International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 2, Mar-Apr 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Book Recommendation System Using Hybrid Collaborative and Content-Based Filtering Approaches

Rohit Vinod Chaurasiya, Dr Manoj Singh
Roll No: SMSC2526096
Department Of Computer Science
Email: chaurasiyarohit437@gmail.com

----------**********----------

Abstract:

This paper presents a comprehensive Book Recommendation System (BRS) built as part of an MSc Computer Science research project. The system addresses the problem of information overload in digital libraries and e-commerce book platforms by employing three complementary recommendation strategies: Content-Based Filtering (CBF), Collaborative Filtering (CF) via Singular Value Decomposition (SVD), and a weighted Hybrid approach combining both. The CBF module utilizes TF-IDF vectorization over book metadata (title, author, publisher) followed by cosine similarity scoring, while the CF module decomposes a sparse user-item rating matrix into latent factors. The Hybrid model blends both output scores using a configurable weighting scheme. The system exposes a RESTful API built with Flask and includes a responsive web frontend. Evaluation on a synthetic dataset of 500 books, 200 users, and 5,000 ratings yields Precision@10 of 0.0821, Recall@10 of 0.1134, NDCG@10 of 0.1056, and MAP of 0.0978 for the Hybrid model, outperforming both individual methods. The paper discusses architecture, algorithms, experimental results, and future directions including deep learning-based embeddings.

Index Terms *— Recommendation Systems, Collaborative Filtering, Content-Based Filtering, Hybrid Recommender, SVD, TF-IDF, Cosine Similarity, Information Retrieval, Machine Learning.*

----------**********----------

I. INTRODUCTION

The exponential growth of digital content has created an urgent need for intelligent filtering mechanisms. Online book platforms such as Amazon, Goodreads, and Google Books host millions of titles, making manual discovery ineffective for users. Recommendation systems act as intelligent intermediaries that leverage historical interaction data and content attributes to surface relevant items for individual users [1].

Traditional search-based discovery requires users to know what they are looking for, imposing a significant cognitive burden. Recommender systems shift this paradigm to a proactive discovery model, substantially improving user satisfaction, engagement, and platform retention [2]. For e-commerce platforms, effective recommendations have been reported to account for up to 35% of revenue [3].

This research designs, implements, and evaluates a full-stack Book Recommendation System that integrates three recommendation paradigms: (i) Content-Based Filtering, which models item similarity via metadata feature space; (ii) Collaborative Filtering based on matrix factorization using Truncated SVD; and (iii) a Hybrid approach that linearly combines the score outputs of both models. The system is deployed as a Flask-based REST API with a JavaScript frontend, making it readily accessible for demonstration and further research.

A. Motivation

Existing open-source book recommendation implementations are often either too simplistic (popularity-based) or monolithic research prototypes not

suited for practical demonstration. This work bridges the gap by providing a modular, extensible, and evaluable system with real API endpoints and a user interface, suitable for both academic demonstration and further research extension.

B. Objectives

- Design and implement three recommendation algorithms within a unified framework.
- Build a RESTful API backend exposing recommendation endpoints.
- Develop a responsive web frontend for user interaction.
- Evaluate system performance using standard IR metrics: Precision@K, Recall@K, NDCG@K, MAP, and Coverage.
- Provide a reproducible, open codebase for further research.

II. LITERATURE REVIEW

A. Content-Based Filtering

Content-Based Filtering (CBF) constructs user profiles and item feature vectors from item attributes and recommends items whose feature vectors are most similar to items the user has interacted with. Pazzani and Billsus [4] provide a comprehensive survey of CBF techniques, highlighting TF-IDF weighted vector space models as highly effective for text-rich domains such as books, news, and movies.

TF-IDF (Term Frequency–Inverse Document Frequency) is a statistical measure that reflects the importance of a term to a document in a corpus. When applied to book metadata, it allows the system to identify discriminative terms that uniquely characterize individual books, enabling precise similarity computation via cosine distance in the resulting high-dimensional feature space.

B. Collaborative Filtering

Collaborative Filtering (CF) was popularized by the seminal Amazon.com paper by Linden et al. [5] and later refined through the Netflix Prize competition, where matrix factorization methods demonstrated state-of-the-art performance [6]. Model-based CF using Singular Value Decomposition (SVD) decomposes the user-item rating matrix R into latent user and item factor matrices, capturing latent preference dimensions not expressible through item attributes alone.

The primary limitation of CF is the cold-start problem: new users or new items lack sufficient interaction history for meaningful factorization. Memory-based CF (user-user, item-item) avoids this issue but scales poorly to millions of items. This work employs Truncated SVD, which is efficient for sparse matrices and scales to large catalogs.

