International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Creating A Weighted Hybridization Approach for A Music Recommendation System to Tackle Significant Challenges Inherent in Recommendation Systems

Dr. M. Sunitha [1], Dr. T. Adilakshmi[2], Dr. Marepalli Radha[3], G. Venkat Rama Reddy[4], Dr. M. Sandhya Rani[5]

Submitted: 12/12/2023 **Revised**: 16/01/2024 **Accepted**: 01/02/2024

Abstract: The Music Recommendation System (MRS) functions as a remedy to the information overload prevalent in the digital music landscape. This research paper addresses prominent challenges encountered by recommendation systems, specifically the Long Tail phenomenon, data sparsity, and the cold-start problem, through the implementation of a weighted hybrid approach. This approach integrates collaborative filtering techniques based on both user preferences and item characteristics. Notably, our proposed system incorporates contextual information in the generation of music recommendations. Experiments were conducted using a benchmark dataset and synthetic data derived from a Music Portal application. The results demonstrate the system's efficacy in accurately capturing user interests, considering diverse factors such as a user's historical preferences, profile, item similarities, timestamps, and social connections.

Keywords: *Music Recommendation system, Information Overloading, Long tail, Sparsity, Cold- start, Weighted hybridization*

1. Introduction

In the contemporary era dominated by the Internet and smartphones, individuals are confronted with a myriad of choices, spanning decisions like selecting a digital music item, choosing a restaurant, picking a cinema, purchasing a book, or various accessories. As users navigate these choices, it is natural for them to develop preferences, encompassing both their likes and dislikes. On closer examination of an individual's preferences, an underlying pattern often emerges [7].

The primary goal of a Recommendation System (RS) is to unveil this inherent pattern that defines a user's taste. Once these preferences are understood, an RS can suggest similar items. For instance, if someone enjoys spinach sandwiches, it is highly likely that they would also appreciate a corn sandwich, as the two share significant similarities, differing only in the substitution of spinach with corn.

Music, being ubiquitous, inundates listeners with an overwhelming number of choices available on digital platforms. Users are consistently seeking music that resonates with their personal taste, driving the need for effective music recommendations. Despite the emergence of services like Gaana, Spotify, and Last.fm [1] aiming to provide a perfect solution, complete success has yet to be achieved. Music preferences are shaped by factors like taste, trust, and affinity for specific artists, which pose challenges in quantifying for machines or software. Consequently, service providers struggle to accurately identify music that genuinely appeals to and satisfies an individual's taste [8].

Every music recommendation system relies on a set of assumptions to deliver effective recommendations. The fundamental types of recommendation systems in the literature are outlined below.

Collaborative Filtering: This method utilizes user interactions and feedback to generate recommendations, assuming that users who have interacted with and liked similar music in the past will have similar preferences in the future [5][6].

User-Song Matrix: In collaborative filtering, a user-item matrix is established, where rows represent users, columns represent items (songs), and each cell contains a rating or interaction score.

User-Based Collaborative Filtering: This approach identifies users with similar music preferences to the target user and recommends songs liked by those similar users but not yet rated by the target user.

Item-Based Collaborative Filtering: This method identifies songs like those the target user has liked in the past and recommends those similar songs.

Content-Based Filtering: This technique focuses on the characteristics and features of songs, as well as user profiles,

[1,2]*Vasavi College of Engineering, Hyderabad-31, India*
[3]*CVR COLLEGE OF ENGINEERING, Hyderabad, India, orcid 0000-0003-8698-5933*
[4]*Senior Technical Business Analyst, AT&T Communication Services India Pvt Ltd.*
[5]*Bhoj Reddy Engineering College for Women, Hyderabad, India, ORCID id : 0009- 0007-1474-988X*

to make recommendations. It assumes that users will prefer songs with attributes like those they have liked previously.

Song Features: Content-based filtering considers various features of songs, such as genre, artist, tempo, lyrics, and musical key. Songs can be classified [19][20] into different categories based on th features.

User Profile: A user profile is created based on the user's historical interactions, including liked songs and the attributes of those songs they have engaged with. Users can be clustered by using K-means [18] clustering to group them based on the profile information.

