration	min_cluster_size	DBCV	Number of clusters
1	2	0.0550	82
5	6	0.0385	15
10	15	0.0088	5
11	20	0.0	3
12	30	0.0	1

Table 2. Parameter iteration table of min_samples with min_cluster_size equal 2

Iteration	min_samples	DBCV	Number of clusters
1	1	0.0851	171
5	5	0.0240	25
10	10	0.0112	11
11	15	0.0088	7
12	20	0.0021	5
13	30	0.0	4

4.1. Matrix reduction using randomized singular value decomposition

The RSVD algorithm is applied to a clustered user-book matrix over a particular cluster for dimension reduction. The selection of RSVD parameters is performed by selecting the number of *n*-components in the singular matrix, which reduces the matrix dimensionality according to the selected *n*-component value with a value smaller than or equal to the smallest row/column size (rank of the matrix). The selection is performed through iterative experiments by comparing the rating value prediction at the final stage of the HDBSCAN-RSVD-CF algorithm.

4.2. Implementation of similarity and index equalization

After obtaining the *U* matrix, the values in the *U* matrix are applied with the Pearson correlation method concerning the index/row, which is the "ID" of the user will be transformed into a user similarity matrix. The process is continued by replacing the NaN data with zero. Next, the process of equalizing the user "ID" index in the similarity matrix with the training data is implemented. This equalization process removes all user "ID" values that are not present in the training data.

4.3. Evaluation of similarity threshold

The average method was used to obtain similarity threshold values with respect to RMSE and computing time. The closest value to the average RMSE and computing time is selected as the similarity threshold. This threshold will be used in the next process. The average RMSE value in Table 3 is 0.8537, while the average working model is 3831.86 seconds. Based on both averages, the closest similarity threshold value is 0.4 (the similarity limit parameter is 0.4).

Table 3. Comparative determination of similarity limit values

Similarity threshold values	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	Mean
Average RMSE	0.8485	0.8488	0.8499	0.8502	0.8512	0.8523	0.8559	0.8576	0.8597	0.8631	0.8537
Time (second)	7934.13	6575.96	5432.49	4817.29	3940.52	2951.20	2167.39	1819.41	1588.83	1091.44	3831.86

4.4. Evaluation of *n*-component retrieval in HDBSCAN-RSVD-CF

The result of DBCV for the HDBSCAN hyperparameter is combined with the similarity threshold value 0.4 as a base for building HDBSCAN-RSVD-CF model. The average of RMSE metric and the average length of time of the model from five quantiles are represented in Table 4. From the results of Table 4, it can be shown that applying RSVD tends to help HDBSCAN handle sparsity and scalability, which is reflected by low errors and faster runtime compared to the simple CF in Table 3. This result can be achieved by reducing noise which sparse data common problems from combining clustering by HDBSCAN and matrix decomposition with RSVD. The clustering of HDBSCAN will first group every user that has similar interaction to a book, which in this case is the rating given by users which results in a group that has a similar taste of book liking. After user grouping, there is some noise caused by the sparsity of the data, which Tables 1 and 2 show. RSVD is then applied to selectively choose the best of the best features that represent the user rating to a book with the purpose of eliminating noise that caused the bias in recommendation results. With the combination of these 2 methods, HDBSCAN and RSVD show excellent performance in handling sparsity and scalability.

Table 4. HDBSCAN-RSVD-CF model evaluation comparison

Model	HDBSCAN-RSVD (50%)-CF	HDBSCAN-RSVD (60%)-CF	HDBSCAN-RSVD (70%)-CF	HDBSCAN-RSVD (80%)-CF	HDBSCAN-RSVD (90%)-CF	HDBSCAN-RSVD (100%)-CF
Average RMSE	0.6977	0.70008	0.6975	0.7125	0.6654	0.7529
Time (Second)	474.59	382.24	326.71	255.73	146.78	100.24

Furthermore, the six models have an average time of 281.05 seconds, so 146.78 seconds is relatively fast compared to 281.05 seconds. From these 2 considerations, the HDBSCAN-RSVD (90%)-CF will be chosen as the best benchmark in this research. The larger the size of the *U* matrix, the greater the computing time required for the HDBSCAN-RSVD-CF model. However, in this case, the computing time decreases as the part of the *n*-component retrieval increases. This condition is closely related to the similarity results between active and inactive users. The larger the *n*-component value taken, the more entries in the *U* matrix approach zero, which reduces the process of forming similarity values between users and ultimately causing fewer users to meet the similarity limit value of 0.4. It is possible to analyze the results from the sample by taking 50% of the total *n*-components compared to 90%, as presented in Figures 5 and 6.

