INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL ARTICLE

Contextualized Movie Recommendation Framework to Improve User Experience for OTT

Biswaranjan Jena[1], Debahuti Mishra[1]*, Smitaprava Mishra[2]

1 Department of Computer Science and Engineering, Siksha 'O' Anusandhan (Deemed to be) University, Bhubaneswar, Odisha, India
2 Department of Computer Science and Information Technology, Siksha 'O' Anusandhan (Deemed to be) University, Bhubaneswar, Odisha, India

OPEN ACCESS

Received: 09/05/2025
Accepted: 23/06/2025
Published: 12/07/2025

Citation: Jena B, Mishra D, Mishra S (2025) Contextualized Movie Recommendation Framework to Improve User Experience for OTT. Indian Journal of Science and Technology 18(26): 2124-2134. https://doi.org/10.17485/IJST/v18i26.869

* **Corresponding author**.

debahutimishra@soa.ac.in

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: The purpose of the present investigation is to build a contextual recommendation framework based on user history and proven machine learning algorithms to improve user experience. New OTT entrants can adopt the findings to create their recommendation system or monetize it for other providers. **Methods:** It follows a hybrid approach where standard, proven algorithms used by large OTT providers and other researchers are evaluated using open-source technologies on the MovieLens-1M dataset. The proposed model describes additional factors like user feedback, contextual search and key performance indicators to evaluate the recommendation framework for improved user experience. **Findings:** As per the findings, a hybrid framework of Content-Based Filtering, Collaborative Filtering (CF), SVD++ and RBM can provide 80% of the tray requirements of modern OTT platforms. By using open-source libraries and various optimization techniques not only we can achieve a lower error rate, but we can improve user experience by considering other factors like novelty, diversity etc. **Novelty and application:** The manuscript provides the minimal set of algorithms that can provide the maximum tray requirement of any OTT platform, along with key performance indicators. It also describes contextual time-based tags generated from a sample video, which can be searched by the user in near real time. The proposed service-based framework once implemented, can also be passed to other providers for monetization.

Keywords: Recommendation; OTT; Contextual Search; Content Based Filtering; CF; SVD

1 Introduction

OTT is growing at a very fast pace [1], and it's imperative to keep users engaged with the platform. Most of the OTT platforms like Netflix, Amazon Prime, Jio Hotstar, etc., provide video on demand (VOD) services and earn their revenue either via subscription (SVOD) or advertisement (AVOD). To keep users engaged to the platform, providers

have to offer the best of the best content, but it's also critical to provide them recommendations on top of the existing content they have. There are multiple machine learning algorithms like content-based filtering (CF)[2], matrix factorization using Singular Value Decomposition (SVD)[3]-[4] and collaborative filtering[5] which can help to generate recommendations for different trays. A content-based filtering technique can generate recommendations on specific content metadata like genre, cast, language, etc. by filtering content metadata and filling in the trays like comedy, horror, action, etc. as illustrated by Sahu S. et al.[6]. Collaborative filtering works on the principle of other users, like the specific user for whom the recommendation is to be generated. It can be a user-to-user relation or item-to-item relation and can fill in the trays like "as other users watched" or "as you watched <<movie>>". Advanced deep learning algorithms using Restricted Boltzmann Machine[7] (RBM) and Autoencoders[3]-[8] can also be used to generate recommendations. Subsequent sections describe the outcome of implementing these algorithms.

2 Methodology

2.1 Proposed Framework

The research aims to introduce a movie recommendation framework that can provide an improved user experience to users. The proposed framework also gives insight into achieving features provided by large OTT enterprises by using a minimal set of machine learning algorithms. It also provides various evaluation criteria that need to be considered while designing an efficient recommendation system for viewers. The proposed recommendation framework (Figure 1) is built and evaluated using existing open-source Python libraries which can be adapted by new entrants with minimal cost, without any vendor lock-in, like relying upon GenAI services provided by large cloud providers.

