Received: November 1, 2023. Revised: January 16, 2024.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

MF-NCG: Recommendation Algorithm Using Matrix Factorization-based Normalized Cumulative Genre

Triyanna Widiyaningtyas[1]* **Aji Prasetya Wibawa[1]** **Utomo Pujianto[1]**
Wahyu Caesarendra[2]

[1]*Department of Electrical Engineering and Informatics, Universitas Negeri Malang, Indonesia*
[2]*Faculty of Integrated Technologies, Universiti Brunei Darussalam, Brunei Darussalam*
* Corresponding author's Email: triyannaw.ft@um.ac.id

Abstract: Collaborative filtering has emerged as one of the most prevalent techniques for various commercial recommendations. Utilizing a similarity measure to identify similar neighbors is essential to collaborative filtering. Behavior scores and user ratings have recently become increasingly important factors in determining similarity. However, the added users' behavior scores generate a more complex computation. This research proposes a new similarity technique incorporating the matrix factorization and users' behavior score-based similarity to minimize computation time. The matrix factorization technique uses singular value decomposition (SVD), and the users' behavior score-based similarity employs normalized cumulative genre (NCG). Compared to the previous algorithm (i.e., users' scores probability-based collaborative filtering), the experimental findings with the MovieLens 1M and 100k datasets demonstrated a faster computing time. In addition, with these datasets, our similarity reduces the root mean square error (RMSE) by 8.14% and 11.99% and the mean absolute error (MAE) by 13.52% and 15.81%.

Keywords: Similarity measure, Matrix factorization, SVD, NCG.

1. Introduction

Nowadays, e-commerce platforms provide an extensive range of products, allowing users to choose the right one based on their interests. Users eventually succumb to information overload because many things are available on e-commerce networks [1–3]. Matching users with the best products are the key to enhancing user loyalty and satisfaction. As a result, recommender systems have grown in popularity on e-commerce platforms because they assess user interest patterns and provide tailored recommendations based on customers' preferences.

The recommender system is a decision advisor that helps users navigate the web's rapid content expansion. Users are recommended personalized services or products that they are more likely to find appealing [4, 5]. The use of recommender systems is becoming increasingly widespread in modern society. Examples of these industries include news, movies, music, healthcare, books, tourism, article recommendations, and many others [3, 4, 6]. An effective recommender system could significantly enhance customer sales and help the company grow.

Recommender systems can be divided into three categories: content-based filtering, collaborative filtering, and hybrid systems, according to a survey of related literature on various implementations in recommendation systems. The content-based strategy uses distinctive data from the goods that customers buy. Interactions between customers and products are used in the collaborative filtering approach. In comparison, hybrid systems implement recommender systems by combining content-based and collaborative filtering techniques [2, 7–9].

Collaborative filtering is a widely used approach for recommendation system development, out of the three strategies for recommender systems where customers can add new product preferences by sharing their likes [3, 10–12]. The main benefit of the collaborative filtering method is that creating profiles doesn't require a lot of consumer or product data.

There are two more categories for the collaborative filtering strategy: model-based and memory-based [3, 7, 13]. In a model-based method, the model is trained, and predictions are produced using partial ratings. Based on the ratings of similar users or objects, a memory-based method predicts missing ratings. In the memory-based technique, choosing the right similarity measure is crucial since it makes finding related users or products easier. The conventional similarities—cosine, person's correlation, and Jaccard—are often used in recommender systems [14].

In a memory-based method, various similarity measures have been developed to determine how similar things are; however, most of these similarity metrics experience data sparsity issues, which appear when the ratio of ratings that must be estimated to those currently available is unusually high. Numerous memory-based researches have strongly emphasized determining how well each pair of users' interests are similar using various similarity measures to overcome these issues.

The study [15] proposed extended-Jaccard similarity to include the value of rates in the similarity computation. Kim et al. [16] also suggest the ConSimMR similarity measure based on the same concept as the Jaccard similarity. In [17], the similarity approach performs user and item clustering and then calculates similarity by integrating the Triangle and Jaccard similarity measures. The multilevel similarity function was applied in [18] to separate the two users' similarity into many degrees according to the preexisting limitations. The similarity was determined by Jin et al. [19] using a nonlinear function and weighted using the singularity factor. These similarity functions only use the user rating data to determine how similar each pair of users is without exploring the other data that may influence the recommendation results.

Therefore, the studies by [20, 21] propose similarity measures to improve high-quality recommendations for users by using rating and behavior data. The ratings submitted directly by users are represented in the user rating data. In the meantime, user behavior data represents an overall score gathered from users who indirectly accessed genre data. The final similarity was computed by incorporating rating-based and behavior-based similarities. The similarity measures were evaluated in the movie dataset. The findings of this research can enhance recommendations' effectiveness (especially prediction accuracy).

The drawback of the similarity metrics is that the user behavior data from the accumulated genres has not been normalized. This results in the accuracy of rating estimations still being high. In addition, the algorithms become more complex, which results in increased processing time as the amount of data increases [20, 21]. One strategy to deal with the expanding amount of data in recommender systems is matrix factorization. The strategy aims to reflect users and items in a latent space with fewer dimensions. The method of matrix factorization known as singular value decomposition (SVD) is among the most used ones.

Based on the drawbacks from previous research, our work proposes a novel recommender system model that incorporates the matrix factorization method and the similarity measures involving normalized user behavior data to alleviate the computational time and improve the estimated rating accuracy. The matrix factorization approach utilizes the SVD technique, and the similarity function employs normalized cumulative genre (NCG). Our proposed model is called matrix factorization-based NCG (MF-NCG).