	0	1	...	1408	1409
0	0.059290	0.014928	...	0.008007	0.004523
1	0.014575	-0.017668	...	0.010328	-0.016188
2	0.002546	0.004086	...	0.009727	-0.000587
3	0.070577	0.042204	...	0.000609	0.002258
4	0.006928	-0.016324	...	0.002523	0.005150
...	...	...	...	...	...
2815	0.001489	0.001868	...	0.005779	0.002280
2816	0.015798	-0.024596	...	0.043969	-0.024703
2817	0.011367	0.013874	...	0.014411	-0.010690
2818	0.004084	-0.002978	...	0.067837	0.006911
2819	0.006109	-0.002513	...	0.014040	-0.004487

2820 rows × 1410 columns

Figure 5. The *U* reduction matrix sample taking 50% user rank in cluster -1

	0	1	...	2536	2537
0	0.059290	0.014928	...	4.065387e-18	2.927946e-17
1	0.014575	-0.017668	...	3.956227e-17	-3.564999e-17
2	0.002546	0.004086	...	-6.323353e-17	1.301011e-16
3	0.070577	0.042204	...	-1.527768e-17	-4.123426e-17
4	0.006928	-0.016324	...	-1.605653e-02	-4.244358e-02
...	...	...	...	...	...
2815	0.001489	0.001868	...	-5.288460e-02	-3.319518e-02
2816	0.015798	-0.024596	...	4.423545e-17	3.512815e-17
2817	0.011367	0.013874	...	3.859760e-17	-2.851452e-17
2818	0.004084	-0.002978	...	-7.632783e-17	7.132017e-18
2819	0.006109	-0.002513	...	-2.257230e-02	-1.095063e-02

2820 rows × 2538 columns

Figure 6. The *U* reduction matrix sample taking 90% user rank in cluster -1

The values are very close to zero in the reduced matrix *U* leads to a decrease in similarity values between users in the similarity matrix. In this research, the suitability limit parameter was limited to 0.4 for a user with high similarity to an active user. Therefore, in Table 4, where the more *n*-components are taken, the lower the similarity value between active and inactive users. The model considers only a few observations of active users, which is inversely proportional to the size of the *n*-component taken. Table 5 clarifies the results of this analysis. Thus, the large *n*-component value taken in the *U* reduction matrix causes the similarity value to decrease so that fewer active users receive recommendations from the HDBSCAN-RSVD-CF model, such as for 50% of *n*-components taken, giving 1,313 active users, compared to 90%, which yields 226, and this score keeps decreasing. So, from the results of Table 5, it can be concluded that, although the *n*-component value increases, which intuitively causes the execution time to increase, the model computing time decreases because there are fewer users meeting the similarity threshold value, as mentioned in Figures 5 and 6.

Table 5. The number of active users recommended by HDBSCAN-RSVD-CF

Model	HDBSCAN-RSVD (50%)-CF	HDBSCAN-RSVD (60%)-CF	HDBSCAN-RSVD (70%)-CF	HDBSCAN-RSVD (80%)-CF	HDBSCAN-RSVD (90%)-CF	HDBSCAN-RSVD (100%)-CF
Number of active users get recommendations	1,313	1,048	808	591	226	62

4.5. Evaluation of *n*-component retrieval in RSVD-CF

Even though RSVD shows a promised performance, it still has a limitation that will be explained in this sub-chapter. Much like the previous result, when taking a larger number of *n*-component values, the similarity value between users decreases, and fewer active users get recommendations from the model. Table 6 presents the evaluation of *n*-component retrieval from the RSVD model. According to Table 7, it can be observed that higher *n*-component values prevent the model from providing recommendations, especially at 100%. This result shows that RSVD has limitations in choosing *n*-component freely.