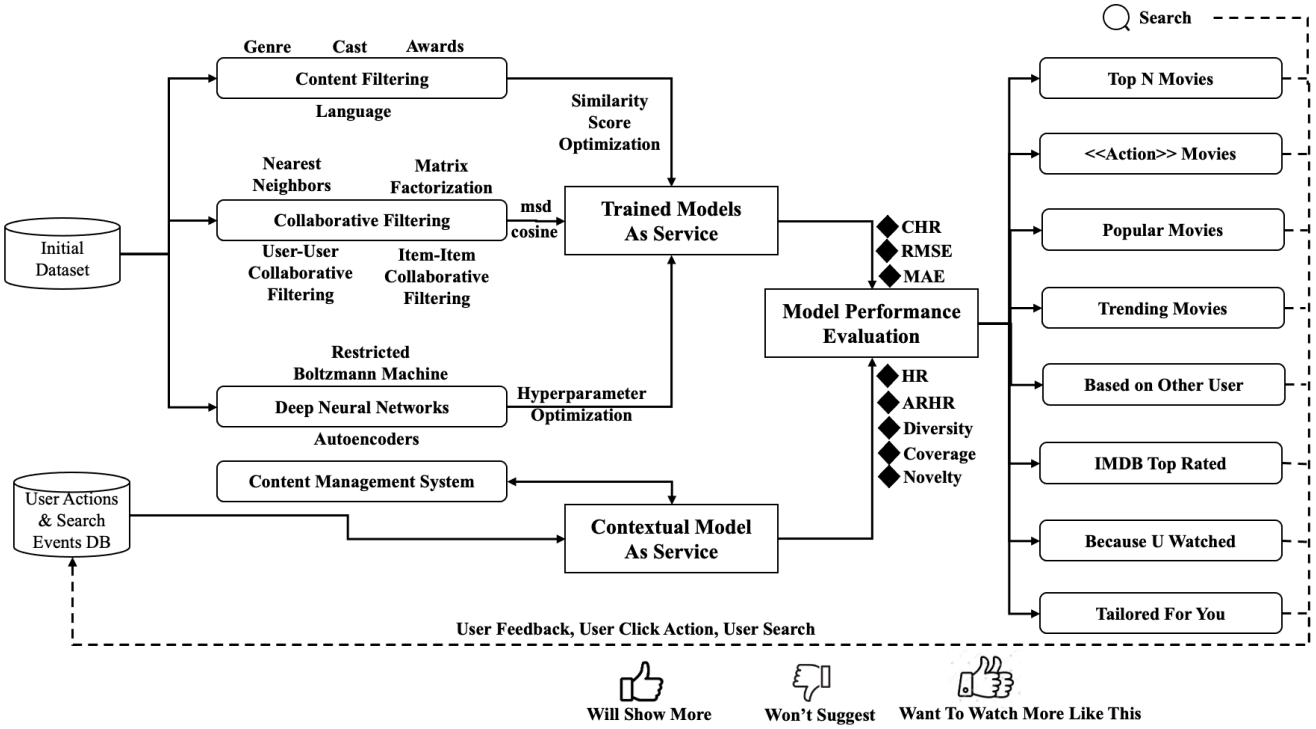

Fig 1. Proposed recommendation framework with contextual input and feedback loop

The framework proposed in Figure 1 follows a hybrid approach to fulfil the required tray listing shown on the user interface of the OTT platform as thumbnails. Not only is it able to recommend tray items, but it also keeps capturing user actions like clicking on a thumbnail, searching for content etc. These user events are periodically stored and processed against the content management system to recommend additional trays like "Tailored for You" to provide user-specific recommendations based on browsing history. The detailed evaluation and finalization of specific algorithms to fulfil the tray requirement is described in detail in the following sections.

2.1.1. Key Performance Measures

We will use a hybrid approach by combining multiple models to build a recommendation system, but how much it's relevant to user in real life scenario is a challenge. A recommendation system should be effective enough to engage users for longer duration with the platform. The traditional way of effectiveness of a model is judged by finding out mean absolute error (MAE) and root mean square error (RMSE) as computable from **Equations (1 and 2)** respectively. The lower these values its better, and the model is considered good. But the true effectiveness of recommendation system is dependent on other supporting metrics [9] as explained in **Equations (3 to 6)**.

$$\text{Mean Absolute Error (MAE)} = \frac{\sum_{i=1}^{n} | y_i - x_i |}{n} \tag{1}$$

$$\text{Root Mean Square Error (RMSE)} = \sqrt{\frac{\sum_{i=1}^{n} (y_i - x_i)^2}{n}} \tag{2}$$

$$\text{Hit Rate (HR)} = \frac{total\ no\ of\ hits\ for\ all\ users}{toal\ no\ of\ users\ in\ the\ test\ set} \tag{3}$$

$$\text{Average Reciprocal Hit Rate (ARHR)} = \frac{\sum_{i=1}^{n} \frac{1}{rank\ of\ first\ hit\ for\ user\ i}}{total\ no\ of\ users\ in\ the\ test\ set} \tag{4}$$

$$\text{Cumulative Hit Rate (cHR)} = \frac{no\ of\ users\ with\ at\ least\ one\ correct\ recommendation}{total\ no\ of\ users\ in\ test\ set} \tag{5}$$

$$\text{Rating Hit Rate (rHR)} = \frac{number\ of\ hits}{total\ no\ of\ users\ in\ test\ set} \tag{6}$$

HR measures the percentage of users for whom the recommendation system provides at least one relevant movie in the top N recommendations. ARHR determines the rank at which the first hit occurs for each user. Recommendation is considered good if the hit occurs at top slot rather than the bottom slot. cHR measures the proportion of users for whom the recommendation system has made at least one correct recommendation within a given list of recommendations based on some threshold value or rating. RHR measures effectiveness at suggesting movies that align with the user's tastes.

Coverage is a metric that measures the extent to which the system can provide recommendations for a diverse set of items in the catalogue. Catalogue Coverage (CC) and User Coverage (UC) as given in **Equations (7 and 8)** measures the proportion of unique items from the entire catalogue that recommender system can recommend and the proportion of users for whom the recommender system can provide personalized recommendations, respectively.

$$\text{Catalogue Coverage (CC)} = \frac{no\ of\ unique\ recommended\ items}{total\ no\ of\ unique\ items\ in\ the\ catalog} \tag{7}$$

$$\text{User Coverage (UC)} = \frac{number\ of\ users\ with\ recommendation}{total\ no\ of\ users\ in\ the\ system} \tag{8}$$

Diversity is a metric that covers how many possible variations the recommendation system provides. It's the opposite to similarity, and high diversity may not be treated as good Diversity $= 1 - S$, S = average similarity between recommended pairs as computed in **Equation (9)**. Novelty, as computed by **Equation (10)** determines the mean popularity rank of recommended items and, if recommended properly, can address the niche requirement of a user.

$$\text{Similarity (S)} = \frac{total\ similarity}{total\ no\ of\ items} \tag{9}$$

$$\text{Novelty} = \frac{total\ \ ranking}{total\ no\ of\ items} \tag{10}$$

The above metrics, as given in **Equations (1 to 10)**, can be used to evaluate top recommendations either individually or via a weighted model as per the needs and preferences of the OTT provider. In this paper, we have considered it individually to judge the effectiveness of the recommended system. Similarities used by Nearest Neighbours are computed by various distance measures as mentioned in **Equations (11 to 14)**.