The structure of this paper's outline is as follows. The first topic covered in section 2 is related works on similarity measures and matrix factorization. Then, section 3 describes the specifics of our suggested approach. Section 4 explores this by illustrating and discussing the experimental findings before providing a section for conclusions.

2. Related work

Memory-based algorithms address the collaborative filtering issue by utilizing the entire database. The goal of this algorithm is to forecast the ranking of active users by identifying people who are similar to the consumers they want to anticipate or who share their interests. In order to calculate similarity, the correlation between two users must be determined. The similarity value between two users ranges from -1 to 1. Two users might have a similarity score of either -1 or 1. When two people rate something exactly the same, they have a value of 1, and when they rate something precisely the opposite, they have a value of -1. In recommender systems, the person's correlation coefficient (PCC) and cosine similarity are the standard similarity measures that are usually employed.

Some research suggests using similarity functions to enhance the effectiveness of recommendation systems. Ayub et al. [15] used the Jaccard idea by considering the proportion of all standard ratings to all ratings where the absolute values of two users or products are identical. Their proposed similarity is called the extended-Jaccard similarity. Their research produced an average MAE value of 0.952 in testing with the MovieLens 100k dataset. Kim et al. [16] also utilized the same concept as Jaccard similarity by employing a parallel algorithm that comprises MapReduce phases: products' partition, and intra-and inter-similarity calculation. Their suggested similarity is introduced as the ConSimMR similarity. Hereafter, Yan et al. [17] introduced a novel similarity algorithm by utilizing enhanced Jaccard similarity and the Gaussian mixture model. The algorithm clusters the user and product by using the Gaussian model and then calculates the similarity by integrating Jaccard and triangular similarity. Using the MovieLens 100k dataset, their research yielded an average MAE value of 0.736. Next, Polatidis et al. [18] proposed the multilevel similarity measure. This similarity adopts the PCC by dividing the user's similarity into many levels. They conducted four-level trials to demonstrate the efficacy of their approach in this regard. Upon evaluating their findings with the MovieLens 1M dataset, they obtained an average MAE value of 0.8. Jin et al. [19] presented a new similarity using a nonlinear function and the singularity factor for weighting. Their investigation yielded an average MAE value of 0.77 when tested using the MovieLens 100k dataset. The drawback of similarity measures was proposed in these studies [15–19], only considering the users' ratings in their similarity calculations, disregarding other variables that might affect the recommendation's performance (i.e., the prediction error is still high, ranging from 0.736 to 0.952).

Furthermore, some researchers [20–22] have proposed similarity measures by integrating the users' ratings-based and behaviors-based similarities. The ratings score directly assigned to the products are what is included in the users' ratings. The users' behavior information, on the other hand, is a cumulative score that was attained from indirectly obtaining genre information. The users' ratings-based similarity applies cosine similarity, while the behaviors-based similarity employs the PCC. After assigning weight to each similarity, the ultimate similarity is then calculated using a combination of these two similarities. There are several methods for allocating similarity weights to these similarities [20–22]. The threshold value determines the weighting for the similarity in [20]. In contrast, the similarities in [21] and [22] utilize correlation coefficients.

These similarity metrics, when applied to the MovieLens dataset, can produce a reasonably accurate rating prediction. Put differently, these similarities can enhance suggestion efficacy compared to similarity algorithms that rely on users' rating information. It occurs due to the two similarities, which employ user behavior data to determine user similarity.

Although the similarities in [20, 21] are better than the conventional similarities that solely include user rating data, both similarities still have some shortcomings; the similarity measures from the accumulated genre data have not been normalized. This results in the rating estimation's accuracy still needing to be improved [20, 21]. Therefore, this study proposes the normalized cumulative genre (NCG) similarity to enhance the prediction accuracy. Moreover, the algorithms behind the two similarities are more intricate, so they take more time to process. These two similarities have a time complexity of $O(m^2n + m^2g)$, whereas the classic similarity has a time complexity of $O(m^2n)$. m, n, and g illustrate the number of users, items, and types/genres of items, respectively. The method performance will drop if the number of users rises tenfold with a fixed number of items and product genres. This is because the computation time for users' behavior-based similarities will increase by 100. It results in issues with the recommender system's scalability. Several recommender system researchers use matrix factorization techniques to solve the scalability issues.

Matrix factorization is one family of well-liked collaborative filtering approaches [23–25]. The goal of this method is to depict a rating matrix R (of size $m \, x \, n$) that is produced from two significantly smaller matrices ($R = U.I$). U denotes the user's latent factor vector with a matrix dimension of $m \, x \, k$, and I represents the item's latent factor vector with a matrix dimension of $k \, x \, n$. Fig. 1 illustrates the matrix factorization of the rating matrix (R), resulting in two matrices (matrix U and matrix I), where the dot product of two matrices produces matrix R' that an approximation of the initial rating matrix (R). Various cell shadings correspond to varying user preference levels.

One of the matrix factorization approaches is SVD. The idea of SVD is to divide the rating matrix R into a unique item of three matrices. The goal is to minimize the RMSE to express the latent component in the rating matrix R. Fig. 2 illustrates the decomposition matrix utilizing SVD.