Table 6. RSVD-CF model evaluation comparison

Model	RSVD (50%)-CF	RSVD (60%)-CF	RSVD (70%)-CF	RSVD (80%)-CF	RSVD (90%)-CF	RSVD (100%)-CF
Average RMSE	0.5388	0.53505	0.4902	0.3342	0.15701	*NaN*
Time (Seconds)	474.59	133.38	139.83	119.88	110.96	107.22

Table 7. The number of active users recommended by RSVD-CF

Model	RSVD (50%)-CF	RSVD (60%)-CF	RSVD (70%)-CF	RSVD (80%)-CF	RSVD (90%)-CF	RSVD (100%)-CF
Number of active users get recommendations	1,576	1,285	1,027	501	57	0

4.6. Evaluation of comparative results of HDBSCAN-RSVD-CF, RSVD-CF, HDBSCAN-CF, and CF

Table 8 presents the performances of the HDBSCAN-RSVD-CF, RSVD-CF, HDBSCAN-CF, and single CF models. The average RMSE is calculated by averaging the RMSE values of a group of active users (testing data) and comparing the predicted book ratings with their actual ratings for the same book. In this research, simple CF is the base model that will be improved with proposed methods called HDBSCAN and RSVD. Table 8 shows that CF fails to perform well when attempting to overcome sparsity and scalability with RMSE and running time of 0.85128 and 3940.52, respectively. The comparison between all possible models compared to simple CF as the base model can be sorted as follows: HDBSCAN-CF is inferior to simple CF, with RMSE and running time being 27.42% and 12.46% worse, respectively; HDBSCAN-RSVD-CF has a 21.83% lower RMSE and 2684.48% faster computing time; RSVD-CF has a 60.74% lower RMSE and 3287.05% faster computing time.

Table 8 shows that CF fails to perform well. Nevertheless, HDBSCAN-CF had an inferior performance to CF with RMSE, and their running times are 27.42% and 12.46% worse, respectively, relative to the CF model. However, the CF model with the RSVD method outperformed the CF model. The HDBSCAN-RSVD-CF has a 21.83% lower RMSE and a 2684.48% faster computing time compared to the CF model. Finally, RSVD-CF has a 60.74% lower RMSE and a 96.95% computing time compared to the CF model see Figures 7 and 8.

Table 8. HDBSCAN-RSVD-CF model evaluation comparative

Model	HDBSCAN-RSVD (90%)-CF	RSVD (80%)-CF	HDBSCAN-CF	CF
Average RMSE	0.6654	0.3342	1.0847	0.85128
Time (Seconds)	146.78	119.88	4431.79	3940.52