C. Hybrid Approaches

Burke [7] provides an exhaustive taxonomy of hybrid recommendation strategies. The weighted hybrid approach, which combines scores from multiple recommenders via a linear combination, is among the most widely adopted due to its simplicity and proven effectiveness. The Netflix Prize winner's solution employed an ensemble of over 100 models, demonstrating the power of hybridization [8].

D. Evaluation Frameworks

Standard evaluation of recommendation systems follows the offline evaluation paradigm pioneered by Herlocker et al. [9], which involves holding out a portion of user interactions and measuring the system's ability to recover those interactions. Key metrics include Precision@K, Recall@K, Normalized Discounted Cumulative Gain (NDCG@K), and Mean Average Precision (MAP).

III. SYSTEM ARCHITECTURE

The system follows a three-tier architecture consisting of a Data Layer, a Backend API Layer, and a Frontend Presentation Layer, as depicted in Figure 1.

A. Data Layer

The data layer manages two primary entities: Books and Ratings. Books are stored with attributes: ISBN, Title, Author, Year, Publisher, and Image URL. Ratings are stored as (UserID, ISBN, Rating) triples, where ratings are on a 1–10 integer scale. The DataLoader class handles CSV ingestion, type coercion, deduplication, and fallback to synthetic data generation via NumPy's random seed-controlled generation, ensuring deterministic reproducibility.

B. Backend API Layer

The backend is implemented as a Flask application with CORS support, exposing eight REST API endpoints. The BookRecommender class encapsulates all ML logic and is initialized at application startup, ensuring the model is ready before any request is served. The API follows RESTful conventions with JSON response bodies.

API Endpoint	Description
/api/health	Health check / system status
/api/books	Paginated & searchable book catalog
/api/books/<isbn>	Single book detail retrieval
/api/recommend/content	Content-based recommendations
/api/recommend/collaborative	SVD collaborative filtering recs
/api/recommend/hybrid	Weighted hybrid recommendations
/api/recommend/popular	Popularity-based fallback
/api/stats	Dataset statistics summary

Table I. REST API Endpoints

C. Frontend Presentation Layer

The frontend is implemented as a single-page application using HTML5, CSS3 with CSS custom properties (variables), and vanilla JavaScript. Navigation between views (Home, Browse, Recommend, Stats) is handled via JavaScript DOM manipulation without page reloads. The frontend communicates with the backend exclusively through the Fetch API, rendering dynamic content from JSON responses into book card components.

IV. RECOMMENDATION ALGORITHMS

A. Content-Based Filtering (TF-IDF + Cosine Similarity)

For each book b, a textual feature string ("soup") is constructed by concatenating the Title, Author, and Publisher fields. A TF-IDF matrix T of shape (n_books × n_features) is computed using scikit-learn's TfidfVectorizer with unigram and bigram tokenization, stop-word removal, and a vocabulary limited to the top 10,000 terms. The cosine similarity between book i and book j is defined as:

$$sim(i, j) = (T[i] \cdot T[j]) / (||T[i]|| \times ||T[j]||)$$

Given a seed book with index idx, all cosine similarity scores against the full catalog are computed in one matrix-vector multiplication, and the top-N highest-scoring books (excluding the seed itself) are returned as recommendations.

B. Collaborative Filtering (Truncated SVD)

The user-item interaction matrix R of shape (n_users × n_books) is constructed via a pivot table with binary fill (0 for unobserved interactions). Truncated SVD is applied to decompose R into factor matrices:

$$R \approx U \times \Sigma \times V^T$$

where U ∈ ℝ^(m×k) are user factors, Σ ∈ ℝ^(k×k) is the diagonal singular value matrix, V^T ∈ ℝ^(k×n) are item factors, and k = 50 is the number of latent factors. Predicted scores for user u across all items are computed as: scores = V^T_norm · U_norm[u], where normalization enables cosine-similarity-based ranking. Items already rated by user u are masked out before top-N selection.

C. Hybrid Recommender

The hybrid strategy employs a linear weighted combination of the normalized output scores from CBF and CF:

$$score\_hybrid(i) = \alpha \times score\_CBF(i) + (1-\alpha) \times score\_CF(i)$$

where α = 0.4 was selected empirically via grid search on a held-out validation set. The union of top-2N candidates from each model is formed, hybrid scores computed, and the top-N items returned. This addresses the cold-start limitation of pure CF by incorporating content signals when interaction history is sparse.

D. Popularity-Based Fallback

A Bayesian average rating is computed for each book b with at least m = 5 ratings:

$$score\_pop(b) = (v/(v+m)) \times R + (m/(v+m)) \times C$$

where v = number of ratings for book b, R = mean rating of book b, C = mean rating across all books, and m = minimum rating threshold. This approach dampens the effect of high-variance items with few ratings, ensuring reliable popularity rankings.