Recommendation: Content-based systems recommend songs that match the attributes and features of the songs the user has shown interest in, aligning with their preferences.

The remaining sections of the paper follow this structure: Section 2 delves into previous research in this domain, Section 3 outlines the suggested approach, Section 4 showcases the outcomes attained using this method, and finally, Section 5 summarizes the conclusion and potential directions for future research.

2. Related Work

In the dynamic landscape of music recommendation systems, considerable research has been devoted to overcoming critical challenges, particularly the cold-start problem, data sparsity, and the long tail phenomenon. This section reviews seminal works that have made significant contributions to advancing our understanding and addressing these challenges.

Hybrid Approaches for Cold-Start Users: Feng et al. [11] proposed an innovative hybrid recommendation system that integrates collaborative filtering with content-based filtering. By leveraging user demographic information and music content features, this approach effectively addresses the cold-start problem, providing accurate recommendations for users with limited interaction history.

Matrix Factorization Techniques for Data Sparsity: Jones and Wang [12] introduced matrix factorization techniques to tackle data sparsity in music recommendation systems. By decomposing the user-item interaction matrix, their approach enhances recommendation accuracy, particularly for users with sparse interaction histories, contributing to a more robust system.

Deep Learning for Long Tail Recommendations: Building upon the work of Chen and Li [13], deep learning models have emerged as powerful tools for addressing the long tail phenomenon. These architectures, designed to capture intricate patterns in user preferences, significantly improve the effectiveness of recommendations for both popular and niche music items.

Context-Aware Systems for Cold-Start Alleviation: Kim and Park [14] laid the groundwork for context-aware recommendation systems to alleviate the cold-start problem. By incorporating user contextual information such as location, time, and mood, these systems enhance their ability to provide relevant recommendations for users with limited interaction history.

Sparse Factorization Machines for Sparsity and Long Tail: The work of Zhang et al. [15] introduced sparse factorization machines to address both data sparsity and the long tail issue. This model efficiently manages sparse user-item interaction data and captures latent factors, contributing to accurate recommendations across a diverse array of music items.

Enhanced Collaborative Filtering for Long Tail: Wang and Liu [16] proposed enhancements to collaborative filtering techniques, incorporating item features and adjusting similarity metrics. These modifications aim to improve the recommendation quality for less popular or newly introduced music items, effectively tackling challenges associated with the long tail.

Transfer Learning for Cold-Start and Sparsity Mitigation: Li et al. [17] introduced transfer learning techniques to address the cold-start problem and data sparsity. Leveraging knowledge gained from a source domain with richer data, their model enhances recommendations for a target domain with limited user interactions, demonstrating the potential of transfer learning in mitigating critical issues.

3. Proposed Work

This section delineates the methodologies integrated into the proposed system. The initial phase of any data mining undertaking involves data pre-processing. In our proposed approach, we utilize user logs sourced from Last.fm [1] for experimental purposes. Last.fm systematically records the musical preferences of registered users, building comprehensive profiles. These profiles are constituted by two datasets. The first dataset encapsulates information about each user's listening preferences in the form of user logs. Each user log is comprised of attributes such as UserID, timeStamp, albumID, albumName, trackID, and trackName. The second dataset contains profile details of all users, including attributes like UserID, Gender, Age, Country, and Registration date. Leveraging this information, we construct a User-Song matrix, and Table 1 serves as an illustrative example of a User-Song matrix, depicting user ratings for various items.

Table 1. User-Song matrix

	S1	S2	….	S14458
U1	7	0	….	0
U2	2	0	….	9

….	….	….	….	….
U200	0	6	….	0

3.1 Data Pre-processing

The data collected from Last.fm exhibits missing values and noise. Prior to applying recommendation algorithms, it is imperative to preprocess the data. Handling missing values involves filling in the song names using the corresponding track ID, while records containing noisy values are removed.

