International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

# An Integrated Recommender System for Automated Playlist Expansion

Shaktikumar V. Patel*[1], H. B. Jethva[2], Dipak C. Patel[3]

**Submitted**: 25/01/2024 **Revised**: 03/03/2024 **Accepted**: 11/03/2024

**Abstract:** The rapid growth of digital music platforms has made it challenging for users to curate and maintain playlists. To tackle this concern, an automated playlist extension system has been devised to aid users in creating seamless and personalized playlists. This article proposes a novel recommender system that initially retrieves large set of tracks/songs through collaborative filtering techniques then re-ranks the retrieved songs to enhance the accuracy of the music playlist recommendations. With this integrated approach of recommending songs, users get seamless experience in continuation of playlist. Through the experimental evaluations using a real-world dataset provided by Spotify, the integrated recommender system exhibits superior performance in recommendation accuracy and user satisfaction compared to alternative collaborative filtering methods. The findings witness the advantages of integrating various recommendation methodologies to enhance the robustness and personalization of music playlist expansion.

***Keywords:*** *Music Recommendation System, Hybrid Recommendation Techniques, Automated Playlist Continuation.*

## 1. Introduction

The growing popularity of digital music platforms has led to a vast amount of music being available to users. Consequently, users often face the challenge of curating and maintaining playlists that align with their preferences and moods. Manual playlist curation can be time-consuming and subjective, leading to playlist stagnation and user dissatisfaction. Automatic playlist expansion [1] systems have emerged as a solution to this problem, aiming to generate personalized and seamless playlists for users.

In this article, we introduce an integrated method for automatic playlist extension through the implementation of a hybrid recommender system. Many researchers commonly combine the collaborative filtering (CF) and content-based filtering techniques in such hybrid systems to leverage their respective strengths [2]. Collaborative filtering leverages the power of user behaviour patterns to identify similarities among users and recommend songs for automatic playlist expansion as per liking of similar users.

Conversely, the content-based filtering approach [3] prioritizes the intrinsic characteristics of songs, such as genre, artist, and mood, to suggest tracks that possess similar attributes, thereby creating relevant additions to the corresponding playlist.

The prime goal of our research is to enhance the accuracy of playlist recommendations by intelligently combining multiple collaborative approaches in first stage followed by re-ranking algorithms within a hybrid recommender system. By leveraging playlist features, song/track features and collaborative filtering methods, our system aims to provide a comprehensive and satisfying playlist continuation experience. The collaborative filtering based algorithms analyses user behaviour patterns, such as play history, likes and skips, to know users with similar tastes and recommend songs that align with their preferences. The pairwise ranking algorithm, on the other hand, utilizes the intrinsic attributes of songs and playlist features to elevate the accuracy of the recommended songs for the playlist.

The hybridization approach we propose combines the outputs of multiple collaborative based techniques to generate a well-rounded playlist recommendation in the first stage. Through the integration of multiple recommendation strategies, our system overcomes the limitations of individual methods and strives to deliver users with recommendations that are more precise and pertinent. Utilizing a blend of collaborative filtering (CF) and ranking models, we effectively obtained 20,000 potential candidates for each playlist across various categories. Subsequently, we devised a pairwise model in the second phase, which assigned a relevance score to each (playlist, song) pair directly. By incorporating features of both playlists and songs into the input, this model was capable of capturing intricate pairwise connections that conventional CF techniques find challenging to articulate. The primary objective of the second phase was to apply ranking to the candidate songs, with a focus on enhancing the accuracy value at the forefront of the recommended list.