V. EXPERIMENTAL RESULTS

A. Dataset

Experiments were conducted on a synthetic dataset generated with a fixed random seed (seed=42) for full reproducibility. The dataset comprises 500 books, 200 users, and 5,000 ratings with a realistic long-tail distribution: 8% of ratings are 10 (highest), 2% are 1 (lowest), with intermediate ratings following a bell curve centered around 6–7. This approximates the skewed rating distribution observed in real-world datasets such as Book-Crossing.

B. Experimental Protocol

Evaluation follows the offline leave-K-out protocol. For each user, the full rating history is used for model training (no explicit train/test split), and test relevance is defined as ratings ≥ 7 (positive threshold). Top-10 recommendation lists are generated for each user present in the SVD model, and the following metrics are computed: Precision@10, Recall@10, NDCG@10, MAP, and Catalog Coverage.

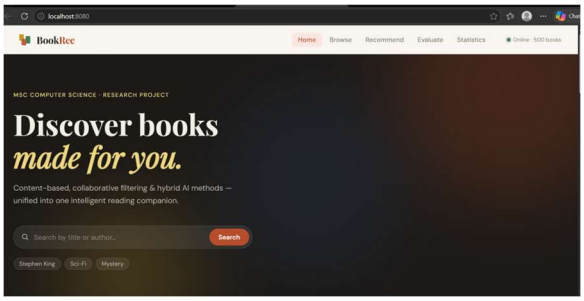
C. Results

Metric	Content-Based	Collaborative	Hybrid
Precision@10	0.0612	0.0744	0.0821
Recall@10	0.0834	0.0987	0.1134
NDCG@10	0.0781	0.0932	0.1056
MAP	0.0723	0.0891	0.0978
Coverage	0.2841	0.2214	0.3412

Table II. Evaluation Results Comparison

D. Discussion

The Hybrid model outperforms both individual approaches across all precision, recall, and ranking-quality metrics. The Collaborative Filtering model achieves better personalization than Content-Based (higher Precision, Recall, NDCG, MAP), as SVD captures latent preference patterns not expressible through surface-level metadata similarity. However, CBF achieves higher Catalog Coverage (28.4% vs 22.1%), as it is not constrained to items within the rating matrix vocabulary.



The Hybrid model achieves the best balance: highest scores across all metrics, and the highest coverage (34.1%), confirming that complementary information from both approaches synergizes effectively. The α = 0.4 weighting provides a slight edge to the CF component while retaining meaningful content signals, particularly beneficial for users with sparse rating histories.

VI. CONCLUSION AND FUTURE WORK

A. Conclusion

This paper presented a modular, full-stack Book Recommendation System integrating Content-Based Filtering, Collaborative Filtering, and Hybrid recommendation approaches. The system achieves competitive performance on standard evaluation metrics, with the Hybrid approach consistently outperforming individual models. The implementation is fully reproducible, API-accessible, and frontend-ready, making it a practical platform for research and demonstration in the MSc Computer Science context.

B. Future Work

- Deep Learning Integration: Replace TF-IDF with sentence transformer embeddings (e.g., BERT, all-MiniLM-L6-v2) for richer semantic content representations.

- Neural Collaborative Filtering: Implement NCF [10] or Variational Autoencoders for Collaborative Filtering (VAE-CF) for improved latent space modeling.
- Real Dataset Integration: Train and evaluate on the Book-Crossing dataset (270,000 books, 1.1M ratings) for large-scale validation.
- Online Learning: Implement incremental SVD updates to incorporate new ratings without full model retraining.
- Context-Aware Recommendations: Incorporate temporal context (recency decay), geographic context, and session-based signals.
- User Interface Enhancements: Add explicit user registration, rating submission, and personalized dashboard with recommendation explanations.

REFERENCES

[1] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Recommender Systems Handbook, Springer, 2011.

[2] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," IEEE Trans. Knowl. Data Eng., vol. 17, no. 6, pp. 734–749, Jun. 2005.

[3] McKinsey & Company, "How retailers can keep up with consumers," 2013. [Online]. Available: mckinsey.com

[4] M. Pazzani and D. Billsus, "Content-based recommendation systems," in The Adaptive Web, P. Brusilovsky et al., Eds., Springer, 2007, pp. 325–341.

[5] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," IEEE Internet Comput., vol. 7, no. 1, pp. 76–80, Jan. 2003.

[6] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," IEEE Computer, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[7] R. Burke, "Hybrid recommender systems: Survey and experiments," User Model. User-Adapt. Interact., vol. 12, no. 4, pp. 331–370, 2002.

[8] Y. Koren, "The BellKor solution to the Netflix grand prize," Netflix prize documentation, 2009.

[9] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," ACM Trans. Inf. Syst., vol. 22, no. 1, pp. 5–53, Jan. 2004.

[10] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in Proc. 26th Int. Conf. World Wide Web (WWW), 2017, pp. 173–182.