$$\text{Cosine Similarity (x,y)} = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}} \tag{11}$$

$$\text{Adjusted Cosine Similarity (x,y)} = \frac{\sum_i ((x_i - \overline{x}\,)(y_i - \overline{y}))}{\sqrt{\sum_i (x_i - \overline{x})^2}\sqrt{\sum_i (y_i - \overline{y})^2}} \tag{12}$$

$$\text{Pearson Similarity (x,y)} = \frac{\sum_i ((x_i - \overline{i}\,)(y_i - \overline{i}))}{\sqrt{\sum_i (x_i - \overline{i}\,)^2}\sqrt{\sum_i (y_i - \overline{i}\,)^2}} \tag{13}$$

$$\text{Mean Square Diff. (MSD) (x,y)} = \frac{\sum_{i \in I_{xy}} (\,x_i - y_i\,)^2}{|I_{xy}|},\ \text{MSD Similarity (x,y)} = \frac{1}{MSD(x,y)+1} \tag{14}$$

2.1.2. Datasets Used

The experimentation involves the pre-processing of the sample dataset to ensure uniformity, making it compatible with the proposed framework, and accommodating diverse data sources. Notably, the MovieLens[10] and IMDB[11] datasets undergo a pre-processing stage to bring them into a common format. It's worth noting that the IMDB dataset lacks user information, rendering user-to-user collaborative filtering (CF) infeasible with this data source. Consequently, we adopt the MovieLens-1m dataset as our base for evaluating all models, ensuring a comprehensive assessment of their performance and capabilities.

2.1.3. Parameter Discussion

In content-based filtering, genres like animation, comedy, adventure, etc. are used to find similar movies. This is the simplest of all models, where based on available movie metadata and the user's previous watch history, top movies are recommended based on sorted values of movie ratings. CF uses the user and movie matrix using rating and movie datasets. In user-based CF, similarity between users is calculated using the nearest neighbour algorithm. A similar principle is also followed by item-based CF where similar movies are determined based on their ratings in place of users. In matrix factorisation techniques, sparse metrics are created, and the evaluation is done based on their accuracy scores. Various models like SVD and SVD++ are evaluated and compared with a custom implementation. As CF involves a lot of computation for calculating the similarity of each user or movie, CPU and memory overhead are calculated to give a proportional view of the computation and memory involved while running the models. Standard deep learning recommendation models like RBM and Auto-encoders are also evaluated by creating training sets, test sets and anti-train sets. The model is also evaluated by following the leave-one-out principle to check if the recommendation system can predict the known left item as part of its recommendation. While training the model using custom code, the latent dimension values of 20 is considered as per standard practice. A standard epoch of 20 is chosen, which can provide the minimum loss function values as shown in the subsequent graph. In auto-adjusted SVD, auto-tuning algorithms are used to find out the maximum accuracy with a learning rate between 0.005 and 0.01, epoch between 20 and 30 and number of latent features between 50 and 100. A threshold value of 4.0 is used in content and CF to reduce unnecessary computation and provide recommendations with better movie ratings.

2.2 Analyzing OTT Platform

One of the major ways to keep users engaged to the platform is by recommending other content available on the platform. OTT vendors like Netflix, Amazon Prime, and YouTube spend huge money to build recommendation platforms using both traditional machine learning and deep learning models. These types of systems are too costly to build and need a lot of experience and effort, which is very difficult to achieve for a new entrant. Along with this, whether the recommendation built is achieving its objective and does it really provide good recommendation is still a challenge, and there is no best way to confirm the same.

2.2.1. OTT Feature Tray Comparison

Prior to assessing various recommendation techniques, it's essential to gain insight into the diverse recommendation strategies employed by leading OTT channels in India, such as Netflix, Amazon Prime, HotStar, and SonyLiv. Our analysis, as presented in Table 1, reveals that most of these OTT platforms incorporate a hybrid approach that combines content-based and CF techniques, with top-n recommender systems being prevalent across the board. The recommendations are generated either specific to Genre or because you have watched specific content or someone like you have watched. Final proposed model will be compared at last to check the feasibility of achieving all these features or trays.

Table 1. Top Indian OTT platforms real recommendation types for TV

OTT Provider	Trays
Netflix	Top 10 Movies in India Today, Exciting Movies, Continue Watching, Trending Now, New Releases, <<Genre>> Movies, Top Searches, only on Netflix, Because You Liked X, <<Language>> Movies, <<Age>> Movies, Popular in Netflix, Because you watched X, Watch Again
Amazon Prime Video	Prime Top 10 in India, Prime Recommended Movies, Continue Watching, Only Spotlight, Prime Recently Added Movies, Prime <<Genres>>, Prime Top-rated TV Shows on IMDB, Prime Watch in Your Language, Prime <<Age>> Movies, Discovery+: Most Popular, Prime Because you watched X, Watch It Again
JioHotstar	Top 10 in India Today, Latest Releases, Popular Movies, Watchlist, Non-Stop Sports, Top Rated in IMDB, Exclusive Indian Movies, Global English Hits, Popular in <<Genre>>
SonyLiv	Hot Picks, Top Rated Originals on IMDB, Popular on Liv, New on Liv, Popular Hindi Movies, South Dubbed Masala, Continue Watching

2.2.2. User Events and Contextual Search

Careful analysis of various trays leads to recommendation models based on content-based filtering and collaborative filtering, which is explained along with results in section 3. However, this doesn't qualify the models as the best as contextual information is missing and the recommendations are quite generic. To improve user experience, recommendations should be provided based on user action. This is collected when the user clicks on the thumbnail to watch a movie, and after watching for a specific time duration. User feedback is also collected after the content is watched by accepting recommendation feedback as a like or a dislike. These events can then be stored in the DB and mapped to user profiles. Based on watched content metadata, similar content can be recommended to the user.

