International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(14s) 2026 pp. 1047-1057
https://doi.org/10.70917/ijcisim-2026-4320
Article

A Lightweight HITS-Assisted Matrix Factorization Framework for Sparse Recommendation Systems

Siva Ramakrishna Sakshi [1], Anuradha T [2]

[1]Department of CS&E, College of Sciences, Acharya Nagarjuna University, Nagarjuna Nagar, Guntur,522510, India
ramakrishna_sakshi@yahoo.co.in
[2]Department of CSE, RVR&JC College of Engineering, Gutur,522019, India
a_talasila@yahoo.com

Abstract: Recommendation systems have been assisting a variety of businesses using data analysis. Information from users is a strong basis to develop such systems, but obtaining sufficient responses is often challenging. Matrix factorization (MF) helps recommendation systems deal with this issue. However, sparse user–item interaction settings degrade the performance. The Hyperlink-Induced Topic Search (HITS) algorithm captures the structural importance within users and items. This model cannot investigate deeper preferences. This paper aims to provide a lightweight and interpretable framework that incorporates the merits of HITS as a preprocessing step to latent factor learning supported by MF. The MovieLens dataset was used to evaluate the proposed work. A three-stage evaluation protocol was employed, with a standard MF as baseline, the HITS model for recommendation, and then the sequential HITS followed by the MF framework. The proposed HITS→MF framework reduces RMSE by approximately 11.3% and MAE by 12.6% compared to baseline MF. Consistent gains in ranking are observed through Precision@10 and NDCG@10. The results demonstrate that HITS alone improves top-N ranking performance, and HITS followed by MF provides a better balance with respect to the evaluation metrics. The proposed framework improves robustness without introducing additional model complexity or auxiliary data, making it suitable for practical recommendation environments. The collaboration between structural graph information and latent factor learning can reduce sparsity without loss of performance. This collaborative approach is an efficient alternative to complex hybrid models, making it a good choice for practical recommendation system applications.
Keywords: Recommendation Systems, Data Sparsity, Matrix Factorization, HITS Algorithm, Graph-Based Ranking, Collaborative Filtering

1. Introduction

Recommendation systems are reliable technical assistants for digital platforms, enabling personalized content across e-commerce, online media streaming, social networking, and digital education. These systems learn data patterns to give suitable recommendations. Collaborative filtering (CF) is one of the most influential approaches and does not expect feature-level data [1,2].

Matrix factorization (MF) has been a major tool for decomposing the sparse user–item interaction matrix into low-dimensional latent representations. Here, we can capture hidden preference patterns that cannot be directly observable [3]. However, it is more sensitive to data sparsity. A practical challenge is to expect each user to interact with all items, resulting in sparse rating matrices. These conditions made the approach unstable, degraded prediction accuracy, and limited generalization capability [4–6].

Past research proposes several extensions to MF to address sparsity. Regularization, neighborhood-aware factorization, and implicit feedback modeling are some of these proposals [6,7]. They give better results, at the cost of careful parameter tuning. Deep learning–based models have demonstrated strong performance. They can model

complex and non-linear relationships [8,9]. Graph neural network (GNN) based models explicitly encode interaction topology and higher-order connectivity, promising good results on many benchmark datasets [10–12]. For effective outcomes, they demand rich computations and often lack interpretability, which limits their practical applicability [13].

Graph-based models consider users and items as two ends of a bipartite graph. These graph-based networks insist on structural relationships. By studying network interactions as bipartite graphs, relevant information can be propagated across users and items [14-16]. Hyperlink-Induced Topic Search (HITS) algorithm is one such approach that was originally developed for web link analysis [17]. The HITS algorithm enables recommendation systems to identify influential users and items, even when explicit rating information is limited [18].

While graph-based techniques are effective for ranking-oriented tasks, they are often applied either independently or combined with collaborative filtering models via late fusion strategies [19,20]. Such integration schemes typically do not directly influence the latent factor learning process and therefore do not fully exploit the complementary strengths of structural propagation and factor-based modeling. Moreover, relying solely on structural importance for rating prediction may lead to higher prediction error, as graph-based scores do not explicitly encode preference intensity.