Following data cleaning, the subsequent step in the recommendation process involves identifying task-relevant data. Two constraints are enforced to obtain such data:

a. Consideration of Users with More Than 200 Songs: Only users who have listened to more than 200 songs are considered in the dataset.

b. Consideration of Songs Listened to by at Least 10 Users: Only songs that have been listened to by at least 10 users are included in the dataset.

The dataset sourced from Last.fm originally comprises listening histories of 992 unique users, resulting in a total of approximately 20 million records. Due to computational feasibility concerns with collaborative filtering methods, the research work focuses on the listening history of 200 users, encompassing around 3 million records. Following the application of pre-processing conditions, the number of songs filtered is 14,458.

3.2 Representation of User-Song matrix

User-Song matrix with Frequency vectors

A User–Song (U-S) matrix incorporating frequency vectors is created, with the frequency of each user for each song represented as a value in the corresponding cell of the matrix, as depicted in Table 1.

Binary User-Song matrix

A user-song matrix in frequency representation is converted to binary notation. Replace non-zero frequency count with 1. The binarized U-S matrix is shown in table 2.

Table 2. U-S matrix after binarization

	S1	S2	….	S14458
U1	1	0	….	0
U2	1	0	….	1
….	….	….	….	….
U200	0	1	….	0

Length Normalization and Root Mean Square Normalization:

Normalization is applied to the binary user-song matrix in the proposed system, employing both length normalization and RMSN normalization. U-S matrix after normalization is shown in the tables 3 and 4.

Table 3. U-S matrix after length normalization

	S1	S2	….	S14458
U1	0.143	0	….	0
U2	0	0.25	….	0.062
….	….	….	….	….
U200	0	0.053	….	0

Table 4. U-S matrix after RMSN normalization

	S1	S2	….	S14458
U1	1.291	0	….	0
U2	0	2.449	….	0.612
….	….	….	….	….
U200	0	0.513	….	0

3.3. User and Item Centric Collaborative Filtering

This technique implements user-centric and item-centric model collaborative filtering to form the

clusters of similar users and items. Once the model of user and item clusters is built, can be used to find the user cluster and item cluster most similar to the target user. This is shown in the pseudocode given in the Fig.1 & Fig. 2. It considers the user past history while forming similar user clusters and item similarities while forming item clusters.

Algorithm Threshold_UC()

1. Begin
2. Initialize the threshold value to th_cutoff
3. Create an empty list of clusters
4. For ui in u1, u2 … un
5. Create a new cluster C
6. Assign u to C
7. For each ui in u2 … un
8. begin

9. Find the similarity of ui with C
10. If sim(ui, C) ≤ th_cutoff
11. Assign ui to C
12. Else
13. Create a new cluster C'
14. Assign ui to C'
15. end
16. end
17. Add C to the list of clusters
18. Return the list of clusters
19. End

Fig. 1. Pseudocode for user-centric model

Algorithm Threshold_IC()

1. Begin
2. Initialize the threshold value to th_cutoff
3. Create an empty list of clusters
4. For each ui in u1, u2 … un
5. Create a new cluster C
6. Assign u to C
7. For each ui in u2 … un
8. begin
9. Find the similarity of ui with C
10. If sim(ui, C) ≤ th_cutoff
11. Assign ui to C
12. Else
13. Create a new cluster C'
14. Assign ui to C
15. end
16. end
17. Add C to the list of clusters
18. Return the list of clusters
19. End

Fig. 2. Pseudocode for item-centric model

3.4 Matrix factorization to address Sparsity

Sparsity represents a significant challenge in recommendation systems. This issue arises because users typically rate only a limited number of items. The sparsity problem is illustrated by the Sparse User-Item Matrix displayed in Figure 3.3.1, where the matrix reveals the ratings provided by m users for n items. Matrix factorization [4] serves as a method for reducing the dimensionality of a high-dimensional sparse matrix into a lower- dimensional dense matrix. SVD is a matrix decomposition technique that takes an mxn matrix and decomposes it into three matrices with dimensions mxk, kxk, and kxn, as expressed by the equation provided below.

$$A_{mxn} = U_{mxk} X\ S_{kxk} X\ V_{kxn}$$

Where U designates about users, V indicates about items in transpose and S shows identity matrix with Eigen values as diagonal elements.

