Publications

Available Online

J. Sci. Res. **16** (3), 705-712 (2024)

JOURNAL OF SCIENTIFIC RESEARCH

www.banglajol.info/index.php/JSR

Exploring Matrix Decomposition Methods for Recommender Systems

A. Sankari[1*], S. Masih[2], M. Ingle[2]

[1]Department of Computer Science & Engineering, Shriram Group of Institutions, Jabalpur, M.P, India

[2]School of Computer Science & Information Technology DAVV, Indore, India

Received 5 January 2024, accepted in final revised form 27 June 2024

Abstract

Matrix factorization, a pivotal technique in recommender systems, is used to uncover latent patterns within user-item interaction matrices. This technique, which reduces the user-product rating matrix into separate lower dimensional matrices, is instrumental in generating personalized recommendations. Our study explores three matrix factorization techniques: U.V. decomposition, singular value decomposition, and the CUR algorithm. We perform a rigorous experimental evaluation on two real-world datasets, ensuring the thoroughness and reliability of our research. We perform a rigorous experimental evaluation to compare the three techniques on two datasets. This provides the thoroughness and reliability of the experimentation.

Keywords: Matrix factorization; Personalized recommendation; Recommender systems.

doi: https://dx.doi.org/10.3329/jsr.v16i3.70831 J. Sci. Res. **16** (3), 705-712 (2024)

1. Introduction

Recommendation systems are mainly based on two primary strategies. They are collaborative filtering methods and content-based filtering methods. While the content-based filtering technique creates a profile for the person or product to explain its attributes, the collaborative filtering method depends on the user's past ratings. Two primary types of collaborative filtering are latent factor models and neighborhood methods. Latent factor models aim to describe users and items to interpret the ratings. The neighborhood methods are centered on computing the relationships between users or items. Numerous successful implementations of latent factor models are grounded in matrix factorization techniques. Matrix factorization techniques describe users and items using vectors of factors derived from patterns in the ratings. [1].

Matrix factorization confronts numerous challenges within recommender systems, including the sparsity of the user-item matrix, the cold start predicament for newly introduced users and items, and the scalability issues of the algorithms. Matrix factorization techniques overcome these challenges by leveraging the implicit feedback from the users,

* *Corresponding author*: abhilasha.sankari@gmail.com

incorporating information about the items or users, and scaling to large datasets using distributed computing frameworks.

Several matrix factorization methods are used in recommender systems, such as Singular Value Decomposition (SVD), Matrix Factorization with neural networks [2], U.V. Decomposition. and Probabilistic Matrix Factorization (PMF). These methods differ in their assumptions about the distribution of the data, the objective function, and the regularization techniques used to avoid overfitting. Matrix factorization is a crucial component of modern recommender systems. It enables accurate and personalized user recommendations in various domains, such as e-commerce, social networks, and online advertising.

2. Methodology

In this section, we present three popular techniques, namely singular value decomposition, U.V. decomposition, and CUR algorithm for matrix factorization, that are extensively used in recommender systems. Singular Value Decomposition [3] is utilized to perform two main tasks. Firstly, SVD is used to identify hidden correlations between users and items to predict the likelihood of a user purchasing a specific product. Secondly, SVD is used to reduce the dimensions of the user-item space. It generates a condensed representation, which is then utilized to compute neighborhoods. Based on this, the system can create a list of top-N recommended products for a given user.

A novel framework [4] is proposed for a top-N recommendation based on non-negative matrix tri-factorization (NMTF), which entails a data imputation module and a top-N recommendation module. The modified NMTF model includes a context factorization component that captures the contextual information in the recommendation process.

A novel approach [5] extends the traditional matrix factorization model by incorporating item metadata from a related domain, such as item descriptions, tags, or item reviews. The item metadata is used to learn a joint low-dimensional representation of the item content and the user-item interactions. The proposed approach also employs a cross-domain regularization term to encourage the similarity between the item embeddings in the two domains.