These observations motivated the proposal of a lightweight, two-stage recommendation framework. HITS-based structural propagation is used as a preprocessing step before matrix factorization. In the first stage, hub and authority scores are computed on the user–item bipartite graph and used to generate structurally informed estimates of missing interactions. In the second stage, matrix factorization is applied to the updated interaction matrix to improve outcomes.

The proposed framework is intentionally designed to be simple, interpretable, and computationally efficient. It avoids deep architecture, auxiliary side information, and complex optimization pipelines beyond those required for standard MF. Experimental evaluation on the MovieLens dataset demonstrates that while HITS alone improves ranking performance, the proposed HITS→MF approach achieves a more balanced improvement by reducing prediction error while maintaining competitive top-N recommendation quality under sparse conditions.

This work supports:

A low-cost, two-stage HITS-assisted matrix factorization framework is introduced to address data sparsity in recommendation systems.

Structural propagation is incorporated as a preprocessing mechanism that stabilizes latent factor learning.

An ablation study compares MF, HITS, and the proposed HITS→MF approach using both accuracy-based and ranking-based metrics.

Experimental results demonstrate improved robustness under sparse conditions without increasing model complexity.

2. Related Work and Background

2.1 Collaborative Filtering and Matrix Factorization

MF is a key contributor to recommendation systems. It assists collaborative filtering, which relies on historical buying patterns [1,2]. MF is scalable and effective in capturing latent user and item representations [3]. In matrix factorization, the rating matrix (R) is represented as a product of latent factor matrices XT and Y, where X is a factor-by-user matrix, and Y is the factor-by-item matrix [9].

$$\begin{pmatrix} r_{11} & \cdots & r_{1n} \\ \vdots & \ddots\ \boldsymbol{Ratings(R)} & \vdots \\ r_{m1} & \cdots & r_{mn} \end{pmatrix} = \begin{bmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & X^T \ddots & \vdots \\ x_{m1} & \cdots & x_{mk} \end{bmatrix} * \begin{bmatrix} y_{11} & \cdots & y_{1n} \\ \vdots & Y \ddots & \vdots \\ y_{k1} & \cdots & y_{kn} \end{bmatrix} \qquad (1)$$

In its micro form, the equation for the estimation of ranks is:

$$\hat{r} = x^T * y \qquad (2)$$

Here, two matrices named factors are selected with random values, and the product is compared with actual figures to get the value of error. Further, the process continued to reduce the error. The process continues towards optimization.

The product matrix provides a prediction for ranks. The quality of prediction depends on the dimension of latent factors [21]. However, the performance of MF deteriorates significantly under sparse interaction conditions, particularly for cold-start users and infrequently rated items [4–6]. Enhancements such as regularization, neighborhood constraints, and implicit feedback modeling deal with these issues [6,7]. However, MF remains sensitive to extreme sparsity. r

2.2 Deep Learning–Based Recommendation Systems

Neural recommendation models supported by deep learning approaches capture complex nonlinear relationships between users and items [8,9]. This was further extended by graph neural networks with higher-order connectivity to explicitly model interaction topology [10–12]. Still, computational overhead is high, and interpretability is limited, reducing the practical deployment possibility [13].

2.3 Graph-Based Recommendation and Structural Propagation

Graph-based recommendation models enable relevant propagation. They use random walks, diffusion processes, or ranking algorithms [14–16]. They are effective in getting additional signals in sparse settings. The HITS algorithm does mutual reinforcement by treating users as hubs and items as authorities [17,18]. Although HITS-based methods improve ranking quality, they are less effective for precise rating prediction.

2.4 Hybrid Integration Strategies and Limitations

Researchers modeled combinations of multiple paradigms to build hybrid recommendation systems that overcome the individual limitations [19,20]. Most of these trials employ late fusion, without influencing internal model learning. Deep hybrid models introduce significant complexity. This situation demands integration frameworks that can allow structural information without increasing system complexity.