User clusters are generated from the factored user matrix U and item matrix V is used to form item clusters. Euclidean distance is used as the proximity measure to form model of users and items by using the algorithm shown in fig.3 and fig.4.

Algorithm UCSSVD()

Input: SVD applied User-Song Matrix

Output: User Clusters

Method:

1. Initialize the threshold_cutoff value
2. Create an empty list of clusters
3. Consider the first user vector U1
4. Create a new cluster C1 and assign U1 to it
5. For each remaining user vector Ui (where i ranges from 2 to k, and k represents the count of unique items rated by all users)
6. Calculate the similarity between Ui and all the clusters that have been created up to this point.
7. If Ui is more similar to an existing cluster than the threshold_cutoff
8. Assign Ui to the most similar cluster
9. Else
10. Create a new cluster Ci
11. Assign Ui to Ci
12. Add Ci to the list of clusters
13. Return the list of clusters

End

Fig. 3. Pseudocode for UC_with SVD to address sparsity

Algorithm ICSSVD()

Input: SVD applied User-Song Matrix

Output: Item Clusters

Method:

1. Initialize the threshold_cutoff value

2. Create an empty list of clusters

3. Consider the first item vector I1

4. Create a new cluster C1 and assign I1 to it

5. For each remaining user vector Ui (where i ranges from 2 to k, and k represents the count of unique users rated the item)

6. Calculate the similarity between Ii and all the clusters that have been created up to this point.

7. If Ii is more similar to an existing cluster than the threshold_cutoff

8. Assign Ui to the most similar cluster

9. Else

10. Create a new Item Ci

11. Assign Ii to Ci

12. Add Ci to the list of clusters

13. Return the list of clusters

End

Fig. 4. Pseudocode for IC with SVD to address sparsity

3.5. Integrating Social Networks to address new user cold-start problem

Another challenge that recommendation systems must tackle is the Cold-start problem. The Cold- start problem refers to the difficulty of providing recommendations for either new users or new items. The former is referred to as the new user cold-start problem, while the latter is known as the new item cold-start problem.

Algorithm to Address Cold-Start Problem for Recommendation (CSReco)
Input: User-Item Matrix, Social Matrix
Output: Set of recommendations for the target User

1. Let U1, U2, U3, ..., Um represent the users, and I1, I2, I3, ..., In represent the items.
2. Create user clusters using a Threshold-based clustering algorithm (Form_User_Clusters()).
3. For the given target_user:
   a. Determine the user cluster that exhibits the highest similarity to the target_user.
   b. Construct a social matrix by identifying the immediate friends of the target_user from the Social Matrix.
   c. Obtain recommendations from the user clusters (Ru1, Ru2, ...) and from the social matrix (Rs1, Rs2, ...).
   d. Suggest items that are common to both recommendation sets.

End of Algorithm

Fig. 5. Pseudocode for addressing new user cold-start problem

To mitigate the new user cold-start problem, the approach involves incorporating a user's social network connections. In case of new target user , the recommendation system lacks information about them, the system leverages their friends' interests to gain insights into the target user. The algorithm for this process is presented in Fig. 5.

Another challenge that recommendation systems must tackle is the Cold-start problem. The Cold- start problem refers to the difficulty of providing recommendations for either new users or new items. The former is referred to as the new user cold-start problem, while the latter is known as the new item cold-start problem.

To mitigate the new user cold-start problem, the approach involves incorporating a user's social network connections. In case of new target user, the recommendation system lacks information about them, the system leverages their friends' interests to gain insights into the target user. The algorithm for this process is presented in Figure 5.

3.6 Addressing Long tail problem

The Long Tail refers to a collection of items that are relatively less popular, with the majority of items falling into this category [3]. In contrast, only a small fraction of items are part of the Heavy Tail, as illustrated in Fig. 6. Presently, we are transitioning towards a paradigm that focuses on distinguishing between Hits and Niche content. However, the challenge lies in effectively filtering and presenting the right artists to users based on their musical preferences, even if those artists do not fall within the Heavy Tail category.