The experimental results on real-world datasets show that this approach surpasses other cutting-edge methods. This approach addresses the user cold start problem, especially when user interaction data is limited. It shows good scalability and can handle large-scale datasets. An SVD-based approach [6] estimating a topic model is advantageous for large-scale corpora because it requires minimal memory and has a low computational cost.

Deep learning-based matrix factorization techniques such as autoencoders and neural collaborative filtering have recently been proposed for recommender systems. These techniques employ neural networks to extract latent features of users and items from the user-item rating matrix. Overall, matrix factorization techniques are widely used and effective in generating personalized recommendations in recommender systems.

3. SVD (Singular Value Decomposition)

Singular Value Decomposition (SVD) [7,8] is a method of matrix factorization that breaks down a matrix into three constituent matrices: a left singular matrix, a diagonal matrix of singular values, and a suitable singular matrix. In other words, given an m x n matrix A, SVD factorizes it into three matrices.

$$A_{[mXn]} = U_{[mXr]}\Sigma_{[rXr]}(V_{[rXn]})^T$$

Where A is the input data matrix, which is vast but sparse; U is the left singular vector, which is considerable; Σ is singular values, which are sparse but small; r is the rank of matrix A, and V is a suitable singular vector, which is dense.

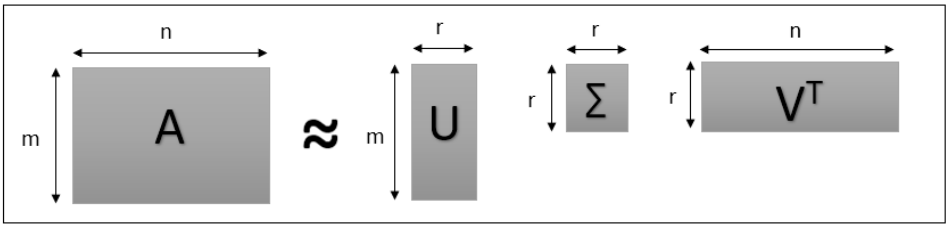

Fig. 1. Illustrating the SVD theorem.

In the context of recommender systems, the singular value decomposition (SVD) technique is used to anticipate missing data in a user-item matrix. The user-item matrix represents the ratings that users have given to items, where each row corresponds to a user, and each column corresponds to an item. SVD factorizes the user-item matrix into three matrices: a user-feature matrix, a feature-feature matrix, and an item-feature matrix. The user-feature matrix represents the latent features that characterize each user's preferences. In contrast, the item-feature matrix represents the latent features that distinguish the properties of each item. The feature-feature matrix represents the correlations between the latent features. Once the user-item matrix is factorized, the missing values can be predicted by computing the dot product of the corresponding rows of the user-feature and item-feature matrices. This predicts the user's rating for the item based on their preferences for the latent features and the item's properties for the same features.

SVD is a popular matrix factorization technique for recommender systems because it can handle sparse and incomplete data, often in real-world scenarios. It can also be used to identify hidden patterns and relationships in the user-item matrix, which can help improve the quality of recommendations.

One drawback of SVD is that it has optimal low-rank approximation. The computational time of SVD is high, resulting in a dense vector that takes more space.

4. U.V. Decomposition Algorithm

The U.V. decomposition is a matrix factorization technique that breaks down a matrix into separate lower-rank matrices. Starting with the utility matrix A, which has n rows

(representing users) and m columns (representing items), we aim to find matrices U (with n rows and d columns) and V (with d rows and m columns). For example, if we have a matrix A of size m x n, we can decompose it into two matrices, U and V, such that:

$$A = U \times V$$

Where U is of size m x k, V is of size k x n, and k is a minor rank than m and n. U.V. decomposition aims to find the values of U and V. This value must minimize the variance between A and the product of U and V.