The SVD process decomposes the matrix rating R (with M users and N items) into three matrices (U, Σ, and V^T). U denotes an orthonormal column with dimension M x r, which explains how users and

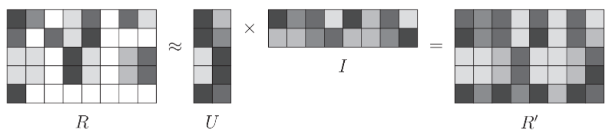

Figure. 1 Illustration of matrix factorization [23]

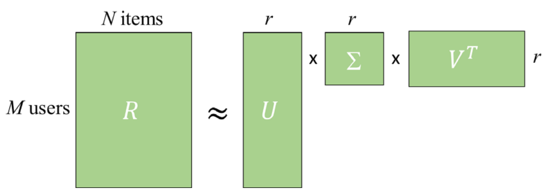

Figure. 2 Illustration of SVD [26]

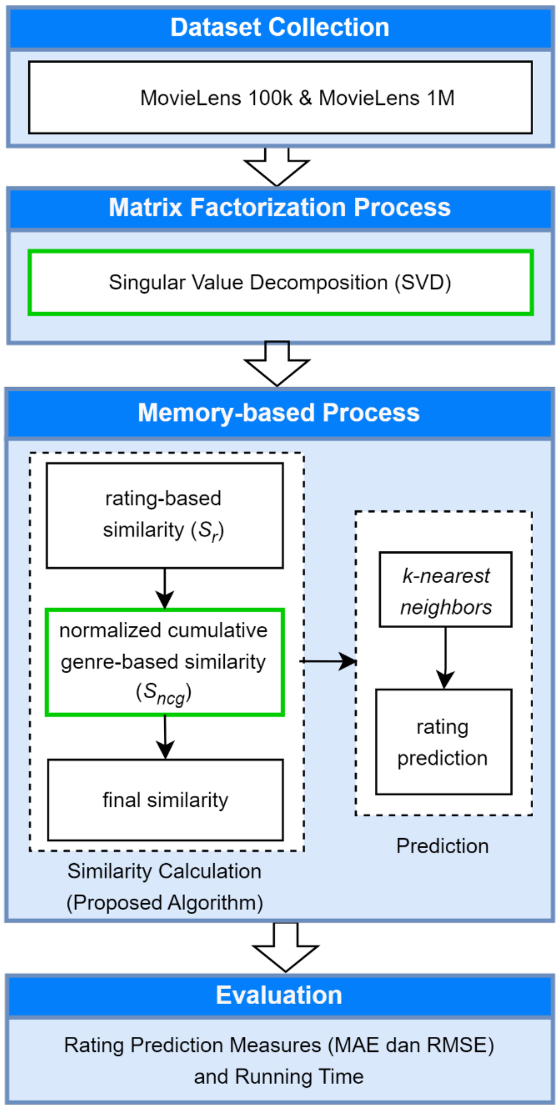
Dataset Collection
MovieLens 100k & MovieLens 1M
Matrix Factorization Process
Singular Value Decomposition (SVD)
Memory-based Process
rating-based similarity (S_r)
normalized cumulative genre-based similarity (S_{ncg})
final similarity
Similarity Calculation (Proposed Algorithm)
k-nearest neighbors
rating prediction
Prediction
Evaluation
Rating Prediction Measures (MAE dan RMSE) and Running Time

Figure. 3 Research stages

latent variables interact. Next, Σ states a diagonal matrix with dimension r x r, which indicates the strength of each latent factor. V^T represents orthonormal rows with dimension N x r, which means the similarity between items and latent factors. Finally, r is the rank of the rating matrix R. This study utilizes matrix U with a lower dimension compared to matrix R, which aims to reduce the processing time to generate user similarity.

3. Research method

This work proposes an approach for making recommendations called MF-NCG, which combines the matrix factorization method and the users' behavior-based similarity function (i.e., NCG). The matrix factorization approach uses SVD to obtain the user-latent matrix for calculating user similarity. Fig. 3 shows the four stages of our study: dataset collection, matrix factorization process, memory-based process, and evaluation.

3.1 Dataset collection

Dataset collection is the initial stage in this work. We evaluated our proposed method on two publicly available datasets (MovieLens 100k and MovieLens 1M). MovieLens 100k is a public ratings dataset including 100,000 ratings that 943 users evaluate on 1,682 movies. With 6,040 users and 3,952 movies, the MovieLens 1M dataset has 1,000,209 ratings. Every user in these two datasets has appraised at least 20 movies with a rating score on the scale of [1, 5]. These two datasets have 93.7% and 95.75% data sparsity, respectively [27]. The sparsity of these datasets is a recommender system's challenge to get an accurate user interest. Hence, we applied matrix factorization with the SVD technique to estimate the unrated products.

3.2 Matrix factorization process

The second stage is the matrix factorization process. We use the widely used SVD algorithm, a matrix decomposition technique that creates smaller matrices from larger ones. The unrated rating in this study is predicted by SVD. A transposed item-factor matrix (Q^T_{nk}) and a user-factor matrix (P_{mk}) are obtained from the rating matrix (R_{mn}). Users, items, and factors are denoted by m, n, and k. Factors describe the qualities that people or things have.

Eq. (1) defines the formula used in the estimated rating computation [28–33].

$$\hat{r}_{u,i} = \mu + b_u + b_i + p_u \cdot q_i^T \quad (1)$$

$\hat{r}_{u,i}$ presents the estimated rating of item i by user u. μdescibes the average rating of all items. b_u and b_i are biased scores for reducing the estimation errors of user and item average ratings. p_u states the user–factor vector, where $p_u \in P_{mk}$ with u=1, 2, 3, ...,m. While q_i^T presents a transposed item–factor vector. $q_i \in Q_{nk}$ with i=1, 2, 3, …, n.