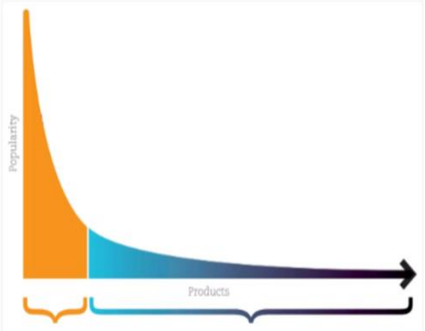

Fig. 6. Depicting Long-tail

To tackle the Long Tail problem, a solution involves categorizing songs based on their popularity frequency. This classification divides items into two categories: Long Tail and Heavy Tail, based on their average frequency. If an item's average frequency is lower than the average frequency of all items, it falls into the Long Tail category; otherwise, it is classified as belonging to the Heavy Tail. Subsequently, songs in the Long Tail and Heavy Tail segments are utilized separately to create clusters of similar songs using the algorithm depicted in Figure 6.

3.7 Session based Music Recommendation system to include context information

To integrate contextual information into music recommendations, sessions are introduced as a technique. This concept is based on the fundamental notion that users' preferences can change throughout the day. For instance, users might have a preference for devotional songs in the early morning and opt for melodic tunes in the evening. To accommodate this variability in users' listening behaviour, the implicit feedback users provide in the form of logs is classified into four specific sessions, as defined below:

- Session 1: Timestamp 0-6
- Session 2: Timestamp 6-12
- Session 3: Timestamp 12-18
- Session 4: Timestamp 18-24

Using the listening history of users, transactions are categorized into these four sessions based on timestamps. For each session, user clusters and item clusters are created using the pseudocode provided in Fig.1 and Fig.2. Within each session, the target user is linked to identify the most similar user and item clusters, facilitating recommendations.

3.8. Mobile Application to combine all the proposed approaches for top n recommendations.

A mobile application has been designed to integrate the various approaches mentioned above for delivering top N recommendations. This Android mobile application, illustrated in Figure 7, enables users to access the entire library of audio songs on their mobile devices. Whenever users listen to a song, the application generates a log entry and stores it in the local database. This log contains specific fields, as depicted in Figure 8.

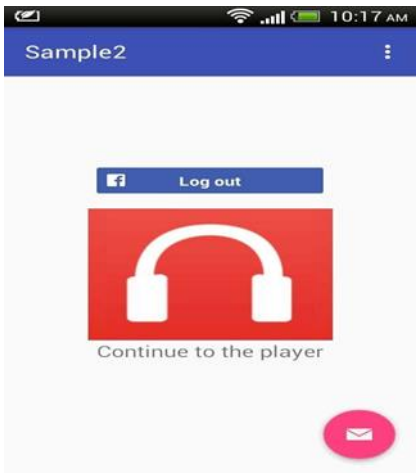

Fig. 7. Mobile Application for MRS

Tue Apr 05 06:48:20 GMT+05:30 2016	Sai Krishna	18462	[iSongs.info] 03 - Mila Mila	10	Bhale Manchi Roju - (2015)

Fig.8. Sample user log

The administrator of the mobile application employs a recommendation algorithm that leverages the approaches discussed earlier to quantify the diverse factors influencing a user's taste. This algorithm takes into account a portion of songs from each technique when creating the top N recommendations. The formula used to generate these top N recommendations is as follows, but it can be rephrased for clarity if needed.

Certainly, here's the equation representing the top N recommendations with the given factors:

Top N Recommendations = α1 * (User and Item Clusters) + α2 * (Matrix Factorization) + α3 * (Social Integration) + α4 * (Long Tail) + α5 * (Context Information)

The α values represent the proportions of songs selected from each technique, effectively serving as weights assigned to each approach. To illustrate, this means that, for instance, 20% of the songs are drawn from User and Item Clusters, 30% from Sessions, 30% from Matrix Factorization, and 20% from Social Integration.