5. CUR Algorithm

This matrix factorization technique selects a subgroup of rows and columns from the original matrix and creates a smaller matrix. The goal of the CUR algorithm is to generate a reduced matrix that retains the essential characteristics of the original matrix.

The CUR algorithm is a low-rank matrix approximation technique that uses column and row subset selection to reconstruct an approximation of the original matrix. The algorithm selects a subset of columns and rows based on their importance, computed using the cumulative scores of the matrix. In the CUR algorithm, given an input matrix A, the goal is to approximate it using a product of C, U, and R matrices such that.

$$A \approx C\ U\ R$$

The matrix C is constructed by selecting a random subset of columns from A, while R is built by choosing a random subset of rows from A. The matrix U is then computed as the product of the pseudoinverse of C, denoted as C^+, the input matrix A, and the pseudoinverse of R, denoted as R^+, that is,

$$U = C^+ A\ R^+$$

The matrix C contains the selected columns from A, while R contains the selected rows from A. The matrix U is the low-rank approximation of A obtained from A's selected columns and rows.

Therefore, the equation A = CUR represents the matrix decomposition where the original matrix A is approximated as the product of matrices C, U, and R. The matrix C contains selected columns of A, the matrix R contains selected rows of A, and U is the low-rank approximation of A obtained from C, U, and R.

The CUR algorithm is computationally efficient and effective for handling sparse matrices. The original paper [9] on the CUR algorithm was published around 2008. Since then, several algorithm variations have been proposed, such as the robust CUR algorithm and the adaptive CUR algorithm.

The CUR algorithm has been applied in various fields, including recommender systems, image and signal processing, and machine learning. The algorithm has been used in recommender systems to construct low-rank approximations of user-item rating matrices, leading to improved prediction. The CUR algorithm is a valuable technique for low-rank matrix approximation, particularly for sparse matrices. Its effectiveness and computational efficiency make it a valuable alternative to other matrix decomposition techniques.

6. Experimentation

In this section, we experimented with U.V. decomposition, SVD, and CUR techniques for recommender systems based on their RMSE and MAE performance on the MovieLens and Epinions dataset.

6.1. *Data sources*

We use two real-world datasets, the MovieLens 100K dataset (http://www.movieLens.org) and Epinions, to evaluate the effectiveness of the algorithms. The dataset comprises 100,000 ratings (1-5) from 943 users on 1682 movies. Each user has rated at least 20 movies [10]. The dataset utilized in our experiments is sourced from the Epinions.com website, which functions as a platform for consumer opinions. Epinions is a consumer opinion website. Here, the users can review and buy various products and rate them numerically on a scale from 1 to 5.

6.2. *Performance evaluation metrics*

We used two datasets to evaluate the performance of the three matrix factorization techniques. The dataset was split into training and test sets. The training set uses various optimization techniques to learn the user and item latent factors. In contrast, a test set is used to evaluate the performance of the learned model by comparing the predicted ratings to the actual ratings. Standard metrics for assessing matrix factorization techniques include mean absolute error (MAE), root mean square error (RMSE), precision, recall, and F1 score. RMSE and MAE measure the accuracy of predicted ratings. In contrast, precision, recall, and F1 scores measure the effectiveness of the recommendations in terms of how well they can identify relevant items for users.

Let m represent the number of actual ratings in an item set. The Mean Absolute Error is determined by averaging the absolute differences between the m pairs. Let $r1, r2, r3 \dots, r_m$ present the predicted ratings of users, and $a1, a2, a3 \dots, a_m$ denote the corresponding actual rating dataset of users.
Then MAE [11] is defined as:

$$MAE = \frac{\sum_{i=1}^{m} |r_{i-} a_i|}{m}$$

A smaller value of MAE indicates better accuracy [12]. To improve the accuracy of the techniques, the following hyperparameter is tuned:

- **n-factor** is the latent factor in the model. Increasing this value can improve accuracy but may lead to overfitting.
- **n-epoch** is the number of iterations of the optimization algorithm. Raising this value may enhance accuracy, but it can also result in overfitting.
- **lr-all** is the learning rate for all parameters. Increasing this value can help the algorithm converge faster and make it more likely to get stuck in local minima.