Recent surveys and empirical studies indicate that despite rapid advances in deep and graph neural recommendation models, concerns remain about scalability, interpretability, and deployment costs [22,23]. Furthermore, evidence suggests that simpler collaborative models continue to perform competitively when enhanced with appropriate structural signals, especially in sparse settings [24].

2.5 Positioning of the Proposed Work

Table 1 is a summary of key differences between latent factor models, graph-based ranking methods, and existing hybrid frameworks.

Table 1. Comparison of Sparsity Handling Techniques in Recommendation Systems

Approach	Method Type	Handles Sparsity	Computational Cost	Interpretability	Key Limitations
Standard MF [3]	Latent Factor	Limited	Low	Medium	Sensitive to sparsity
Regularized MF [6]	Latent Factor	Partial	Medium	Medium	Requires tuning
Deep CF Models [8]	Neural	Strong	High	Low	Complex, resource-intensive
Graph-Based Ranking (HITS) [9,13]	Graph	Strong	Medium	High	Weak rating prediction
Hybrid CF Models [14,15]	Hybrid	Strong	High	Medium	Feature-dependent
Proposed HITS → MF	Hybrid (Lightweight)	Strong	Low	High	—

This work proposes a recommendation model that uses HITS as a basic step, followed by Matrix factorization as a concluding step. This lightweight model aims to preserve the qualities of matrix factorization. By embedding structural importance into latent representation learning, the framework improves robustness without increasing computational overhead.

3. Methodology

The methodology consists of three components: (i) baseline matrix factorization, (ii) HITS-based structural propagation, and (iii) the proposed two-stage integration of HITS with matrix factorization.

3.1 Problem Formulation

Let denote the user–item rating matrix, where and represent the number of users and items, respectively. Each observed entry corresponds to a rating provided by the user for item , while missing entries indicate unobserved interactions. Due to real-world user behavior, is highly sparse. R∈RU×IUIruiu iR

The objective is to predict missing ratings accurately and generate high-quality top-N recommendations under sparse conditions.

3.2 Baseline: Matrix Factorization (MF)

Matrix factorization approximates the sparse rating matrix as the product of two low-dimensional latent matrices: R

$$R \approx \hat{R} = UV^T$$

where:

$U \in \mathbb{R}^{U \times K}$represents user latent factors,

$V \in \mathbb{R}^{I \times K}$represents item latent factors,

$K \ll \min(U, I)$is the latent dimension.

In this work, truncated singular value decomposition (SVD) is used for efficient and stable factorization. MF serves as the primary baseline due to its widespread adoption and strong predictive performance.

Limitation: Under high sparsity, MF struggles to learn reliable latent representations, particularly for users and items with limited interactions.

3.3 HITS-Based Structural Propagation

To exploit structural information in sparse interaction data, the user–item matrix is interpreted as a bipartite graph, where: users act as hubs, and items act as authorities. The Hyperlink-Induced Topic Search (HITS) algorithm iteratively computes hub and authority scores through mutual reinforcement:

$$a_i = \sum_{u \in U(i)} h_u \, , h_u = \sum_{i \in I(u)} a_i$$

where:

$a_i$denotes the authority score of item i,$h_u$denotes the hub score of users u,$U(i)$and $I(u)$represent connected nodes. The scores are normalized at each iteration to ensure numerical stability. Using the learned hub and authority scores, missing ratings are estimated by propagating influence across the bipartite graph, resulting in a structurally smoothed rating matrix R_{HITS}.

Strength:
HITS effectively captures popularity and influence patterns, improving ranking quality under sparsity.

Limitation:
Structural propagation alone lacks the ability to model preference intensity accurately.

3.4 Proposed Method: HITS → Matrix Factorization

To combine the strengths of structural propagation and latent factor learning, we propose a two-stage HITS→MF framework.

Step 1: HITS-Based Preprocessing

HITS is applied to the sparse rating matrix to generate structural estimates for missing interactions.