[1]*Research Scholar, Computer Engineering Department, Gujarat Technological University, Gujarat, India.*
*ORCID ID : 0009-0003-8504-1951*
[2]*Computer Engineering Department, GEC Patan, Gujarat, India.*
*ORCID ID : 0000-0003-2954-117X*
[3]*Information Technology Department, VGEC Chandkheda, Gujarat, India.*
*Email: Dipak.patel@vgecg.ac.in*
*ORCID ID : 0000-0002-4163-2261*
* *Corresponding Author Email: shakti.infredible@gmail.com*

The model was tested on distinct playlists of 10,000 records, where a portion of songs was held back. It is noteworthy that the lengths of these test playlists varied significantly, ranging from no songs (cold start) to hundred songs.

## 2. Related Work

There are several methods / techniques and approaches that can be used automatic music playlist expansion to generate songs recommendation [4], some of which are discussed below:

### 2.1 Playlist Expansion Systems

Automatic playlist expansion [15] has gained significant attention in the field of songs recommendation. Many previous studies have focused on developing techniques and algorithms to address the challenge of generating seamless and personalized playlists. A widely used method is collaborative filtering, which analyses user behaviour data to find similar users and suggest songs as per their preferences. Content-based filtering techniques, alternatively, focus on the intrinsic characteristics of songs, such as genre, artist, and mood, to recommend tracks that share similar attributes. While these methods have shown promise individually, researchers have also explored hybridization approaches to combine the strengths of different techniques and deliver more accurate and varied playlist continuation.

### 2.2 Collaborative Filtering Approach

Collaborative filtering [5] stands as a prevalent technique employed in music recommendation systems, drawing upon user behaviour and preferences to formulate music suggestions.

In collaborative filtering, the system identifies playlists with similar musical tastes and preferences and suggests music that these playlists have played in the past. This is done by analysing data on user listening history, playlists, and ratings to identify patterns and similarities between songs and playlists.

Collaborative filtering has several advantages in music recommendation systems. First, it can deliver personalized recommendations that return the playlist's musical preferences and tastes. Second, it can help playlists to discover new songs that they might not have found otherwise. Finally, it can help to increase user engagement and retention on music platforms by providing a continuous stream of personalized recommendations. Collaborative filtering [9, 10] techniques have certain drawbacks. First biggest issue is of cold start problem in which new playlists or playlists with no songs would be difficult to generate recommendations. Second issue is of biasing where this approach tends to recommend only popular songs.

### 2.3 Content-Based Filtering Approach

Content-based filtering [3] is another commonly used approach in music recommendation systems that focus on the characteristics of the music itself rather than playlist-song behaviour and preferences.

In content-based filtering, the system analyses the musical features of tracks such as tempo, genre, rhythm, instrumentation, and lyrics to generate music recommendations that are similar to the music tracks that the playlist already contained.

Content-based filtering has several advantages in music recommendation systems. First, it can provide recommendations for niche or lesser-known music that may not have a large user base or listening history. Second, it can provide recommendations that are not biased towards popular music or trends. Finally, it can help users discover new music that is similar to what they already enjoy but with different musical features, thereby expanding their musical horizons.

However, content-based filtering may have some drawbacks, such as the incapability to arrest the subjective inclinations of users and the difficulty of accurately characterizing the complex and subjective nature of music. Hence, integrating both content-based filtering and collaborative filtering approaches often results in music recommendations that are more precise and impactful.

### 2.4 Hybrid Filtering Approach

Hybrid filtering [2, 11] technique is a blended mode of content-based filtering and collaborative filtering techniques in music recommendation systems. This strategy is commonly adopted to address the constraints of individual filtering techniques and to offer music continuation that are both more precise and varied.

In the hybrid filtering [11] technique, the system initially produces recommendations by employing two filtering techniques separately. Then, the system combines these recommendations using a variety of weighting and ranking algorithms to provide a final set of personalized and diverse music recommendations.

Hybrid filtering offers numerous advantages in music recommendation systems. Firstly, it enhances recommendation accuracy and diversity by merging the strengths of collaborative filtering and content-based filtering methods. Secondly, it tailors recommendations more precisely by considering both user behaviour and musical attributes. Lastly, it aids in addressing the shortcomings of individual filtering techniques, such as the cold start problem or data sparsity.