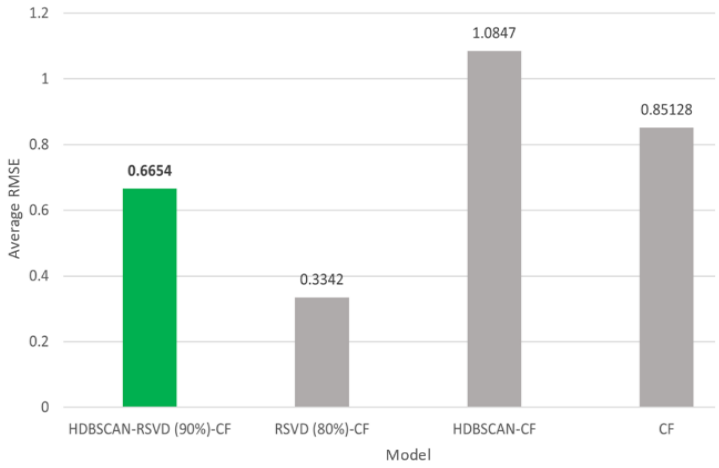

Figure 7. Comparison of average RMSE among the CF models

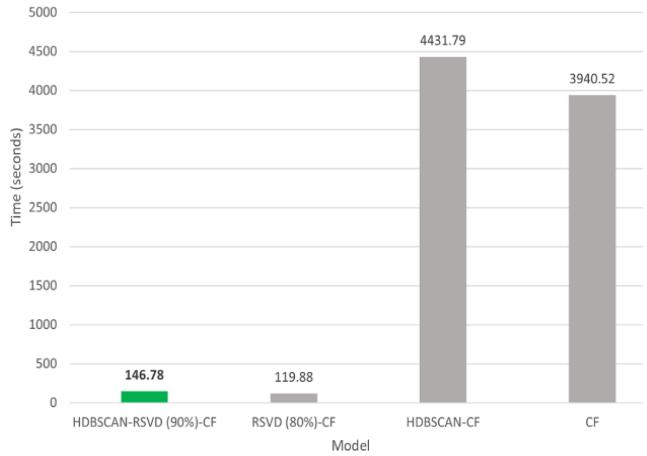

Figure 8. Comparison of computing time among the CF models

5. CONCLUSION

The HDBSCAN-RSVD-CF model demonstrates excellent performance in addressing sparsity and scalability issues. This is evident from the comparison between the HDBSCAN-RSVD-CF and CF models, which shows an improvement of 21.83% in RMSE evaluation and a reduction of 3793.73 seconds in computing time compared to the simple CF model. Furthermore, the application of the HDBSCAN method in the HDBSCAN-RSVD-CF model resolves the limitation of the RSVD-CF model by retrieving the number of n-components from the rank matrix of clustered user-books. The RSVD-CF model is robust and fits especially well for CF model by giving the absolute best performance compared to other models. Even though RSVD-CF is superior, RSVD-CF requires more attention in taking the number of n-components due to its limitations in providing similar user recommendations to active users. On the other hand, the HDBSCAN-CF model is ill-suited to dealing with sparsity and scalability problems, where its performance is weakest. The HDBSCAN-RSVD-CF model is a fusion of the HDBSCAN and RSVD methods. It works by first grouping users with the same cluster into a single matrix, followed by dimension reduction using RSVD by taking the most dominant part of the matrix. Even though HBDSCAN-RSVD-CF has an inferior RMSE compared to RSVD-CF, this model can still deal with sparsity and scalability problems quite satisfactorily, with a relatively wider

n-component retrieval compared to RSVD-CF. In short, the HDBSCAN-RSVD-CF model is well-suited to solving the sparsity and scalability problem in terms of improving and accelerating RMSE and computing time, especially for book recommendations on E-commerce. The HDBSCAN-RSVD-CF model also has some disadvantages, especially for HDBSCAN, when the chosen hyperparameters of min_cluster_size and min_samples can take many iterations and must be fine-tuned with care. The oversight of the chosen number of these two primary parameters might be included in a more conservative model in which too much data is flagged as noise, and biased grouping potentially leading to a biased recommendation of results. This is also a challenge for any extremely sparse dataset, which might need some preprocessing steps to avoid any degradation of the model's performance. This research also shows that the benefit of using the proposed method can improve the recommendation of books, resulting in more books being sold in industry cases. The problem of sparsity and scalability, which consecutively reflected on sparse rating books and computing time, also strengthens the usability in the common industry cases, with promising results. This can help any book industry to increase its revenue by applying the proposed model to give book recommendations.

6. FURTHER WORKS

In this research paper, the scope of work mainly focused on showing the performance of limited novelty methods for CF called HDBSCAN and RSVD with the CF as the based model and benchmark. Further works of the recommendation system journey, especially the HDBSCAN-RSVD-CF model, this model can be compared to other commonly used industrial recommendation system models, such as deep learning, graph embedding, matrix factorization-based models, and any other limited novelty or sophisticated method. The model can also be applied to other datasets with larger sizes, such as MovieLens 10M, medical field of diagnosing recommendations, music datasets, and general online marketplace recommendations, to overcome sparsity and scalability problems. Furthermore, this proposed model can also be used as a comparison benchmark to overcome more complex problems in recommendation systems, such as cold start problems or classification needs of benchmarking. Overall, the proposed model shows a promising result to overcome sparsity and scalability problems. This method is considered new and needs further study to help further development of recommendation systems.