Step 2: Matrix Completion

Observed ratings are preserved, while missing entries are replaced with HITS-based estimates:

$$R'_{ui} = \begin{cases} R_{ui}, & \text{if } R_{ui} > 0 \\ R_{ui}^{\text{HITS}}, & \text{otherwise} \end{cases}$$

Step 3: Refactorization

Matrix factorization is reapplied to the updated matrix R', producing final predictions:

$$\hat{R} = U'V'^{T}$$

This design allows structural information to directly influence latent factor learning, resulting in improved robustness under sparse conditions.

3.5 Algorithm Descriptions

Algorithm 1: Matrix Factorization (MF)

Input: Sparse rating matrix R, latent dimension K
Output: Predicted rating matrix $\hat{R}$

1) Apply truncated SVD to R
2) Compute latent matrices $U$and V
3) Return $\hat{R} = UV^{T}$

Algorithm 2: HITS-Based Structural Propagation

Input: Rating matrix R, number of iterations T
Output: Structurally smoothed matrix R_{HITS}

1) Initialize hub and authority vectors with uniform values
2) Iteratively update authority and hub scores
3) Normalize scores after each iteration
4) Estimate missing ratings using propagated scores
5) Return R_{HITS}

Algorithm 3: Proposed HITS→MF Framework

Input: Sparse rating matrix R, latent dimension K
Output: Final predicted matrix $\hat{R}$

1) Apply HITS to compute R_{HITS}
2) Construct an updated matrix $R'$by filling in missing entries
3) Apply matrix factorization to R'
4) Return final predictions $\hat{R}$

3.6 System Flow Diagram

The overall workflow of the proposed framework is illustrated in Figure 1 and consists of the following stages:

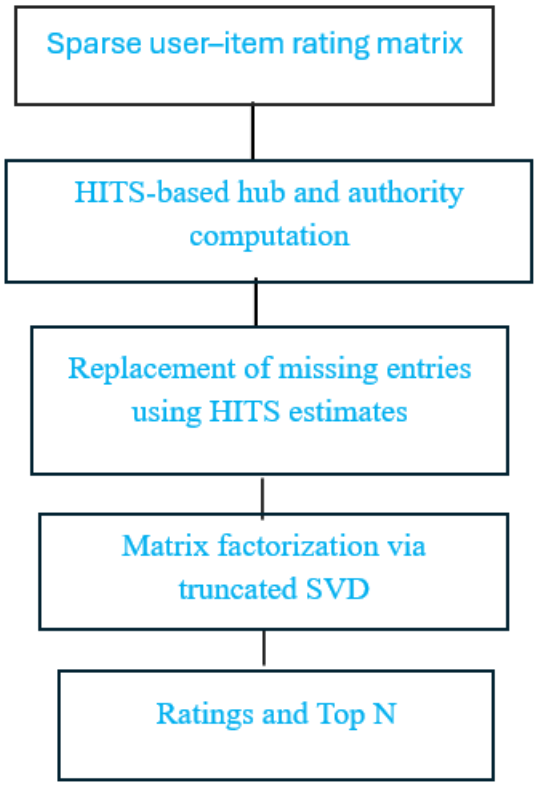

Fig. 1. System flow diagram of the proposed HITS → MF recommendation framework.

The input layer stores a sparse user–item rating matrix. The structural propagation module does HITS-based hub and authority computation. The matrix update module replaces missing entries using HITS estimates. The latent factor learning module does matrix factorization via truncated SVD. Finally, the output layer gives final rating predictions and top-n recommendations. This modular design ensures interpretability, scalability, and ease of integration into existing recommendation pipelines.

3.7 Computational Complexity

The proposed framework introduces minimal overhead compared to standard MF. HITS propagation has linear complexity with respect to observed interactions, while matrix factorization dominates runtime. Since HITS is applied only once as a preprocessing step, the overall computational cost remains comparable to MF, making the approach suitable for large-scale sparse datasets.

4. Experimental Setup

4.1 Dataset

Experiments are conducted on the widely used MovieLens dataset, which contains explicit user–item ratings on a numerical scale. The dataset represents a typical sparse recommendation scenario, where each user interacts with only a small fraction of available items. Such sparsity makes the dataset well-suited for evaluating recommendation robustness and sparsity-handling strategies.