4. Results

Last.fm user logs were considered for 3 years. 200 users listening history is taken to conduct experimental work. After pre-processing of the data U-S matrix obtained for is shown in Table 1. 70% of users randomly selected for building user and Item clusters. Remaining 30% of users are used to evaluate the proposed recommendation system. To fix a threshold value for cluster formation Elbow method is used. Started with a randomly selected threshold value and formed clusters with that value. Computed SSE value for the clusters generated. Updated threshold value and re-generated the clusters. Compared SSE of new clusters with the previous clusters. These steps are repeated until clusters with minimum SSE is obtained. After user and item cluster formation recommendation process started for test users. Top-k items are recommended to the users by applying the algorithms shown in Fig. 1 & 2. The recommendations are evaluated by using the measures such as accuracy and precision. The results obtained for different k values are shown in the table 5.

Table 5. Performance of UC and IC clusters

Top-k items	User Clusters (UC)		Item Clusters	
	Accuracy	**Precision**	**Accuracy**	**Precision**
Top-50	0.433	0.1636	0.328	0.1729
Top-30	0.319	0.164	0.311	0.1676
Top-20	0.273	0.176	0.291	0.1598

Sparsity in the U-S matrix is addressed by applying SVD. SVD is applied on the matrix and obtained a dense User and Item matrices in U and V^T. UC and IC algorithms shown in Fig.1 & 2 are applied on dense user, iem matrices to form user and item clusters respectively. The results obtained for Uc with SVD and IC with SVD are shown in table 6 & 7 respectively. The results shows that SVd improved the performance of UC and IC clusters notably.

Table 6. Performance of UC comparison with UC_SVD

Top-k items	UC		UC with SVD	
	Accuracy	**Precision**	**Accuracy**	**Precision**
Top-50	0.433	0.1636	0.453	0.2537
Top-30	0.319	0.164	0.411	0.2467
Top-20	0.273	0.176	0.371	0.2128

Table 7. Performance of IC comparison with IC_SVD

Top-k items	IC		IC with SVD	
	Accuracy	**Precision**	**Accuracy**	**Precision**
Top-50	0.328	0.1729	0.413	0.287
Top-30	0.311	0.1676	0.411	0.266
Top-20	0.291	0.1598	0.348	0.238

New-user cold-start problem is the inability of the recommendation system to provide recommendations for new-users.Social intergration of users with user listening history has addressed new-user cold-start problem. The results obtained shown in table 8. the ability of improved recommendation system with new-user cold-start problem.

Table 8. Performance of UC comparison with UC_SVD

Top-k items	UC		UC with Cold-start	
	Accuracy	**Precision**	**Accuracy**	**Precision**
Top-50	0.433	0.1636	0.415	0.207
Top-30	0.319	0.164	0.391	0.198
Top-20	0.273	0.176	0.306	0.183

Another important issue in user-centric collaborative filtering is long-tail problem. Long-tail is the set of items not popular but may be interesting to the user. Items are divided into head and tail. User centric and Item centric clustering agorithms are applied separately to both sets of items. While performing recommendation task, long tail clusters are given more weightage compared to tail items. The results shown in the table 9. address the tail items as tail precision improved with tail clusters.

Table 9. Performance of UC comparison with UC_Longtail for tail items

Top-k items	UC		UC with Long tail	
	tail_Accuracy	**tail_Precision**	**Accuracy**	**Precision**
Top-50	0.261	0.12	0.482	0.205
Top-30	0.193	0.092	0.455	0.194
Top-20	0.191	0.077	0.427	0.192

As users listening taste differs at different time of the day, the context information is captured by dividing the user history into four sessions. Uc and IC clusters formed for different sessions and the session context is considered for recommendation process. The results obtained with sessions are better compared to UC and IC clusters as shown in table 10 & 11.