3.3 Memory-based process

The memory-based process is the third stage of this study. The two steps in this stage are similarity computation and rating estimation. In this work, similarity computation employs NCG, which refers to Eq. (2). The rating prediction method utilizes k nearest neighbor (kNN).

$$Sim(u_1,u_2)^{NCG} = \beta . S_r(u_1,u_2) + (1-\beta). S_n(u_2,u_2) \quad (2)$$

$S_r(u_1,u_2)$ represents the users' ratings-based similarity between two users ($user_1$ and $user_2$) that employed the cosine similarity, which is formulated in Eq. (3). $S_n(u_1,u_2)$ indicates the normalized users' behavior-based similarity between two users ($user_1$ and $user_2$) that utilized the PCC similarity (by modifying the users' ratings with the normalized users' behavior), which is formulated in Eq. (4). The threshold value, denoted as β, is between 0 and 1.

$$Sim(u_1,u_2)^{COS} = \frac{\vec{r}_{u_1}\cdot\vec{r}_{u_2}}{\|\vec{r}_{u_1}\|\cdot\|\vec{r}_{u_2}\|} = \frac{\sum_{i\in I_{u_1}\cap I_{u_2}} r_{u_1 i}\cdot r_{u_2 i}}{\sqrt{\sum_{i\in I_{u_1}\cap I_{u_2}} r_{u_1 i}^2}\cdot\sqrt{\sum_{i\in I_{u_1}\cap P_{u_2}} r_{u_2 y}^2}} \quad (3)$$

$$S_n(u_1,u_2) = \frac{\sum_{g\in G_{u_1}\cap G_{u_2}}(N_{u_1 g}-\bar{N}_{u_1})(N_{u_2 g}-\bar{N}_{u_2})}{\sqrt{\sum_{g\in G_{u_1}\cap G_{u_2}}(N_{u_1 g}-\bar{N}_{u_1})^2}\cdot\sqrt{\sum_{g\in G_{u_1}\cap G_{u_2}}(N_{u_2 g}-\bar{N}_{u_2})^2}} \quad (4)$$

$r_{u_1 i}$ and $r_{u_2 i}$ describe the rating score for item i from two users (u_1 and u_2), respectively. I_{u_1} and I_{u_2} denote the collection of items evaluated by $user_1$ (u_1) and $user_2$ (u_2), respectively. i is one of the items evaluated by both users. $\bar{r}_{u_1}$ and $\bar{r}_{u_2}$ show the average rating of all items evaluated by $user_x$ (u_x) and $user_y$ (u_y), respectively. $N_{u_1 g}$ and $N_{u_2 g}$ present the normalized genre score to item type/genre g from $user_1$ (u_1) and $user_2$ (u_2), respectively. $\bar{N}_{u_1}$ and $\bar{N}_{u_2}$ represent the average normalized genre scores of all item genres evaluated by $user_1$ (u_1) and $user_2$ (u_2), respectively. G_{u_1} and G_{u_2} indicate the set of item types evaluated by $user_1$ (u_1) and $user_2$ (u_2), respectively. g is one of the item genres evaluated by two users.

3.4 Evaluation

Evaluation is the last phase of this research method. This phase assesses the effectiveness of the suggested algorithm compared to the previous algorithms. The performance evaluation in this work uses measures for running times and predictions. Metrics for predictions use RMSE and mean absolute error (MAE), shown in Eqs. (5) and (6) [34, 35].

$$RMSE = \sqrt{\frac{1}{N}\sum_{u\epsilon U,i\epsilon I}(r_{ui}-\hat{r}_{ui})^2} \quad (5)$$

$$MAE = \frac{1}{N}\sum_{u\epsilon U,i\epsilon I}|r_{ui}-\hat{r}_{ui}| \quad (6)$$

4. Experiment

In this section, we first present an overview of the experimental setup. The results of comparing the suggested method with previous algorithms are then discussed. We conclude by talking about the results of our experiment.

4.1 Experimental setting

We used Python programming to test the efficacy of the suggested MF-NCG approach using two MovieLens datasets: MovieLens 100k and MovieLens 1M. We used an experimental setting following previous research [21]. Each dataset is split using the k-fold cross-validation (CV) approach into testing and training data. Testing data is used to evaluate the efficacy of the recommendation system, while training data is utilized to construct the similarity model. To split 20% of the testing data and 80% of the training data, we set the value of k to k = 5. Additionally, we configured the number of closest neighbors (k) to vary between 10 and 100.

We compared our proposed MF-NCG algorithm with three previous similarity algorithms, namely cosine similarity, users' probability scores-based collaborative filterings (UPCF) [20], and users' profiles correlation-based similarity (UPCSim) [21].

Note that in this study the experimental data using the MovieLens 100k dataset on previous algorithms (Cosine, UPCF, and UPCSim) was taken from research [21]. Meanwhile, to test the MovieLens 1M dataset, we conducted all experiments on the four algorithms (Cosine, UPCF, UPCSim, and the proposed MF-NCG).