Before model training, the dataset is transformed into a user–item rating matrix, where rows correspond to users and columns correspond to items. Missing ratings are represented as zero values. No additional side information or content features are used, ensuring that the evaluation focuses solely on collaborative and structural information.

4.2 Experimental Protocol

To ensure fair and robust evaluation, 5-fold cross-validation is employed. In each fold, the user–item matrix is partitioned into training and test subsets. Three methods are evaluated:

Matrix Factorization (MF): Standard latent factor model using truncated singular value decomposition.

HITS-Based Prediction: Structural propagation using hub and authority scores on the user–item bipartite graph.

Proposed HITS→MF Framework: Two-stage approach where HITS-based preprocessing is followed by matrix factorization.

This setup enables a controlled ablation study that isolates the contribution of structural propagation and its interaction with latent factor learning.

4.3 Implementation

Python is used with standard scientific computing libraries, including NumPy, SciPy, Pandas, and Scikit-learn. Matrix factorization is performed using truncated SVD with a fixed latent dimension . The HITS algorithm is run for a fixed number of iterations with L1 normalization to ensure numerical stability. To ensure reproducibility, fixed random seeds are used across all runs. K

4.4 Evaluation Metrics

To comprehensively evaluate recommendation quality, both prediction-based and ranking-based metrics are employed.

4.4.1 Prediction Accuracy Metrics

Prediction accuracy is measured using Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) over observed ratings in the test set.

$$\text{RMSE} = \sqrt{\frac{1}{|T|}\sum_{(u,i)\in T}(r_{ui} - \hat{r}_{ui})^2}$$

$$\text{MAE} = \frac{1}{|T|}\sum_{(u,i)\in T}|r_{ui} - \hat{r}_{ui}|$$

where T denotes the set of observed test ratings. T

4.4.2 Ranking-Based Metrics

While RMSE and MAE assess rating prediction accuracy, they do not fully capture recommendation usefulness. Therefore, ranking-based metrics are also considered. For each user, items are ranked according to predicted scores, and the top 10 unseen items are selected. Precision@10 measures the relevant ratio with the top 10. Normalized Discounted Cumulative Gain (NDCG@10) evaluates ranking quality by assigning higher importance to correctly ranked items at higher positions. A rating threshold is used to define relevance, and metrics are averaged across users.

4.5 Evaluation Summary

By combining prediction-based and ranking-based metrics, the evaluation framework balances numerical accuracy and recommendation effectiveness. This comprehensive evaluation enables meaningful comparison between MF, HITS-based prediction, and the proposed HITS→MF framework under sparse conditions.

5. Results and Discussion

A comparison among standard matrix factorization (MF), HITS-based recommendation, and the proposed HITS→MF framework is implemented.

5.1 Quantitative Results

Table 2 summarizes the performance of all evaluated methods using both prediction-based and ranking-based metrics.

Table 2. The performance evaluation of MF, HITS, and the proposed Model on the MovieLens dataset.

Method	RMSE	MAE	Precision@10	NDCG@10
Matrix Factorization (MF)	3.4455	3.2387	0.3183	0.3345
HITS-based Recommendation	3.2225	3.0521	**0.4102**	**0.4302**
HITS → MF (Proposed)	**3.0554**	**2.8314**	0.3581	0.3717

MF achieves an RMSE of 3.445 and an MAE of 3.239, demonstrating reasonable rating prediction performance but moderate ranking quality. HITS-based prediction reduces RMSE to 3.222 and MAE to 3.052, indicating that structural propagation improves robustness under sparsity. Additionally, HITS achieves the highest ranking performance, with Precision@10 = 0.410 and NDCG@10 = 0. 430. The proposed HITS→MF framework achieves the best prediction accuracy, with an RMSE of 3.055 and an MAE of 2.831, while maintaining competitive ranking performance (Precision@10 = 0.358, NDCG@10 = 0.372). These results confirm that a combination of structural information and latent factor learning is a win-win strategy.