To extend further, a novel approach is proposed to use search as part of the recommendation. Instead of clicking the recommended tray thumbnails, the user may search for a particular scene to find out about a movie. Search engines need to be contextualised on content metadata to recommend a tray of similar search items. This provides a better recommendation close to the user choice and caters to the changing preferences of users.

3 Results and Discussion

Core technologies like content-based filtering, collaborative filtering, along with other deep learning models like SVD, RBM, are explained and implemented [12] using movielens-1m datasets and showcase the RMSE and MAE values achieved after due optimisation. A straightforward comparison has been presented with the best models observed from available models, and subsequently showcases the improved RMSE and MAE scores achieved.

3.1 Content-based Filtering Framework

Content-based filtering is one of the simplest and straightforward techniques to provide recommendations based on content metadata like genre, cast, director, language, etc. Content metadata is stored in the content management system or associated

storage during content upload. Content can be filtered based on a specific content attribute like "Comedy" and shown in the "Comedy Movies" Tray. In all the implementations Top 10 recommendations are chosen as a best practice followed by OTT market leaders, to keep the application light and useful. The output of content-based filtering, as shown in Figure 2, is compared with a standard random recommendation model as provided by the Surprise library [13]. The metrics of Content-based filtering are presented in Table 2.

Table 2. Content-based filtering metric

Model	MAE	RMSE	Evaluation Time in Sec	CPU Overhead	Memory Overhead
Random	1.2058	1.5044	4.08499	3.80%	0.20%
Content-KNN-Cosine	0.8752	1.109	4561.64	5%	0.10%

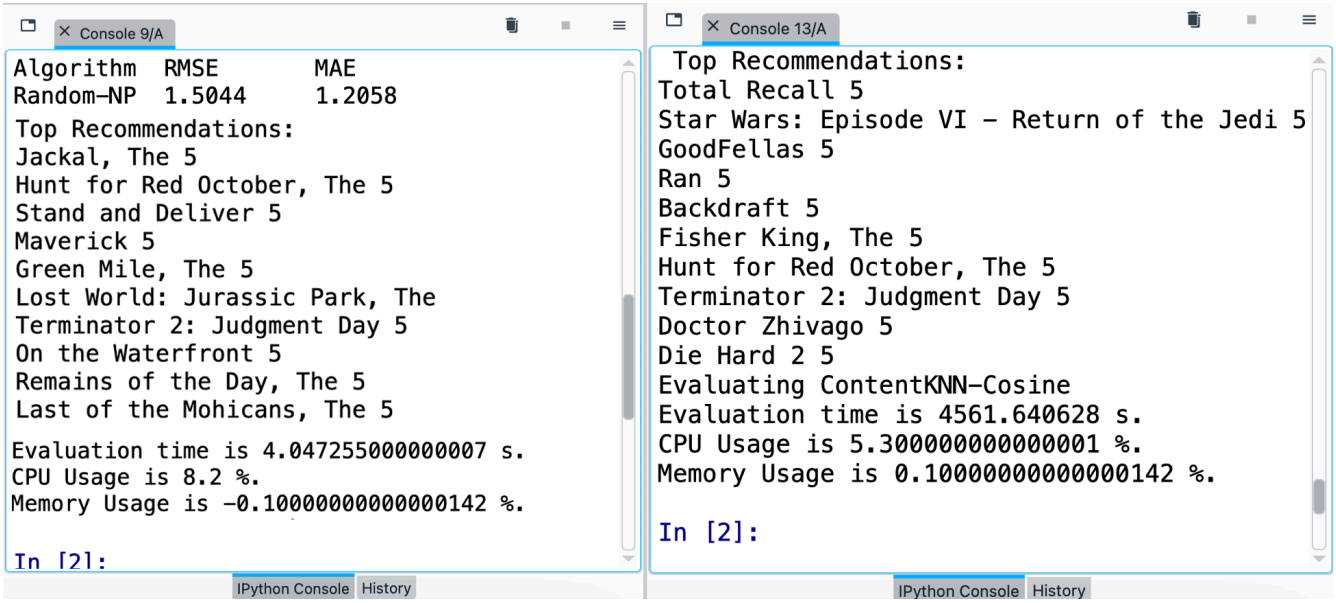

Fig 2. Content-based filtering Top recommendations

3.2 Collaborative Filtering

Collaborative filtering is a robust contender for a recommendation system that relies on behavioural data collected from other users who have watched the content. Data sparsity is a bigger challenge in this, which can be handled by using sparse arrays to save substantial computation overhead. Both User-based CF and Item-Based CF are evaluated, and results are shown in subsequent subsections.

3.2.1. User-based Collaborative Filtering Framework

User-based CF focuses on similar users or users with similar tastes and preferences, to recommend content for another user. Similarity is the measure driver for collaborative filtering, and it's observed that adjusted cosine similarity is more effective compared to cosine similarity as defined in **Equations 11 and 12**. It gauges similarity in relation to the mean of other items or users, mitigating issues coming from potential user behaviour biases. Pearson similarity, as defined in **Equation 13** considers the total mean rating of an item by all users, making it suitable for new or unhandled items. MSD as defined in **Equation 14** operates like Mean Square Error (MSE), assessing how two users x and y rate the same item i. Similarity may be established between users who have both rated a specific movie with the same score, potentially leading to misleading recommendations.

To mitigate this issue, it is advisable to introduce a threshold, such as a minimum number of times a particular movie must be rated. In essence, the user-based CF process involves generating user-item ratings and user-user similarity matrices, followed by candidate generation and scoring phases. The output then undergoes filtering based on predefined thresholds or conditions, such as not having been watched, to yield the final top recommendations, as demonstrated in Figure 3.