Table 10. Performance of UC comparison with UC_Context

Top-k items	UC		UC with Context	
	Accuracy	**Precision**	**Accuracy**	**Precision**
Top-50	0.433	0.1636	0.434	0.271
Top-30	0.319	0.164	0.411	0.267
Top-20	0.273	0.176	0.384	0.228

Table 11. Performance of IC comparison with IC_Context

Top-k items	IC		IC with Context	
	Accuracy	**Precision**	**Accuracy**	**Precision**
Top-50	0.328	0.1729	0.491	0.297
Top-30	0.311	0.1676	0.491	0.276
Top-20	0.291	0.1598	0.421	0.218

The comparison of UC and IC methods are shown table 12 & 13 respectively. The results are evident to say that UC_SVD and IC_Context are perorming better compared to UC and IC methods.

Table 12. Comparison of UC methods

Top-k items	UC		UC with SVD		UC with Cold-start		UC with Long tail		UC with Context	
	A	**P**	**A**	**P**	**A**	**P**	**A**	**P**	**A**	**P**
Top-50	0.433	0.1636	0.453	0.2537	0.415	0.207	0.482	0.205	0.434	0.271
Top-30	0.319	0.164	0.411	0.2467	0.391	0.198	0.455	0.194	0.411	0.267
Top-20	0.273	0.176	0.371	0.2128	0.306	0.183	0.427	0.192	0.384	0.228

Table 12. Comparison of IC methods

Top-k items	IC		IC with SVD		IC with Context	
	A	**P**	**A**	**P**	**A**	**P**
Top-50	0.328	0.1729	0.413	0.287	0.491	0.297
Top-30	0.311	0.1676	0.411	0.266	0.491	0.276
Top-20	0.291	0.1598	0.348	0.238	0.421	0.218

Hybrid recommendation system proposed in the paper combines all the methods to improve the accuracy and precision. The results obtained for the equal weightage given to all the approaches and best weightage combination are shown in table 13. From the results obtained, weighted hybrid method out performed all the other UC and IC methods as shown in fig 9.

Table 12. Performnce of weighted hybrid method

Top-k items	α= 0.2 (all equal)		Best α's combination	
	A	P	A	P
Top-50	0.618	0.331	0.826	0.445
Top-30	0.611	0.334	0.818	0.442
Top-20	0.582	0.341	0.818	0.422

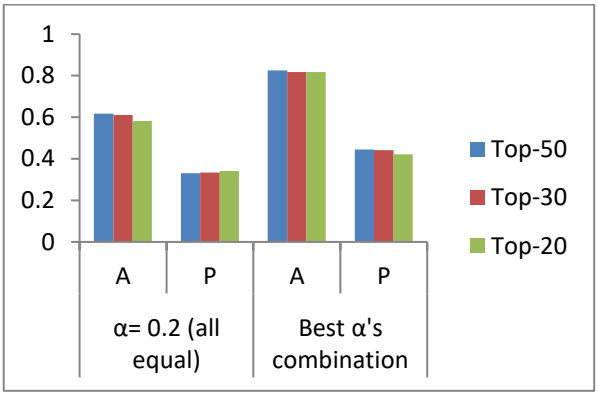

Fig.9. Performnce of weighted hybrid method

5. Conclusion & Future Scope

The research proposed here, have developed an Android-based hybrid music application that takes into account various aspects of users when making recommendations. The recommendation algorithm considers users' historical preferences, item similarities, and contextual information, effectively addressing the challenges of cold start, long tail, and sparsity in recommendation systems. Our findings demonstrate that the hybrid recommendation system we proposed outperforms baseline approaches like the most popular or most recent recommendation systems. Furthermore, the results indicate that the combination of multiple techniques for providing top N recommendations yields better performance compared to using individual techniques separately. Additionally, we can extend this hybrid mobile-based music recommendation system to incorporate user generated content, such as postings about new items on social networks, to tackle the new item cold-start problem effectively.