FUNDING INFORMATION

This research is fully supported by FMIPA Universitas Indonesia Research Grant 2023-2024 No. NKB-019/UN2.F3.D/PPM.00.02/2023.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Muhammad Ichsanudin	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓			
Bevina Desjwiandra Handari	✓	✓		✓	✓	✓		✓		✓		✓		
Bambang Dwi Wijanarko				✓		✓				✓				
Gatot Fatwanto Hertono		✓			✓			✓		✓	✓	✓	✓	✓

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

In this research, the book e-commerce dataset can be accessed from third party open-source data Kaggle for electronic book trading [21]. The data is still maintained by the Kaggle community with

corresponding author and collaborator [BJ & SG], where the source of the data comes from Goodreads Book Datasets and already certified with CC0 license. For the scope of this research, the data collection is limited from 2008 to 2023 (15 years), divided into two parts: the dataset with the details of a book which contains 19 columns (Id, Name, Authors, CountsOfReview, Description, ISBN, Language, PublishDay, PublishMonth, PublishYear, Publisher, Rating, RatingDist1, RatingDist2, RatingDist3, RatingDist4, RatingDist5, RatingDistTotal, and pagesNumber) that contains ±5,000,000 entries, and a second set containing detailed book ratings by users (Id, Name, and Rating) that contains ±1,100 users with user ratings of more than 360,000 books. Datasets that has been created, now march into preprocessing steps, such as dropping unnecessary values (e.g., null data & unknown symbols), harmonizing upper and lower case letters, and adjusting table columns with Id and Name (book name) columns as the keys for joining, results in a user-books matrix with the user ID as the index, columns for the book names, and rating as the contents of the matrix. The matrix has 3,336 users as rows and 7,591 columns. The data with the value NaN is replaced by zero as a neutral rating number.

REFERENCES

[1] U. Kużelewska and K. Wichowski, "A modified clustering algorithm DBSCAN used in a collaborative filtering recommender system for music recommendation," *Advances in Intelligent Systems and Computing*, vol. 365, pp. 245–254, 2015, doi: 10.1007/978-3-319-19216-1_23.

[2] G. Guo, "Improving the performance of recommender systems by alleviating the data sparsity and cold start problems," *Twenty-Third International Joint Conference on Artificial Intelligence*, pp. 3217–3218, 2013.

[3] O. Georgiou and N. Tsapatsoulis, "Improving the scalability of recommender systems by clustering using genetic algorithms," *Artificial Neural Networks – ICANN 2010*, pp. 442–449, 2010, doi: 10.1007/978-3-642-15819-3_60.

[4] H. Zarzour, Z. Al-Sharif, M. Al-Ayyoub, and Y. Jararweh, "A new collaborative filtering recommendation algorithm based on dimensionality reduction and clustering techniques," *2018 9th International Conference on Information and Communication Systems (ICICS)*, pp. 102–106, 2018, doi: 10.1109/IACS.2018.8355449.

[5] M. Nilashi, O. Ibrahim, and K. Bagherifard, "A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques," *Expert Systems with Applications*, vol. 92, pp. 507–520, 2018, doi: 10.1016/j.eswa.2017.09.058.

[6] K. Indira and M. K. K. Devi, "Multi cloud based service recommendation system using DBSCAN algorithm," *Wireless Personal Communications*, vol. 115, no. 2, pp. 1019–1034, 2020, doi: 10.1007/s11277-020-07609-3.

[7] Y. Yang, H. Yao, R. Li, and S. Wang, "A collaborative filtering recommendation algorithm based on user clustering with preference types," *Journal of Physics: Conference Series*, vol. 1848, no. 1, 2021, doi: 10.1088/1742-6596/1848/1/012043.

[8] X. Liao, X. Li, Q. Xu, H. Wu, and Y. Wang, "Improving ant collaborative filtering on sparsity via dimension reduction," *Applied Sciences*, vol. 10, no. 20, pp. 1–10, 2020, doi: 10.3390/app10207245.