3.2.2. Item-based Collaborative Filtering Framework

Item-based CF works similarly like User based CF with a difference on item-to-item comparison in place of user-to-user comparison. This becomes effective when user preferences and taste change over time, but the item or content remains constant. This is also suitable for scenarios where the number of contents is smaller compared to number of users. The decision is dependent on how much a user i likes item j' compared to how much everyone else likes j'. If user i likes a movie j' and j is similar to j', then it's imperative that user i will like j. Here, the weight for movies j and j' will be higher and they have a high similarity score as given in **Equation (15)**.

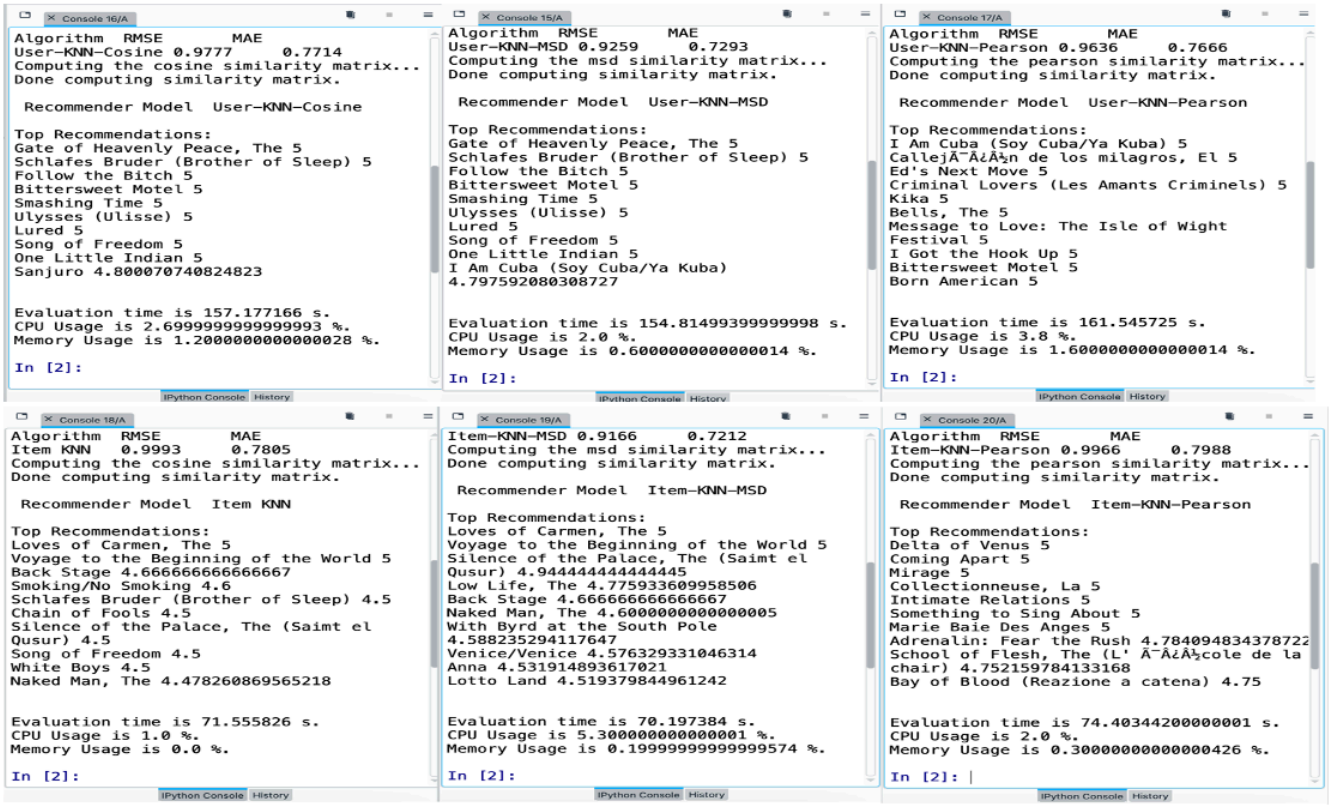

Fig 3. User-based CF and Item-based CF recommendations

$$S(i,j) = \overline{r_j} + \frac{\sum_{j' \in \Omega_i} w_{jj'} \{r'_{ij} - \overline{r_{j'}}\}}{\sum_{j' \in \Omega_i} |w'_{jj}|} \tag{15}$$

3.2.3. User-Based CF and Item-Based CF Result Comparison

User-based CF and Item-Based CF are two effective models that work on similarity calculation and provide multiple recommendation trays. Table 3 Provides a detailed comparative analysis of various metrics. Item-KNN-MSD provides the best accuracy with MAE and RMSE scores of 0.7212 and 0.9166 respectively. Although there is a slight computational overhead as compared to others, it doesn't have much significance as once the models are trained, subsequent computation overhead goes down. Compared to all models, K-Nearest Neighbour (KNN)[14]-[15] based Item-KNN-MSD seems to provide precise recommendation among all collaborative models.