Through an examination of current literature on playlist expansion systems, we came on conclusion that single recommendation systems suffer from one or more drawback. To overcome the drawback of individual method and combine the advantages from multiple approaches, we introduced integrated recommendation approach for automatic playlist expansion, which integrates multiple recommendation techniques to improve recommendation accuracy and diversity. Subsequent sections present the methodology, experimental setup of our approach, comparative results and analysis with existing approaches

## 3. Methodology

### 3.1 Integrated Recommendation Approach

In the initial phase, the latent CF (collaborative filtering) model using WRMF (Weighted Regularized Matrix Factorization) concept [5] swiftly identifies 20,000 potential songs for each playlist. Further, the user to user and item to item neighbour based models are considered to generate score for each identified songs. Subsequently, scores from all models and their weighted combinations are merged with the retrieved playlist-song data and fed into the next stage of the model. The gradient boosting model then again prioritizes all candidate songs, producing the concluding ranking. With the exception of the cold start scenario, all playlists adhere to a uniform two-stage paradigm regardless of their duration. This deliberate choice streamlines complexity and reduces training time. The notations employed are detailed below:

- R represents a matrix of playlists and songs, where $R_{ij}$ is set to 1 if ith song is present in jth playlist, and $R_{ij}$ is set to 0 otherwise. V(i) is considered as an universal set of songs in ith playlist, similarly U(j) is a set of playlists containing song j.
- U and V represent Latent representations of playlists and songs, where $U_i$ and $V_j$ represent the latent representations for ith playlist and j song, correspondingly.
- S represents relevance scores of predication, where Sij indicates the relevance score of ith playlist and jth song.

WRMF (Weighted Regularized Matrix Factorization) [5] is a widely recognized latent model utilized for binary/implicit collaborative filtering. Although, it was introduced over a decade ago, our findings reveal that if it is appropriately tuned, we can get sufficient improvement in the accuracy.

Neighbor-based Models: We utilize two widely adopted neighbor-based collaborative filtering models [12]: User to User and Item to Item [6]. The User to User method assesses significance by evaluating the match from rows of the playlist-song matrix, R. Specifically, for a given (i, j) which is playlist-song pair, User to User model compares all playlists containing song j to playlist i:

$$S_{i,j}^{playlist} = \sum_{i' \in U(j)} \frac{R_i \cdot R_{i'}}{\|R_i\| \cdot \|R_{i'}\|}$$

Here, $R_i$ represents the ith row in R matrix. The underlying concept says that if jth song is present in numerous playlists having similarities with playlist i, then the relevance score $S_{i,j}^{playlist}$ is expected to be high, indicating that jth song should be considered for the recommendation.

Likewise, the Item to Item model assesses matching by calculating the relationship between columns of the playlist-song matrix, R. This technique involves comparing song j with songs present in ith playlist.

$$S_{i,j}^{track} = \sum_{j' \in V(j)} \frac{R_j \cdot R_{i'}}{\|R_j\| \cdot \|R_{j'}\|}$$

In this context, Rj represents the jth column of the matrix R. Similar to the User to User method, if song j exhibits similarity to already included songs in ith playlist, then the relevance score S_(i,j)^track will attain higher value and song j is considered for the recommendation.

**Model Combination**: In preceding sections, various collaborative filtering (CF) models were detailed, including WRMF (Weighted Regularized Matrix Factorization), Item-Item, and User-User [6]. In this section, we delineate the approach to combining these models. Opting for a single best model might seem straightforward. However, this method may lead to excessively confident predictions and heightened variability, as it overlooks uncertainty of the model in favor of explicit model assumptions. Therefore, the preference lies in combining multiple models.