4.2 Experimental results

This section evaluates the performance of our suggested MF-NCG method against the previous

Table 1. The comparison of MAE scores of the four algorithms using the MovieLens 100k dataset

k	MAE			
	Cosine	UPCF	UPCSim	MF-NCG
10	0.8227	0.7792	0.7669	0.6814
20	0.8099	0.7631	0.7483	0.6725
30	0.8051	0.7565	0.7410	0.6691
40	0.8048	0.7551	0.7387	0.6679
50	0.8043	0.7544	0.7369	0.6657
60	0.8041	0.7535	0.7364	0.6649
70	0.8049	0.7529	0.7359	0.6639
80	0.8051	0.7526	0.7355	0.6628
90	0.8056	0.7525	0.7347	0.6616
100	0.8072	0.7521	0.7337	0.6602
Avg	0.8074	0.7572	0.7408	0.6670

Table 2. The comparison of RMSE scores of the four algorithms using the MovieLens 100k dataset

k	RMSE			
	Cosine	UPCF	UPCSim	MF-NCG
10	1.0417	0.9835	0.9793	0.9674
20	1.0248	0.9643	0.9541	0.9500
30	1.0189	0.9574	0.9453	0.9166
40	1.0163	0.9558	0.9427	0.9116
50	1.0162	0.9556	0.9393	0.9107
60	1.0154	0.9547	0.9389	0.9096
70	1.0162	0.9544	0.9383	0.9074
80	1.0170	0.9543	0.9381	0.9069
90	1.0173	0.9542	0.9364	0.9058
100	1.0176	0.9538	0.9359	0.9033
Avg	1.0201	0.9588	0.9448	0.9169

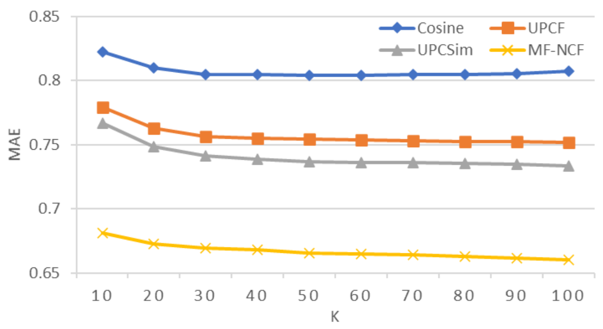

Figure. 4 Comparison of MAE scores using MovieLens 100k dataset

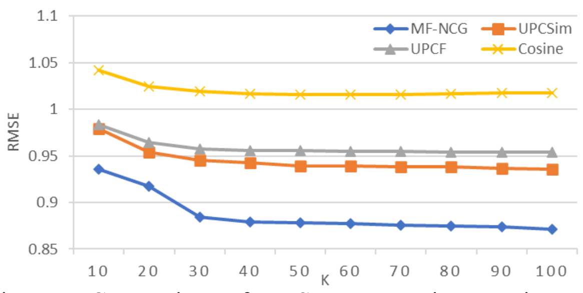

Figure. 5 Comparison of RMSE scores using MovieLens 100k dataset

similarity techniques using training and testing data split (80%:20%. Running-time, RMSE, and MAE are three metrics used to assess these performances.

Table 1 shows the comparison of MAE scores of four algorithms using MovieLens 100k. Based on Table 1, our proposed algorithm declines in the mean MAE values compared to UPCSim is 11.06 %, UPCF is 13.52 %, and Cosine is 21.04 %. It indicates that, on average, prediction accuracy increased while utilizing the matrix factorization process instead of not using it. Fig. 4 illustrates how the four methods' MAE scores fall as the number of closest neighbors increases using the MovieLens 100k.

When the number of closest neighbors rises at the start of the curve, the MAE value decreases extremely sharply; however, when the number of most immediate neighbors grows toward the curve’s terminus, the MAE value tends to remain steady. The MAE score is probably influenced by the number of closest neighbors, with an increase in most immediate neighbors leading to a decrease in the MAE value. With a fixed number of closest neighbors, the MF-NCG method’s MAE score is continuously lower than other methods. In other words, the suggested MF-NCG has the slightest discrepancy between its actual and anticipated scores.

Table 2 shows the comparison of RMSE scores of four algorithms using MovieLens 100k. Based on Table 2, the decrease in the average RMSE values of our proposed algorithm compared to UPCSim is 6.57%, UPCF is 8.14%, and Cosine is 15.06%. It demonstrates that MF-NCG's accuracy is the lowest, suggesting that the recommended approach is better and becomes MF-NCG's benefit.

Fig. 5 illustrates the impact of altering the number of nearest neighbors on the RMSE score. It shows the RMSE score decreases initially for all four techniques and becomes stable when the neighbors surpass 40. The MF-NCG algorithm's RMSE value consistently offers the lowest value for every variation in the number of neighbors. It demonstrates that MF-NCG outperforms the other three algorithms and is superior. It occurs due to our proposed algorithm's usage of the SVD matrix factorization method, which predicts all unrated ratings using the user latent component.

Next, we present the experimental results of MAE and RMSE measurements on the MovieLens 1M dataset. Tables 3 and 4 show the comparison of MAE and RMSE scores of four algorithms using

Table 3. The comparison of MAE scores of four algorithms using the MovieLens 1M dataset

k	**MAE**			
	Cosine	UPCF	UPCSim	MF-NCG
10	0.8048	0.7570	0.7217	0.6327
20	0.7857	0.7356	0.7045	0.6298
30	0.7839	0.7276	0.6996	0.6281
40	0.7836	0.7232	0.6977	0.6269
50	0.7831	0.7207	0.6963	0.6263
60	0.7822	0.7193	0.6957	0.6254
70	0.7837	0.7180	0.6949	0.6245
80	0.7854	0.7170	0.6947	0.6234
90	0.7871	0.7165	0.6939	0.6227
100	0.7890	0.7160	0.6937	0.6211
Avg	0.7869	0.7251	0.6993	0.6261