3.3 Matrix Factorization Based Recommendation Framework

Matrix Factorization[14] is one of the advanced machine learning techniques that can analyze user behaviours and preferences to deliver tailored content suggestions. Netflix has employed SVD to predict movie ratings. Table 4. Briefly provides comparative

Table 3. User-based and item-based CF system metrics

Model	MAE	RMSE	Evaluation Time(s)	CPU Overhead	Memory Overhead
Random	1.2058	1.5044	4.048	3.80%	0.20%
Custom-UserCF-NoML	0.8504	1.179453	7838.72	3%	1.90%
User-KNN-Cosine	0.7714	0.9777	157.177	2.70%	1.20%
User-KNN-MSD	0.7293	0.9259	154.815	2.00%	0.60%
User-KNN-Pearson	0.7666	0.9636	161.546	3.80%	1.60%
Custom-ItemCF-NoML	0.6374	0.8309	3102.96	5.50%	0.10%
Item-KNN-Cosine	0.7805	0.9993	71.5558	1.00%	0.00%
Item-KNN-MSD	**0.7212**	**0.9166**	**70.1974**	**5.30%**	**0.20%**
Item-KNN-Pearson	0.7988	0.9966	74.4034	2.00%	0.30%

metrics across various recommendation models like SVD, SVD++ and Auto-Adjusted SVD against a plain custom matrix factorization model using Python and NumPy libraries.

As per Table 4, SVD++ outperforms the other models with the lowest MAE (0.6735) and RMSE (0.8645), suggesting it provides the most accurate recommendations. SVD closely follows with slightly higher MAE (0.689) and RMSE (0.8782), reaffirming its commendable accuracy. The Auto-adjusted SVD, which is based on tuning on epoch and features, also performs well, with MAE (0.6855) and RMSE (0.8733) scores on par with SVD. Moving to computational efficiency, the evaluation time metric reflects the time required for model computations. Here, SVD++ stands out but at a considerable computational cost, as it takes significantly more time (425.16) compared to SVD (12.59) and Auto-Adjusted SVD (11.58). This highlights the trade-off between accuracy and computation time, with SVD++ excelling in precision but demanding more resources. CPU and memory overhead metrics reveal resource consumption. SVD++ incurs a relatively high CPU overhead (5.30%) compared to SVD (5.00%) and Auto-adjusted SVD (1.50%). In terms of memory usage, all models perform well due to the basic principle of matrix factorization, which reduces overhead storage. SVD offers balanced performance, combining decent accuracy with lower resource utilisation. Auto-Adjusted SVD, while on par with SVD in accuracy, boasts minimal CPU overhead and memory usage, making it a compelling choice for scenarios where computational efficiency is a critical consideration without compromising recommendation quality.

3.3.1. Deep Learning Based Recommendation Framework

Deep learning is the most advanced technique that explores the power of deep neural networks and advanced optimization technique to provide personalised recommendations. Advanced techniques like stochastic gradient descent, activation functions and optimization algorithms can learn complex patterns from large datasets efficiently. Optimization techniques like RMSProp, Adam complement gradient descent, and overfitting is avoided by early stopping, dropout and regularization techniques. Deep learning architecture, like RBM [7] and Auto-encoders [8] can extract meaningful information from user-item interaction. Top recommendations using RBM and Auto-encoders are given on Figure 4. It also shows time-based contextual tagging generated from a sample video, which can be stored in a repository and can provide contextual output on search.

Table 4 depicts in detail the performance metrics of two distinct deep learning models, RBM and Auto-encoders. RBM model yields a better result with the lowest MAE and RMSE score with little overhead of processing time.

Table 4. SVD, SVD++ and Auto-Adjusted SVD, RBM and Auto-encoder metrics

Model	MAE	RMSE	Evaluation Time(s)	CPU Overhead	Memory Overhead
Custom-MF	0.5755	0.7374	1074.54	0.00%	0.00%
SVD	0.689	0.8782	12.5897	5.00%	0.00%
SVD++	**0.6735**	**0.8645**	**425.153**	**5.30%**	**0%**
Auto-Adjusted SVD	0.6855	0.8733	11.5787	1.50%	0%
Random	1.2058	1.5044	4.048	3.80%	0.20%
RBM	**1.0521**	**1.2579**	**2985.5**	**9%**	**0%**
Autoencoders	1.4205	1.7833	10130.9	0%	4%