5.2 Analysis of Prediction Accuracy

The improvement in prediction accuracy achieved by the proposed framework can be attributed to the stabilizing effect of HITS-based preprocessing. In sparse matrices, MF struggles to learn reliable latent factors due to the insufficient number of observations for many users and items. HITS can offer good propagation, and latent factor learning becomes more stable leading to lower RMSE and MAE.

5.3 Analysis of Ranking Performance

HITS's graph-structured approach achieves the highest Precision@10 and NDCG@10.The proposed HITS→MF framework slightly underperforms HITS in ranking metrics but still improves upon MF. Nevertheless, the proposed method maintains competitive ranking performance while significantly improving prediction accuracy, offering a balanced solution.

5.4 Ablation Study and Synergistic Effects

An ablation study comparing MF, HITS, and HITS→MF provides insight into the contribution of each component. MF alone captures latent preference patterns but is sensitive to sparsity. HITS alone leverages structural importance but lacks the ability to model preference intensity. The proposed HITS→MF framework combines these strengths by using HITS as a preprocessing step rather than a standalone predictor or late fusion component.

The observed improvements indicate a synergistic effect, where structural propagation enhances the learning environment for matrix factorization. This synergy confirms that HITS contributes meaningful information beyond what MF can capture on sparse data.

5.5 Discussion and Practical Implications

A practical balance between accuracy, ranking quality, and computational efficiency makes the model suitable for deployment in real-world systems where computational resources and transparency are important considerations.

5.6 Cross-Validation Analysis

Cross-validation evaluation assesses the robustness of the proposed approach. A fivefold cross-validation aims to examine the consistency of prediction performance across different data splits. The comparative results are illustrated in Figures 2 and 3.

Figure 2 presents the RMSE comparison across five cross-validation folds for the two approaches.

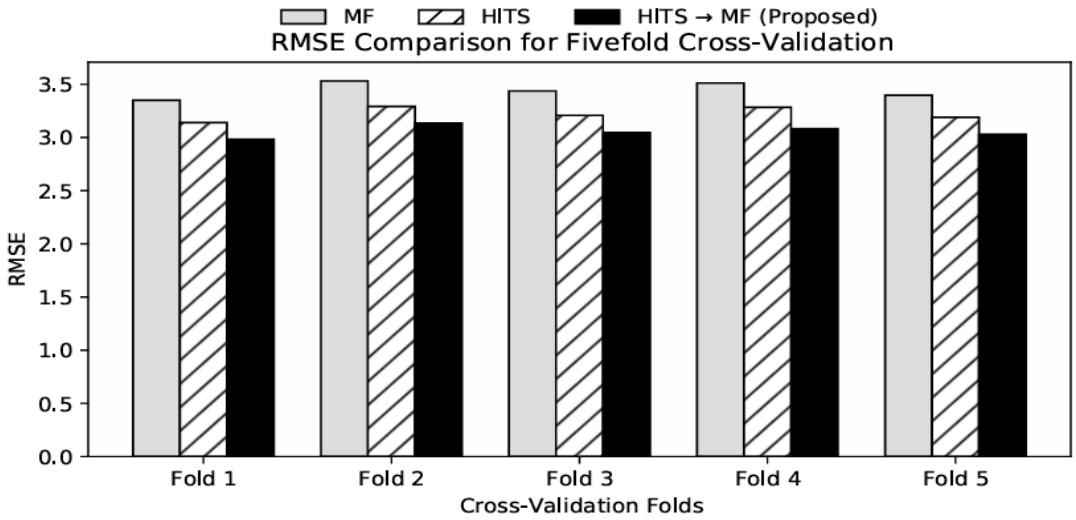

Fig. 2. RMSE comparison for fivefold cross-validation.

The RMSE values are computed independently at each fold. It can be observed that, across all five folds, the proposed method consistently achieves lower RMSE values compared to the existing MF method, indicating improved prediction accuracy and enhanced stability under varying training–testing partitions.

Figure 3 shows the MAE comparison across five cross-validation folds.