Table 4. The comparison of RMSE scores of four algorithms using MovieLens 1M dataset

k	**RMSE**			
	Cosine	UPCF	UPCSim	MF-NCG
10	0.8442	0.9241	0.9598	1.0345
20	0.8318	0.8998	0.9330	1.0081
30	0.8288	0.8921	0.9233	1.0010
40	0.8247	0.8888	0.9182	0.9997
50	0.8182	0.8873	0.9153	0.9995
60	0.8178	0.8861	0.9138	0.9991
70	0.8171	0.8859	0.9127	1.0007
80	0.8146	0.8856	0.9116	1.0018
90	0.8137	0.8855	0.9111	1.0030
100	0.8125	0.8853	0.9106	1.0044
Avg	0.8223	0.8921	0.9209	1.0052

MovieLens 1M. Based on Table 3, the decline in the average MAE values of our proposed algorithm compared to UPCSim is 11.69%, UPCF is 15.81%, and Cosine is 25.68%. Meanwhile, the decrease in the average RMSE values of our proposed algorithm compared to UPCSim is 8.48%, UPCF is 11.99%, and Cosine is 22.23%, according to the experiment results in Table 4. These results show that the MAE and RMSE values of the proposed algorithm in testing with the MovieLens 1M dataset also produce the smallest error values. So, it can be said that the proposed algorithm represents its robustness after being tested on different datasets.

Apart from evaluating the RMSE and MAE values as recommendation measures, this study also assessed the execution times of four different algorithms to ascertain the impact of matrix factorization on algorithm execution times. A comparison of running times on the MovieLens 100k with an 80%:20% training and testing data distribution is presented in Fig. 6, and the comparison of running times on the MovieLens 1M is represented in Fig. 7.

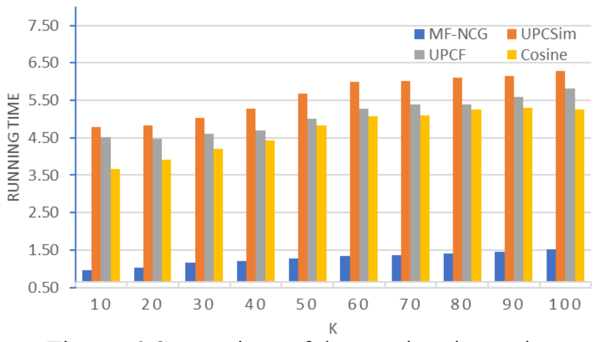

Figure. 6 Comparison of the running time using MovieLens 100k dataset

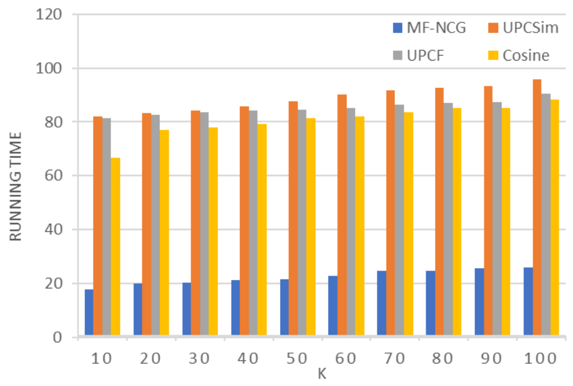

Figure. 7 Comparison of the running time using MovieLens 1M dataset

Compared to the other three algorithms (i.e., UPCSim, UPCF, and Cosine), the running time following the matrix factorization process (MF-NCG) is quicker. The average running time in the MovieLens 100k dataset is 3.425 seconds for Cosine, 3.797 seconds for UPCF, and 4.336 seconds for UPCSim. In contrast, the MovieLens 1M dataset's average running time is 66.219 seconds, 62.840 seconds for UPCF, and 58.241 seconds for Cosine. It indicates that the matrix factorization technique utilizing SVD assists in accelerating the running time to create recommendations; it demonstrates that the performance of the running time following the matrix factorization procedure is superior to that of the other three techniques. It happens due to MF-NCG processing less data than the three similarity algorithms. In addition, comparing the execution time values in the two datasets, the MovieLens 1M dataset has a more excellent execution time value than the MovieLens 100k.

4.3 Discussion

We provide a recommendation system in this paper that combines matrix factorization with memory-based techniques. The memory-based approach uses the normalized cumulative genre to take user behavior and rating scores into account (i.e., NCG). In contrast, SVD is used in the matrix factorization approach. We tested the system on two widely-used datasets: MovieLens 1M and MovieLens 100k. The values of density and sparsity for these two datasets are 6.3% and 4.25%, respectively, and 93.7% and 95.75%, respectively. The k-fold CV technique is applied at k=5 to distribute testing and training data on the dataset. Running time and predicted accuracy (RMSE and MAE) are used to assess the algorithms' results.

Compared to state-of-the-art algorithms (i.e., UPCSim, UPCF, and Cosine), the experiment results demonstrated that combining memory-based and matrix factorization approaches might enhance the prediction performance by lowering MAE and RMSE. Furthermore, following matrix factorization, the recommendation processing time performs better than the other three approaches. It happens due to singular value decomposition caused by the similarity computation, which only takes user latent data into account.

Although the suggested technique may enhance rating estimation and running time performance, the recommended method has drawbacks since it requires serial similarity computing.