References

[1] Last. FM – A popular music web portal http://www.last.fm

[2] Pandora – A free internet radio. http://www.pandora.com

[3] C. Anderson. The long Tail. Wired Magazine, 12(10): 170-177, 2004

[4] Singular Value Decomposition http://en.wikipedia.org/wiki/Singular_value_decomposition

[5] Maria N. Moreno, Saddys Segrera, Vivian F Lopez, Maria Dolores Munoz and Angel Luis Sanchez, Mining Semantic Data for Solving Firstrater and Cold-start Problems in Recommender system ACM IDEAS 11 2011, September 2102

[6] A Collaborative Filtering Recommendation Algorithm Based on User Clustering and Item Clustering, SongJie Gong, JOURNAL OF SOFTWARE, VOL. 5, NO. 7, JULY 2010

[7] The Million Song Dataset Challenge, Brian McFee, Thierry Bertin-Mahieux, Daniel P.W. Ellis, Gert R.G. Lanckriet, WWW 2012 Companion, April 16–20, 2012, Lyon, France

[8] Context-aware item-to-item recommendation within the factorization framework, Balázs Hidasi, Domonkos Tikk, CaRR'13, February 5, 2013, Rome, Italy

[9] Session Aware Recommender System In ECommerce, Jian WangA dissertation submitted in partial satisfaction of the requirements for the degree of DOCTOR OF PHILOSOPHY, June 2013

[10] Method of Collaborative Filtering Based on Uncertain User Interests Cluster, Xiang Cui, Guisheng Yin, Long Zhang, Yongjin Kang, JOURNAL OF COMPUTERS, VOL. 8, NO. 1, JANUARY 2013.

[11] Junmei Feng , Zhaoqiang Xia , Xiaoyi Feng , Jinye Peng, RBPR: A hybrid model for the new user cold start problem in recommender systems, Knowledge-Based Systems, Volume 214, 28 February 2021, 106732

[12] Wang, S., Wang, Y., Sivrikaya, F. et al. Data science for next-generation recommender systems. Int J Data Sci Anal 16, 135–145 (2023). https://doi.org/10.1007/s41060-023-00404-w

[13] Bing Bai , Yushun Fan, Wei Tan, Jia Zhang DLTSR: A Deep Learning Framework for Recommendation of Long-tail Web Services, March 2017, IEEE Transactions on Services Computing PP(99):1-1

[14] DOI:10.1109/TSC.2017.2681666

[15] D. H. Park, H. K. Kim, I. Y. Choi and J. K. Kim, "A literature review and classification of recommender systems research", *Expert Syst. Appl.*, vol. 39, no. 11, pp. 10059-10072, Sep. 2012.

[16] Fajie Yuan, Guibing Guo, Joemon M. Jose, Long Chen, Haitao Yu, Weinan Zhang, LambdaFM: Learning Optimal Ranking with Factorization Machines Using Lambda Surrogates, CIKM '16: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, 2016.

[17] Cheng, M., Liu, Q., Zhang, W. et al. A general tail item representation enhancement framework for sequential recommendation. Front. Comput. Sci. 18, 186333 (2024). https://doi.org/10.1007/s11704-023-3112-y

[18] J. Li, K. Lu, Z. Huang, L. Zhu and H. T. Shen, "Heterogeneous Domain Adaptation Through Progressive Alignment," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 5, pp. 1381-1391, May 2019, doi: 10.1109/TNNLS.2018.2868854.

[19] K. L. P. Suryanarayana, G. ., Swapna, N. ., Bhaskar, T. ., & Kiran, A. . (2023). Optimizing K-Means Clustering using the Artificial Firefly Algorithm. International Journal of Intelligent Systems and Applications in Engineering, 11(9s), 461–468.

[20] Sudhakara, M. ., Meena, M. J. ., Madhavi, K. R. ., Anjaiah, P. K, L. P. , Fish Classification Using Deep Learning on Small Scale and Low-Quality Images. International Journal of Intelligent Systems and Applications in Engineering, 10(1s), 279

[21] Ugendhar Babu Illuri,Sridhar Reddy Vulapula, Marepalli Radha, Sukanya K,Fayadh Alenezi,Sara A. Althubiti,Kemal Polat, A Novel Intelligent-Based Intrusion Detection System Approach Using Deep Multilayer Classification, Mathematical Problems in EngineeringVolume 2022 | Article ID 8030510 | https://doi.org/10.1155/2022/8030510