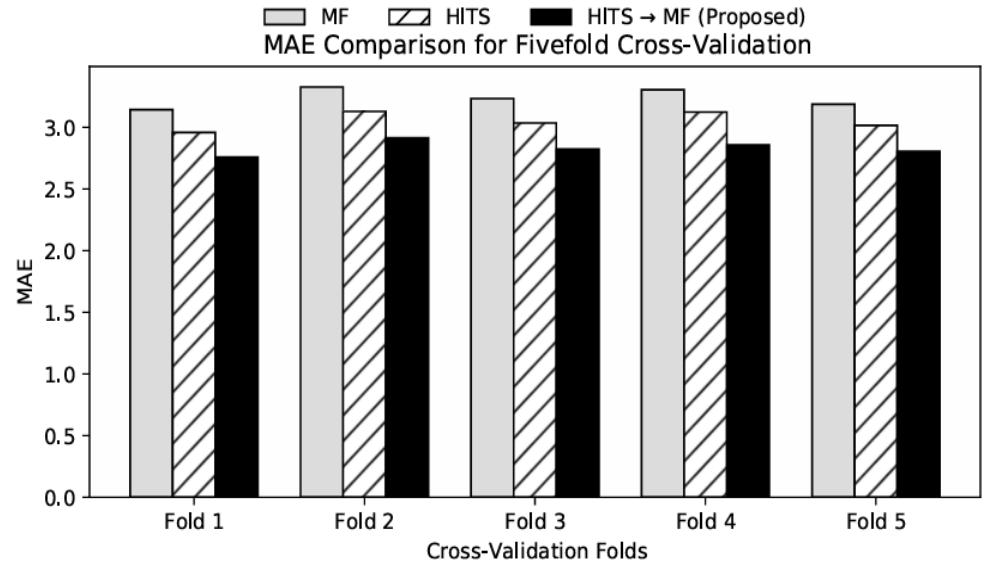

Fig. 3. MAE comparison for fivefold cross-validation.

The observed MAE trend also favors the proposed approach. This elevates the more reliable and robust performance of the proposed method.

6. Conclusion and Future Work

This paper proposes a combination of structural informative matrix factorization to overcome the existing limitations of MF. Though advanced models like deep learning and latent fusion are in application, they are more complex. In this context, the proposed framework offers an interpretable and computationally efficient technical companion to recommendation systems. Experimental results on the MovieLens dataset exhibited desired metric values. It is observed that structural propagation and latent factor learning play complementary roles in providing

robustness and improved preferability. This synergy allows the proposed method to maintain competitive ranking performance while significantly improving rating prediction accuracy.

Future Work

This work has possibilities for further extensions. Personalized weighting strategies could be introduced based on user activity levels or sparsity patterns. The framework can be extended to build fairness-aware models. Overall, this study demonstrates that thoughtfully integrating classical graph-based techniques with latent factor models remains a viable and effective strategy for addressing sparsity in Recommendation systems.

Statements *and Declarations*

Funding

The authors did not receive any funding for this research.

Competing Interests

The authors declare no competing financial or non-financial interests.

Data and Code Availability:

The dataset used in this study is publicly available. Code will be shared upon reasonable request.

Author's contribution statement

Sakshi Siva Ramakrishna: Experimental work, investigation, data gathering, writing review, and editing. T Anuradha: examine and correct the manuscript, supervision, and writing review.