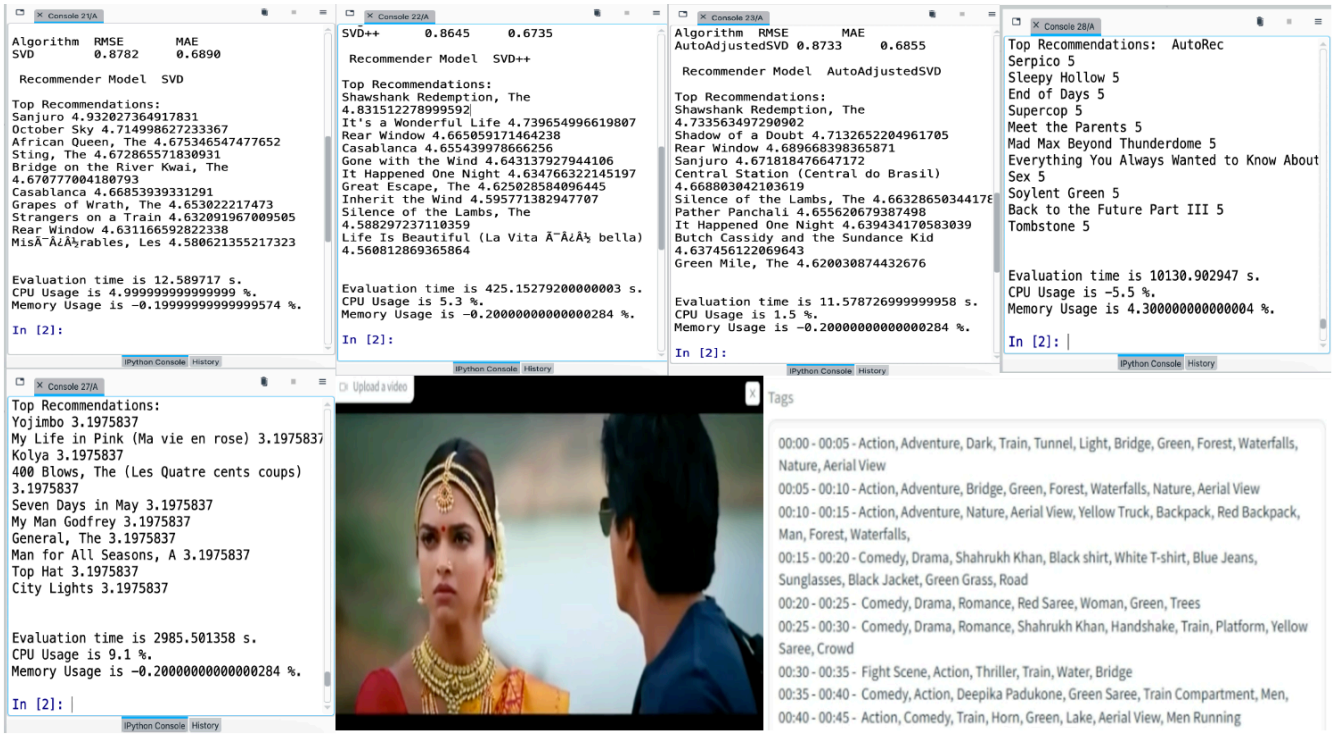

Fig 4. SVD, SVD++, Auto-Adjusted SVD, RBM, Auto-encoder and Contextual Recommendations Tagging

3.4 Result Summarization

Based on the comprehensive evaluation conducted above, we can summarize to have a hybrid model to meet the different tray requirements of OTT platform. Content-based filtering using KNN has high accuracy and excels in providing metadata-based recommendations like genre, cast, crew etc. Item-based collaborative filtering can generate recommendations based on how other users has viewed the specific movie and rated, like tray "As you watched movie X". Among matrix factorization SVD++ stands out as a top performer with great accuracy. In the deep learning space RBM emerges as a promising model. So, following a hybrid multi-modal approach and adapting top models, we can provide a much better recommendation for users. New OTT entrants can easily adapt and build their recommendation engine using available open-source libraries as given in Table 1. A new contextual tray can also be provided based on user events and user search patterns and recommend top contents from the content management system.

As we already highlighted in section 2.1.1, a model with lowest error rate measured by MAE and RMSE values doesn't make it a candidate for good recommendation in real life. Users urge them to watch new and popular content, irrespective of whether they have watched similar movies before or not. Also, with user preferences changing over time, the traditional models may not give the best user experience and hence need to include factors like diversity while providing recommendations. Irrespective of which model is used, the true effectiveness of a recommendation system is based on real user feedback measured by A/B testing in a real environment. Table 5 provides a comparative analysis of all shortlisted models based on all key performance indicators and leaves it to the specific platform to consider the weightage of each attribute. Once these models are baselined, it can further be monetized by providing service-based access to other consumers. Other consumers can consume the base model and further tune it with their organisation-specific user data to build an effective and efficient recommendation system.

The results are shown in Table 5 shows the summarised view of all shortlisted models along with improved MAE and RMSE results. In [16], KNN and SVD on the movielens-1m database give MAE values of 0.8301 and 0.8501, respectively. RMSE value, which is the square root of MSE, is 1.0458 and 1.0954 for KNN and SVD, respectively. Similarly, in [17], SVD has MAE score of 0.7194. By applying SVD++, we can achieve a much better MAE score of 0.6735 and RMSE score of 0.8645 compared to SVD. Similarly, in [18] UserKNN MAE value of 0.81 is higher than the achieved value of 0.7293. The RMSE score of 0.95 is also lower compared to the achieved value of 0.9259.

Table 5. Comparative analysis of all shortlisted models

Models	MAE	RMSE	HR	cHR	ARHR	Coverage	Diversity	Novelty	Evaluation Time(s)
KNN [16]	0.8301	1.0458	-	-	-	-	-	-	-
SVD [16]	0.8501	1.0954	-	-	-	-	-	-	-
SVD [17]	0.7164	-							
ItemKNN [18]	-	0.8930	-	-	-	-	-	-	-
UserKNN [18]	0.81	0.95	-	-	-	-	-	-	-
User-KNN-MSD	0.7293	0.9259	-	-	-	-	-	-	154.815
Content-KNN-Cos	0.8752	1.109	0.0137	0.014	0.0048	0.9932	0.0508	669.1566	4610
Item-KNN-MSD	0.7212	0.9166	0.0068	0.007	0.0017	0.9912	0.8578	3080.5444	5793.24
SVD++	0.6735	0.8645	0.0341	0.034	0.0123	0.9868	0.0336	657.051	9573.95
RBM	1.0522	1.2579	0.0023	0.002	0.0004	0	0.0244	1395.2067	15002.18

4 Conclusion

This study's experiments provide insights into open source-based recommendation strategies which can be leveraged by any Streaming or OTT service provider and follow the path of leading OTT platforms available in India, like Netflix, Amazon Prime, HotStar, and SonyLiv. These platforms primarily employ hybrid approaches combining content-based and CF techniques, along with deep learning techniques to provide multiple recommendation trays to improve user experience. While accuracy of the model is one of the criteria but not sufficient to generate recommendations, and hence A/B testing is to be carried out to capture real user feedback and the models are retrained. User behaviour and psychological parameters are considered while providing recommendations. Providing a mix of popular movies along with new movies, providing a diverse set of movies in place of limiting to a single genre etc. gives the user confidence to explore new ones. A service-based recommendation model not only provides flexibility and extensibility on building a recommendation system but also provides the option to monetize and standardize the recommendation system. Any new entrant in the OTT space can extend these minimum set of models to build its own recommendation system or use the existing services and save initial capital expenditure.