To mitigate overfitting, we employ a linear weighted ensemble method. This involves linearly combining all model scores using model-specific weights:

$$S^{Blend} = w_1 S^{wrmf} + w_2 S^{playlist} + w_3 S^{track}$$

Before blending, the scores for each model undergo standardization by deducting the mean and division by the standard deviation. By ensuring that the scores are rescaled into the same range, standardization makes it easier to compare models. In order to further reduce overfitting, the ensemble's weights are chosen haphazardly from the set {0.4, 0.3, 0.3}. This set is purposefully kept tiny. We arrive at the following weight combination after certain rounds of optimization: w1 = 0.4, w2 = 0.3, w3 = 0.3. This particular value of weights yields the high accuracy and is used in all our experiments. The three scores $\{S^{wrmf}, S^{playlist}, S^{track}\}$

are fed into the next stage of experiment for each contestant song.

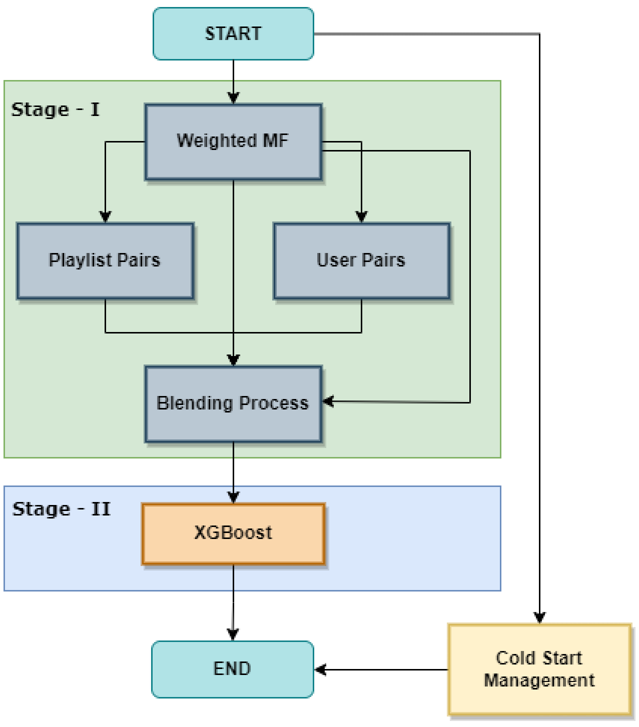

**Fig1:** Two Stage Integrated Recommendation Model

### 3.2 Playlist Generation

The goal of the second stage is to carefully reorder the candidates that were retrieved in the first stage in order to optimize accuracy at the top of the suggested list. Since the pool of candidates is small, accuracy can take precedence above efficiency in the second stage of model. Consequently, we concentrate on pairwise interactions to create a model which concurrently associates relevance ratings with (playlist, song) pairings. This method enhances the initial phase, in which pairwise interactions are not taken into account by any of the models. Both the model architecture and input features which are described in more detail below are the main elements of the second stage.

**Feature Extraction:** To capture every important facet of playlist-song relevance, a thorough feature engineering process was undertaken. The following groupings are considered to the final list of features used:

• **Input from First Stage-** The first stage's scores are used directly as input characteristics. This makes it possible for the second stage model to quickly extract the first stage's performance and then concentrate on improving it.

• **Playlist Features-** These features provide details on the music and content that are included in playlists. They consist of elements including song homogeneity, average popularity of songs, artists, and albums, playlist name and duration, and more. By comparing the latent representation of the playlist with every song on it, homogeneity is ascertained. We found that certain playlists are mostly made up of music from particular genres or styles, which makes the suggestion process easier, while other playlists include a much wider variety. The homogeneity score stands out as one of the most important characteristics in this category and shows a strong association with song diversity.

• **Song Features-** Our goal is to summarize data on song content and the kinds of playlists that include the song, just like playlist features do. We include information like playlist statistics, song duration, and artist/album/title of the song. Furthermore, we compare the song's representation to every playlist that includes it in order to determine the homogeneity score. This function evaluates whether the music has shown to be beneficial and frequently appears in playlists of a similar kind.