References

1. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994).GroupLens: An open architecture for collaborative filtering of netnews.Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW), pp. 175–186.
2. Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007).Collaborative filtering recommender systems.In The Adaptive Web (pp. 291–324). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-72079-9-9
3. Koren, Y., Bell, R., & Volinsky, C. (2009).Matrix factorization techniques for recommender systems.IEEE Computer, 42(8), 30–37. https://doi.org/10.1109/MC.2009.263
4. Adomavicius, G., & Tuzhilin, A. (2005).Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions.IEEE Transactions on Knowledge and Data Engineering, 17(6), 734–749. https://doi.org/10.1109/TKDE.2005.99
5. Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013).Recommender systems survey.Knowledge-Based Systems, 46, 109–132. https://doi.org/10.1016/j.knosys.2013.03.012
6. Hu, Y., Koren, Y., & Volinsky, C. (2008).Collaborative filtering for implicit feedback datasets.Proceedings of the IEEE International Conference on Data Mining (ICDM), pp. 263–272.https://doi.org/10.1109/ICDM.2008.22
7. Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009).BPR: Bayesian personalized ranking from implicit feedback.Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI).
8. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017).Neural collaborative filtering.Proceedings of the International World Wide Web Conference (WWW), pp. 173–182. https://doi.org/10.1145/3038912.3052569
9. Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019).Deep learning based recommender system: A survey and new perspectives.IEEE Access, 7, 140–158. https://doi.org/10.1109/ACCESS.2018.2884086
10. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., & Wang, M. (2020).LightGCN: Simplifying and powering graph convolution networks for recommendation. Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 639–648. https://doi.org/10.1145/3397271.3401063
11. Wang, X., Jin, H., Zhang, Y., & Chua, T.-S. (2021).Disentangled graph collaborative filtering.Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 100–109. https://doi.org/10.1145/3404835.3462868
12. Zhou, K., Wang, H., Zhang, W., & Yu, Y. (2021).Self-supervised graph learning for recommendation.Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 131–140. https://doi.org/10.1145/3404835.3462862
13. Ferrari Dacrema, M., Cremonesi, P., & Jannach, D. (2021). Are we really making much progress? A worrying analysis of recent neural recommendation approaches.ACM Transactions on Information Systems, 39(1), Article 5. https://doi.org/10.1145/3432726
14. Zhou, T., Ren, J., Medo, M., & Zhang, Y. (2007).Bipartite network projection and personal recommendation.Physical Review E, 76(4), 046115. https://doi.org/10.1103/PhysRevE.76.046115

15. Lü, L., Medo, M., Yeung, C. H., Zhang, Y., Zhang, Z.-K., & Zhou, T. (2012).Recommender systems based on complex networks.Physics Reports, 519(1), 1–49. https://doi.org/10.1016/j.physrep.2012.02.006
16. Wu, L., He, X., Wang, X., & Zhang, Y. (2020).Graph convolutional networks for recommendation: A survey.IEEE Transactions on Knowledge and Data Engineering, 34(1), 1–20. https://doi.org/10.1109/TKDE.2020.2981333
17. Kleinberg, J. M. (1999).Authoritative sources in a hyperlinked environment.Journal of the ACM, 46(5), 604–632. https://doi.org/10.1145/324133.324140
18. Deng, S., Huang, L., Xu, G., & Wu, X. (2014).A HITS-based recommendation algorithm. Expert Systems with Applications, 41(4), 1891–1898. https://doi.org/10.1016/j.eswa.2013.08.066
19. Burke, R. (2002).Hybrid recommender systems: Survey and experiments.User Modeling and User-Adapted Interaction, 12(4), 331–370. https://doi.org/10.1023/A:1021240730564
20. Deldjoo, Y., Schedl, M., Cremonesi, P., & Pasi, G. (2020).Hybrid recommender systems: A survey of methodologies and challenges.ACM Computing Surveys, 52(6), Article 112. https://doi.org/10.1145/3377450
21. Chen, H.-H., & Chen, P. (2019). Differentiating regularization weights – A simple mechanism to alleviate cold start in recommender systems. ACM Transactions on Knowledge Discovery from Data,13(1), Article8,1–22. https://doi.org/10.1145/3285954
22. Zhang, S., Yao, L., Sun, A., & Tay, Y. (2022). Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys, 54(1), Article 12. https://doi.org/10.1145/3453154
23. Wang, X., He, X., Wang, M., Feng, F., & Chua, T.-S. (2022). Neural graph collaborative filtering: A survey. IEEE Transactions on Knowledge and Data Engineering, 34(11), 5163–5179. https://doi.org/10.1109/TKDE.2021.3101857
24. Ferrari Dacrema, M., Boglio, S., & Cremonesi, P. (2023). The unreasonable effectiveness of nearest neighbors in recommender systems. ACM Transactions on Information Systems, 41(2), Article 30. https://doi.org/10.1145/3570471