5. Conclusion

Applying the MF-NCG algorithm results in a more excellent prediction accuracy value and faster running time than the prior algorithms, based on the findings and explanation in the preceding section. The mean MAE values of our suggested method using MovieLens 100k are 11.06%, 13.52%, and 21.04% lower than those of UPCSim, UPCF, and Cosine, respectively. Furthermore, our suggested algorithm's average RMSE values are lower than those of UPCSim, UPCF, and Cosine by 6.57%, 8.14%, and 15.06%, respectively. Meanwhile, the average running times for Cosine, UPCF, and UPCSim are 3.425 seconds, 3.797 seconds, and 4.336 seconds, respectively. Nevertheless, as the MF-NCG operates serially, more techniques can still be improved.

To enhance recommendation performance, system developers might investigate alternative hybrid approaches and parallel processing to determine user similarities.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, TW and APW; methodology, TW,UP, and WCR; software, UP; validation, APW, UP, and WCR; formal analysis, TW and UP; investigation, TW and WCR; resources, UP; data curation, UP; writing—original draft preparation, TW, APW, UP, and TBA; writing—review and editing, TW, APW, UP, and TBA; visualization, TW; supervision, APW and WCR; project administration, UP; funding acquisition, TW.

Acknowledgments

This research was supported by Universitas Negeri Malang, Research Grant: "Penelitian Internal UM" with contract number 5.4.883/UN32.20.1/LT/2023.

References

[1] G. Behera and N. Nain, "Collaborative Filtering with Temporal Features for Movie Recommendation System", *Procedia Comput. Sci.*, Vol. 218, pp. 1366–1373, 2022, doi: 10.1016/j.procs.2023.01.115.

[2] C. Udokwu, R. Zimmermann, F. Darbanian, T. Obinwanne, and P. Brandtner, "Design and Implementation of a Product Recommendation System with Association and Clustering Algorithms", *Procedia Comput. Sci.*, Vol. 219, No. 2022, pp. 512–520, 2023, doi: 10.1016/j.procs.2023.01.319.

[3] G. Jain, T. Mahara, and S. C. Sharma, "Performance Evaluation of Time-based Recommendation System in Collaborative Filtering Technique", *Procedia Comput. Sci.*, Vol. 218, No. 2022, pp. 1834–1844, 2023, doi: 10.1016/j.procs.2023.01.161.

[4] Z. Z. Darban and M. H. Valipour, "GHRS: Graph-based hybrid recommendation system with application to movie recommendation", *Expert Syst. Appl.*, Vol. 200, No. November 2020, p. 116850, 2022, doi: 10.1016/j.eswa.2022.116850.

[5] P. Mondal, P. Kapoor, S. Singh, S. Saha, J. P. Singh, and N. Onoe, "Task-Specific and Graph Convolutional Network based Multi-modal Movie Recommendation System in Indian Setting", *Procedia Comput. Sci.*, Vol. 222, pp. 591–600, 2023, doi: 10.1016/j.procs.2023.08.197.

[6] L. N. H. Nam, "Towards comprehensive approaches for the rating prediction phase in memory-based collaborative filtering recommender systems", *Inf. Sci. (Ny).*, Vol. 589,

No. 227, pp. 878–910, 2022, doi: 10.1016/j.ins.2021.12.123.

[7] H. Khojamli and J. Razmara, "Survey of similarity functions on neighborhood-based collaborative filtering", *Expert Syst. Appl.*, Vol. 185, No. June, p. 115482, 2021, doi: 10.1016/j.eswa.2021.115482.

[8] R. Duan, C. Jiang, and H. K. Jain, "Combining review-based collaborative filtering and matrix factorization: A solution to rating's sparsity problem", *Decis. Support Syst.*, No. December 2021, p. 113748, 2022, doi: 10.1016/j.dss.2022.113748.

[9] N. Bhalse and R. Thakur, "Algorithm for movie recommendation system using collaborative filtering", *Mater. Today Proc.*, No. xxxx, pp. 1–6, 2021, doi: 10.1016/j.matpr.2021.01.235.

[10] Y. Afoudi, M. Lazaar, and M. A. Achhab, "Hybrid recommendation system combined content-based filtering and collaborative prediction using artificial neural network", *Simul. Model. Pract. Theory*, Vol. 113, No. June, p. 102375, 2021, doi: 10.1016/j.simpat.2021.102375.

[11] Z. Z. Darban and M. H. Valipour, "GHRS: Graph-based hybrid recommendation system with application to movie recommendation", *Expert Syst. Appl.*, Vol. 200, No. November 2020, 2022, doi: 10.1016/j.eswa.2022.116850.

[12] F. Horasan, A. H. Yurttakal, and S. Gündüz, "A novel model based collaborative filtering recommender system via truncated ULV decomposition", *J. King Saud Univ. - Comput. Inf. Sci.*, Vol. 35, No. 8, 2023, doi: 10.1016/j.jksuci.2023.101724.

[13] A. Fareed, S. Hassan, S. B. Belhaouari, and Z. Halim, "A collaborative filtering recommendation framework utilizing social networks", *Mach. Learn. with Appl.*, Vol. 14, No. September, p. 100495, 2023, doi: 10.1016/j.mlwa.2023.100495.

[14] N. Sun, Q. Luo, L. Ran, and P. Jia, "Similarity matrix enhanced collaborative filtering for e-government recommendation", *Data Knowl. Eng.*, Vol. 145, No. October 2021, p. 102179, 2023, doi: 10.1016/j.datak.2023.102179.