[9] Z. Wang, J. Zhou, J. Gan, and J. Wang, "An improved collaborative filtering algorithm based on dimension reduction and improved clustering," *ACM International Conference Proceeding Series*, pp. 11–16, 2021, doi: 10.1145/3456415.3457220.

[10] D. Wang, Y. Yih, and M. Ventresca, "Improving neighbor-based collaborative filtering by using a hybrid similarity measurement," *Expert Systems with Applications*, vol. 160, 2020, doi: 10.1016/j.eswa.2020.113651.

[11] J. Chen, C. Zhao, Uliji, and L. Chen, "Collaborative filtering recommendation algorithm based on user correlation and evolutionary clustering," *Complex and Intelligent Systems*, vol. 6, no. 1, pp. 147–156, 2020, doi: 10.1007/s40747-019-00123-5.

[12] M. Li, L. Wen, and F. Chen, "A novel collaborative filtering recommendation approach based on soft co-clustering," *Physica A: Statistical Mechanics and its Applications*, vol. 561, 2021, doi: 10.1016/j.physa.2020.125140.

[13] S. Natarajan, S. Vairavasundaram, S. Natarajan, and A. H. Gandomi, "Resolving data sparsity and cold start problem in collaborative filtering recommender system using Linked Open Data," *Expert Systems with Applications*, vol. 149, 2020, doi: 10.1016/j.eswa.2020.113248.

[14] A. Szwabe, M. Ciesielczyk, and T. Janasiewicz, "Semantically enhanced collaborative filtering based on RSVD," *Computational Collective Intelligence. Technologies and Applications*, pp. 10–19, 2011, doi: 10.1007/978-3-642-23938-0_2.

[15] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 376–390, 2020, doi: 10.1007/s11036-019-01246-2.

[16] L. F. G. Morales, P. V.- Diaz, R. Reátegui, and L. B.- Guaman, "Drug recommendation system for diabetes using a collaborative filtering and clustering approach: development and performance evaluation," *Journal of Medical Internet Research*, vol. 24, no. 7, 2022, doi: 10.2196/37233.

[17] L. J. Aaldering, J. Leker, and C. H. Song, "Recommending untapped M&A opportunities: a combined approach using principal component analysis and collaborative filtering," *Expert Systems with Applications*, vol. 125, pp. 221–232, 2019, doi: 10.1016/j.eswa.2019.02.004.

[18] H. Liu *et al.*, "EDMF: Efficient deep matrix factorization with review feature learning for industrial recommender system," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 7, pp. 4361–4371, 2022, doi: 10.1109/TII.2021.3128240.

[19] D. Li *et al.*, "CARM: Confidence-aware recommender model via review representation learning and historical rating behavior in the online platforms," *Neurocomputing*, vol. 455, pp. 283–296, 2021, doi: 10.1016/j.neucom.2021.03.122.

[20] H. Liu *et al.*, "Multi-perspective social recommendation method with graph representation learning," *Neurocomputing*, vol. 468, pp. 469–481, 2022, doi: 10.1016/j.neucom.2021.10.050.

[21] B. Jannesar and S. Ghaderi, "Goodreads book datasets with user rating 2M," *Kaggle*. Accessed: Mar. 14, 2023. [Online]. Available: https://www.kaggle.com/datasets/bahramjannesarr/goodreads-book-datasets-10m

[22] R. Logesh, V. Subramaniyaswamy, D. Malathi, N. Sivaramakrishnan, and V. Vijayakumar, "Enhancing recommendation stability of collaborative filtering recommender system through bio-inspired clustering ensemble method," *Neural Computing and Applications*, vol. 32, no. 7, pp. 2141–2164, 2020, doi: 10.1007/s00521-018-3891-5.

[23] F. S. de A. Neto, A. F. da Costa, M. G. Manzato, and R. J. G. B. Campello, "Pre-processing approaches for collaborative filtering based on hierarchical clustering," *Information Sciences*, vol. 534, pp. 172–191, 2020, doi: 10.1016/j.ins.2020.05.021.