This study also recommends a novel way of contextual tray which is based on user search patterns. Existing contents or new contents can be processed to generate keywords or tags, which can be stored in the repository. User in place of searching movie name, can search for a scene as normal text string. The text string can then be matched against the repository and detailed content is given as a response. This search event can be contextualized to recommend similar scenes as part of the contextual recommendation tray. This is not implemented in the current study and kept for future enhancements, with sample output shown in Figure 4. User event data generated on a real-time basis and going as a feedback loop to the existing base model is also an area of exploration. Processing millions of real-time events and generating top recommendations using this is also an area to be looked upon.

Looking to the future, there are several avenues for further enhancement in this domain. These may include exploring methodologies and other deep learning models to further optimize model performance using concepts like parallel processing, multi-threading support, clustering techniques, and event-based architecture. Efficient processing of available datasets and explicit data collected from user events based on user activities is also an area to be explored. While all these services can run in isolation on top of available open-source libraries or custom algorithms, monitoring the system in place is also an area to be explored further. Additionally, there is room for standardizing recommendation services in the industry, allowing new entrants to leverage existing models and services to minimise initial capital expenditure. This manuscript serves as a valuable reference for those seeking to understand and improve recommendation systems in the ever-evolving OTT landscape.

References

1) Statista. OTT video - India. [Internet]. [cited 2025 May 9]. 2025. Available from: https://www.statista.com/outlook/amo/media/tv-video/ott-video/india#revenue.
2) Kannikaklang N, Wongthanavasu S, Thamviset W. A hybrid recommender system for improving rating prediction of movie recommendation. *2022 19th International Joint Conference on Computer Science and Software Engineering (JCSSE)*. 2022;p. 1–6. https://doi.org/10.1109/JCSSE54890.2022.9836257.
3) Gupta A, Shrinath P. Link Prediction based on bipartite graph for recommendation system using optimized SVD++. *Procedia Computer Science*. 2023;218:1353–1365. https://doi.org/10.1016/j.procs.2023.01.114.

4) Do PM, Nguyen TT. Semantic-enhanced neural collaborative filtering models in recommender systems. *Knowledge-Based Systems*. 2022;257:109934. https://doi.org/10.1016/j.knosys.2022.109934.
5) Sahu S, Kumar R, Pathan MS, Shafi J, Kumar Y, Ijaz MF. Movie popularity and target audience prediction using the content-based recommender system. *IEEE Access*. 2022;10:42044–42060. https://doi.org/10.1109/ACCESS.2022.3168161.
6) Pujahari A, Sisodia DS. Modeling side information in preference relation based restricted Boltzmann machine for recommender systems. *Information Sciences*. 2019;490:126–145. https://doi.org/10.1016/j.ins.2019.03.064.
7) Al-Sbou AM, Rahim NHA. An improved hybrid semi-stacked autoencoder for item-features of recommendation system (iHSARS). *Indonesian Journal of Electrical Engineering and Computer Science*. 2023;30(1):481–490. https://doi.org/10.11591/ijeecs.v30.i1.
8) Berahmand K, Daneshfar F, Salehi ES, Li Y, Xu Y. Autoencoders and their applications in machine learning: a survey. *Artificial Intelligence Review*. 2024;57(2):28. https://doi.org/10.1007/s10462-023-10662-6.
9) Alhijawi B, Fraihat S, Awajan A. Multi-factor ranking method for trading-off accuracy, diversity, novelty, and coverage of recommender systems. *International Journal of Information Technology*. 2023;15(3):1427–1433. https://doi.org/10.1007/s41870-023-01158-1.
10) GroupLens. MovieLens dataset [Internet]. [cited 2025 May 9]. 2025. Available from: https://grouplens.org/datasets/movielens/.
11) IMDb. IMDb Non-Commercial Datasets [Internet]. 2025 May 2 [cited 2025 May 9]. 2025. Available from: https://developer.imdb.com/non-commercial-datasets/.
12) Hug N. Surprise: A Python library for recommender systems. *Journal of Open Source Software*. 2020;5(52):2174. https://doi.org/10.21105/joss.02174.
13) Delimayanti MK, Laya M, Warsuta B, Faydhurrahman MB, Khairuddin MA, Ghoyati H, et al. Web-Based Movie Recommendation System using Content-Based Filtering and KNN Algorithm. *2022 9th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*. 2022;p. 314–318. https://doi.org/10.1109/ICITACEE55701.2022.9923974.
14) Fiagbe R. Composite Movie Recommendation System, Movie Recommender System Using Matrix Factorization. *Data Science and Data Mining*. 2025. Available from: https://stars.library.ucf.edu/data-science-mining/4.
15) Li N, Xia Y. Movie recommendation based on ALS collaborative filtering recommendation algorithm with deep learning model. *Entertainment Computing*. 2024;51. https://doi.org/10.1016/j.entcom.2024.100715.
16) Alsekait DM, Shdefat AY, Mostafa N. Next-Generation Movie Recommenders: Leveraging Hybrid Deep Learning for enhanced personalization. *Applied Mathematics & Information Sciences*. 2024. http://dx.doi.org/10.18576/amis/180504.
17) Alieva OA, Gangan ES, Ilyushin EA, Kachalin AI. Automatic Evaluation of Recommendation Models. *Modern Information Technologies and IT-education*. 2020;16(2):398–406. https://doi.org/10.25559/SITITO.16.202002.398-406.
18) Alam MM, Ahmed M. Deep Learning Based Collaborative Filtering Recommendation System. *Procedia Computer Science*. 2025;258:2362–2371. https://doi.org/10.1016/j.procs.2025.04.499.