- **reg-all** is the regularization parameter for all parameters. Increasing this value can help prevent overfitting but can also reduce the accuracy of the proposed model.

Table 1 summarizes the time complexity of the three techniques. Overall, the time complexity of these techniques can vary widely depending on the specific implementation, the size of the input matrix, and the rank or the number of factors used for decomposition. In practice, the CUR algorithm can be much faster than SVD for large sparse matrices, especially when the rank is small, or the number of non-zero entries is small compared to the matrix size. However, SVD is often more accurate and can handle dense matrices and larger ranks at the expense of higher computational complexity.

Table 1. Time Complexity for all three algorithms.

Technique	Time Complexity
CUR decomposition	O(nm)
U.V. decomposition	O(knm)
Singular Value Decomposition	O(kn+m)

7. Results

The results indicate that U.V. decomposition outperformed SVD and CUR regarding mean absolute error and root mean square error performance. We found that U.V. decomposition has lower RMSE and MAE values than both datasets' SVD and CUR algorithms. Here, the smaller value indicates better performance. Table 2 indicates the RMSE values of comparison of three algorithms on the Epinions dataset. Table 3 reports the comparative values of RMSE for the three algorithms on the MovieLens dataset. U.V. Decomposition had the lowest RMSE value of 0.93. This indicates that it provided the most accurate recommendations on the testing dataset. SVD decomposition had a slightly higher RMSE value of 0.95, suggesting that it offered less accurate recommendations than U.V. Decomposition. CUR had the highest RMSE value of 1.00, indicating that it provided the least accurate recommendations among the three algorithms.

These results suggest that U.V. decomposition is a more effective algorithm than SVD and CUR for collaborative filtering-based recommender systems. This comparative analysis provides insights into U.V. decomposition, SVD, and CUR algorithm performance for recommender systems based on their RMSE and MAE performance on the MovieLens and Epinions dataset.

Table 2. MAE value for all three algorithms on MovieLens and Epinions datasets.

Technique	MovieLens Dataset	Epinions Dataset
CUR decomposition	0.74	0.79
U.V. decomposition	0.65	0.47
Singular Value Decomposition	0.69	0.72

Table 3. RMSE value for all three algorithms on MovieLens and Epinions datasets.

Technique	MovieLens Dataset	Epinions Dataset
CUR decomposition	0.98	1.00
U.V. decomposition	0.93	0.96
Singular Value Decomposition	0.95	0.98

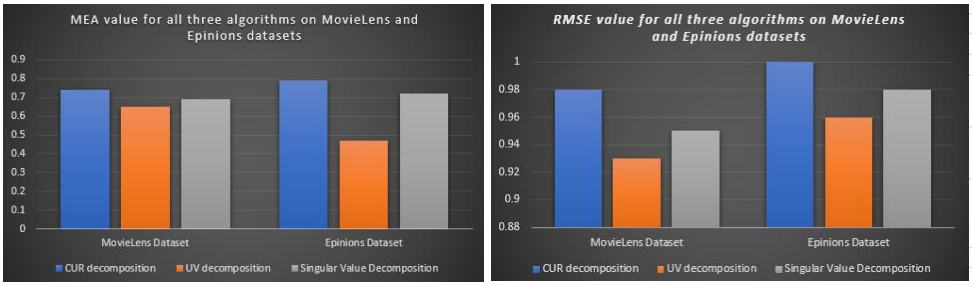

Fig. 2. Comparative analysis of the three techniques.

8. Conclusion

In conclusion, we have presented a comparative analysis of three matrix decomposition techniques: U.V. decomposition, SVD, and CUR algorithm for recommender systems based on their MAE and RMSE values using MovieLens and Epinions dataset.