[15] M. Ayub, M. A. Ghazanfar, M. Maqsood, and A. Saleem, "A Jaccard base similarity measure to improve performance of CF based recommender systems", *Int. Conf. Inf. Netw.*, Vol. 2018-Janua, pp. 1–6, 2018, doi: 10.1109/ICOIN.2018.8343073.

[16] S. Kim, H. Kim, and J. K. Min, "An efficient parallel similarity matrix construction on MapReduce for collaborative filtering", *J. Supercomput.*, Vol. 75, No. 1, pp. 123–141, 2019, doi: 10.1007/s11227-018-2271-3.

[17] H. Yan and Y. Tang, "Collaborative Filtering Based on Gaussian Mixture Model and Improved Jaccard Similarity", *IEEE Access*, Vol. 7, pp. 118690–118701, 2019, doi: 10.1109/ACCESS.2019.2936630.

[18] N. Polatidis and C. K. Georgiadis, "A multi-level collaborative filtering method that improves recommendations", *Expert Syst. Appl.*, Vol. 48, pp. 100–110, 2016, doi: 10.1016/j.eswa.2015.11.023.

[19] Q. Jin, Y. Zhang, W. Cai, and Y. Zhang, "A new similarity computing model of collaborative filtering", *IEEE Access*, Vol. 8, pp. 17594–17604, 2020, doi: 10.1109/ACCESS.2020.2965595.

[20] C. Wu, J. Wu, C. Luo, O. Wu, C. Liu, Y. Wu, and F. Yang, "Recommendation algorithm based on user score probability and project type", *Eurasip J. Wirel. Commun. Netw.*, Vol. 2019, No. 1, 2019, doi: 10.1186/s13638-019-1385-5.

[21] T. Widiyaningtyas, I. Hidayah, and T. B. Adji, "User profile correlation-based similarity (UPCSim) algorithm in movie recommendation system", *J. Big Data*, Vol. 8, No. 1, 2021, doi: 10.1186/s40537-021-00425-x.

[22] T. Widiyaningtyas, I. Hidayah, and T. B. Adji, "Recommendation algorithm using clustering-based upcsim (Cb-upcsim)", *Computers*, Vol. 10, No. 10, pp. 1–17, 2021, doi: 10.3390/computers10100123.

[23] U. Ocepek, J. Rugelj, and Z. Bosnić, "Improving matrix factorization recommendations for examples in cold start", *Expert Syst. Appl.*, Vol. 42, pp. 6784–6794, 2015, doi: 10.1016/j.eswa.2015.04.071.

[24] S. Chen and Y. Peng, "Matrix factorization for recommendation with explicit and implicit feedback", *Knowledge-Based Syst.*, Vol. 158, pp. 109–117, 2018, doi: 10.1016/j.knosys.2018.05.040.

[25] K. Ji, R. Sun, X. Li, and W. Shu, "Improving matrix approximation for recommendation via a clustering-based reconstructive method", *Neurocomputing*, Vol. 173, pp. 912–920, 2016, doi: 10.1016/j.neucom.2015.08.046.

[26] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, *Recommender Systems Handbook*. New York: Springer, 2015, doi: 10.1007/978-1-4899-7637-6.

[27] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context", *ACM Trans. Interact. Intell. Syst.*, Vol. 5, No. 4, 2015,

doi: 10.1145/2827872.

[28] X. Guan, C. T. Li, and Y. Guan, "Enhanced SVD for collaborative filtering", in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2016, Vol. 9652 LNAI, pp. 503–514. doi: 10.1007/978-3-319-31750-2_40.

[29] X. Guan, C. T. Li, and Y. Guan, "Matrix Factorization with Rating Completion: An Enhanced SVD Model for Collaborative Filtering Recommender Systems", *IEEE Access*, Vol. 5, pp. 27668–27678, 2017, doi: 10.1109/ACCESS.2017.2772226.

[30] M. Pan, Y. Yang, and Z. Mi, "Research on An Extended SVD Recommendation Algorithm Based on User ' s Neighbor Model", *2016 7th IEEE Int. Conf. Softw. Eng. Serv. Sci.*, pp. 81–84, 2016.

[31] Z. Xian, Q. Li, G. Li, and L. Li, "New Collaborative Filtering Algorithms Based on SVD++ and Differential Privacy", *Math. Probl. Eng.*, Vol. 33, pp. 2133–2144, 2017, doi: 10.3233/JIFS-162053.

[32] L. Cui, W. Huang, Q. Yan, F. R. Yu, Z. Wen, and N. Lu, "A novel context-aware recommendation algorithm with two-level SVD in social networks", *Futur. Gener. Comput. Syst.*, Vol. 86, pp. 1459–1470, 2018, doi: 10.1016/j.future.2017.07.017.

[33] S. Sali, "Movie rating prediction using singular value decomposition", *Mach. Learn. Proj. Rep. by Univ. California, St. Cruz*, 2008, [Online]. Available: http://classes.soe.ucsc.edu/cmps242/Winter08/proj/serdar_report.pdf

[34] M. Jalili, S. Ahmadian, M. Izadi, P. Moradi, and M. Salehi, "Evaluating Collaborative Filtering Recommender Algorithms: A Survey", *IEEE Access*, Vol. 6, pp. 74003–74024, 2018, doi: 10.1109/ACCESS.2018.2883742.

[35] X. Li and D. Li, "An Improved Collaborative Filtering Recommendation Algorithm and Recommendation Strategy", *Mob. Inf. Syst.*, Vol. 2019, 2019, doi: 10.1155/2019/3560968.