[24] L. McInnes, J. Healy, and S. Astels, "HDBSCAN documentation release 0.8.1," *Read the Docs*, 2018. [Online]. Available: https://app.readthedocs.org/projects/hdbscan/downloads/pdf/0.8.16/
[25] pypi, "HDBSCAN," pypi.org. Accessed: Jul. 04, 2023. [Online]. Available: https://pypi.org/project/hdbscan/
[26] D. Moulavi, P. A. Jaskowiak, R. J. G. B. Campello, A. Zimek, and J. Sander, "Density-based clustering validation," *SIAM International Conference on Data Mining 2014, SDM 2014*, vol. 2, pp. 839–847, 2014, doi: 10.1137/1.9781611973440.96.
[27] S. L. Brunton and J. N. Kutz, *Data-driven science and engineering: machine learning, dynamical systems, and control*. Cambridge, United Kingdom: Cambridge University Press, 2019.
[28] M. G. Vozalis and K. G. Margaritis, "Using SVD and demographic data for the enhancement of generalized collaborative filtering," *Information Sciences*, vol. 177, no. 15, pp. 3017–3037, 2007, doi: 10.1016/j.ins.2007.02.036.
[29] K. Bansal and V. Jain, "Performance analysis of collaborative filtering based recommendation system on similarity threshold," *020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 442–448, 2020, doi: 10.1109/ICCMC48092.2020.ICCMC-00083.
[30] G. Geetha, M. Safa, C. Fancy, and D. Saranya, "A hybrid approach using collaborative filtering and content based filtering for recommender system," *Journal of Physics: Conference Series*, 2018, doi: 10.1088/1742-6596/1000/1/012101.

BIOGRAPHIES OF AUTHORS

Muhammad Ichsanudin a Bachelor of Mathematics degree from Universitas Indonesia (2023), is a person that very fond of everything that connected to data, especially data science path. He is currently a consultant at 1 of Google's partners in Indonesia company as data scientist and machine learning engineer. His passion for machine learning and data scientists leads him to contribute to society and the knowledge of data to help the improvement of machine learning to ease the work of humanity. His interest in research includes system recommendation, large language machine, machine learning, deep learning, and data science. He can be contacted at email: muhammad.ichsanudin@sci.ui.ac.id.

Bevina Desjwiandra Handari is an associate professor in Applied Data Science in Health Science in the Department of Mathematics and Natural Sciences, Universitas Indonesia. She finished her Ph.D. in Mathematics from the Department of Mathematics, The University of Queensland, Brisbane, Australia. She completed her M.Sc. in Computational Mathematics from the Department of Mathematics, Michigan State University, East Lansing, USA, and her bachelor in Mathematics from the Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Indonesia. Her research interests are data science and biomathematics. She can be contacted at email: bevina@sci.ui.ac.id.

Bambang Dwi Wijanarko holds a Doctor of Computer Science degree from Universitas Bina Nusantara in 2020. He also received his M.Kom. in Information Technology from STTI Benarif Indonesia in 1995 and a bachelor's degree in Mathematics from Universitas Diponegoro in 1992. He has currently a lecturer at Universitas Bina Nusantara, Jakarta, Indonesia since 2017. His research includes machine learning, natural language processing, and recommendation systems, with a focus on applications in education and sentiment analysis. In the last five years, he has published over 45 papers in international journals and conferences. From 2020 to 2025, he served as Deputy Campus Director at Binus University, Semarang. He can be contacted at email: bwijanarko@binus.edu or bdwjaya@gmail.com.

Gatot Fatwanto Hertono holds a Doctor of Mathematics degree from the University of Queensland, Australia, in 2002. He also received his M.Sc. in Computational Mathematics from Michigan State University, USA, in 1991, and a bachelor's degree in mathematics from Universitas Indonesia in 1986. He has currently an Associate Professor at Department of Mathematics, Universitas Indonesia, Depok, Indonesia, since August 1987. His research includes scientific computing, data science, machine learning, and high-performance computing. In the last ten years, he has published over 10 papers in international journals and conferences. From January 2020 to January 2025, he was in charge as a Director of Academic Development and Learning Resources unit at Universitas Indonesia. He can be contacted at email: gatot-f1@ui.ac.id or gatot-f1@sci.ui.ac.id.