It is essential to acknowledge that the performance of these algorithms may perform differently depending on the dataset and the particular problem under consideration. Therefore, a comparative analysis of multiple algorithms on different datasets is recommended to determine the most suitable algorithm for a given situation. Additionally, it is crucial to consider factors like computational efficiency, scalability, and interpretability when choosing an algorithm for a recommender system. SVD can be computationally expensive for large datasets, and CUR may not be as scalable as U.V. decomposition and SVD. SVD and U.V. decomposition are also more explainable since they directly factorize the user-item rating matrix into latent factor matrices of the user and item. Therefore, the selection of an algorithm should be based on a comprehensive evaluation of multiple factors. In future work, we will compare the performance of matrix factorization techniques, such as alternating least squares (ALS), non-negative matrix factorization (NMF), and stochastic gradient descent (SGD), to identify the most effective technique for the given dataset and recommendation task.

References

1. Y. Koren, R. Bell, and C. Volinsky, Computer **42**, 30 (2009). https://doi.org/10.1109/MC.2009.263
2. F. Camilli and M. Mézard, Phys. Rev. E **107,** ID 064308 (2023). https://doi.org/10.1103/PhysRevE.107.064308

3. B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, Application of Dimensionality Reduction in Recommender System - A Case Study - *Proc. of the Workshop on Knowledge Discovery in the Web* (WebKDD), 2000.
4. J. Tian and Y. Qu, A Novel Framework for Top-N Recommendation Based on Non-negative Matrix Tri-Factorization – *Proc. of the 24th Int. Conf. on Industrial Engineering and Engineering Management 2018* (2019). https://doi.org/10.1007/978-981-13-3402-3_36
5. I. Fernández-Tobías, I. Cantador, P. Tomeo, V. W. Anelli, and T. D. Noia, User Model. User-Adapted Interaction **29**, 443 (2019). https://doi.org/10.1007/s11257-018-9217-6
6. Z. T. Ke and M. Wang, J. Am. Stat. Assoc. **119**, 434 (2024). https://doi.org/10.1080/01621459.2022.2123813
7. G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R, A, Harshman et al., ACM SIGIR Forum **90**, 465 (1998). https://doi.org/10.1145/62437.62487
8. Y. A. Maneetah, S. M. Elsibai, A. A. Bouras, A. H. Alhabbh, and F. Elbadri, Sci. J. Faculty Sci.-Sirte University **4**, 80 (2024). https://doi.org/10.1016/j.envexpbot.2012.10.010
9. P. Drineas, M. W. Mahoney, and S. Muthukrishnan, SIAM J. Matrix Anal. Applicat. **30**, 844 (2008). https://doi.org/10.1137/07070471X
10. F. M. Harper and J. A. Konstan, Acm Transact. Interactive Intelligent Syst. **5**, ID 19 (2015). https://doi.org/10.1145/2827872
11. T. Widiyaningtyas, A. P. Wibawa, U. Pujianto, and W. Cacsarendra, Int. J. Intel. Eng. Syst. **17**, 180 (2024). https://doi.org/10.22266/ijies2024.0430.16
12. S. Gon, J. Software 5.7, (2010).
13. S. Zhang, L. Liu, Z. Chen, and H. Zhong, Knowledge-Based Syst. **183**, 104864 (2019). https://doi.org/10.1016/j.knosys.2019.07.035
14. P. Symeonidis and A. Zioupos, Matrix and Tensor Factorization Techniques for Recommender Systems (Springer International Publishing, New York, 2016). https://doi.org/10.1007/978-3-319-41357-0
15. Paterek and Arkadiusz, Improving Regularized Singular Value Decomposition for Collaborative Filtering - *Proc. of KDD cup and workshop* (2007).
16. A. Rajaraman and J. D. Ullman, Mining of Massive Data Sets (Cambridge University Press, UK, 2020).