• **Playlist-Song Features-** Each playlist-song pair's pairwise similarity is directly characterized by this feature group, which makes it essential. We calculate similarity metrics between the playlist's songs and the target song, as well as between the target playlist and playlists that contain the target song. Similar with item to item and user to user matching, these features also provide extra content information. For instance, we compute length, variances in time, overlap between artists and albums, and average latent score similarity. The greatest notable improvement over the first stage was obtained by incorporating this feature group, indicating that further work should be focused on improving this area.

**Model Architecture:** Because of its outstanding performance, we chose to use a tree-based gradient boosting model (GBM) in the second stage, which makes use of the XGBoost [7] library. Using the 20,000 song candidates that the first step had collected, we create the training set by selecting (up to) 20 relevant songs and 20 irrelevant songs at random for each playlist. The training set consists of these samples, which have binary objectives that denote relevance or irrelevance. Next, considering training loss, we use pairwise ranking loss in a gradient boosting mode. Based on empirical investigation, we discovered that for this specific task, ranking loss performed better than the binary cross-entropy objective. Our gradient boosting model uses 150 trees at a depth of 10 for all entries.

## 4. Experimental Setup

### 4.1 Dataset for Experiment

Spotify made available to the public the Million Playlist Dataset [8], which consists of one million playlists including track listings, titles of playlist, and extra metadata for each item. The playlists range in size from 5 to 250 tracks. To address playlists with fewer tracks, we leverage playlist titles to glean contextual information associated with each playlist, as these textual descriptors provide insight into the playlist's purpose. Regarding the tracks themselves, the dataset includes 2,262,292 distinct tracks, of which 1,073,419 (47.45%) only appear once throughout the playlists. 99.9971% is the computed data sparsity. We focus solely on the artist only, even though each track of the dataset contains artist and album information. The dataset also includes 734,684 unique albums and 295,860 unique artists.

### 4.2 Preprocessing

Due to the lack of statistically meaningful patterns in analysing rare samples, we took a decision to remove rare tracks and artists. For our experiment, we considered threshold value of 5 and 3 for selection of tracks and artists. Following this selection process, we able to reduce the number tracks to 5,98,293 which is 26% of total and similarly artists results into 1,55,942 which is 53% of total. As a result, the data sparsity is reduced to 99.99%. While this pruning results in the loss of some information regarding playlists and their contents, it has minimal impact on the accuracy of our model. In terms of playlist titles, we handle 41 characters, including 10 numbers (0-9), 26 alphabets (a-z), and 5 special characters (/<>+-). The maximum title length is capped at 25 characters.

### 4.3 Evaluation

Predictions are assessed using two distinct metrics [13] R-precision and Normalized Discounted Cumulative Gain (NDCG) metric, considering the validation set called Gt and the recommended set called Rt.

**R-precision:** This measure is assessed at correctly recommended tracks and any other track of similar artist (i.e. artist level. The calculation of R-precision for the level of track is as follows where we denote set of tracks by $G_t$ and Retrieved tracks by $R_t$.

$$R_{prec\_t} = |G_t \cap R_t|G_t| \,|\, |G_t|$$

For every track that hasn't received matching at track level, Let $G_a$ stand for the validation set of having distinct artists from $G_t$ and $R_a$ for the suggested list of artists from set $R_t$. The following formula is used to determine the artist level R-precision:

$$R_{prec\_a} = |G_a \cap R_a| \,|G_a|$$

There is only one count per artist and playlist for a match at level of artist. The following formula determines the final score of R-precision is:

$$R_{prec} = R_{prec\_t} + 0.25 * R_{prec\_a}$$

Higher values in the Normalized Discounted Cumulative Gain (NDCG) indicate that relevant tracks are ranked higher in the list. This metric evaluates the ranking quality of the recommended songs. Discounted Cumulative Gain (DCG) is calculated, and then divided by the ideal DCG (where the suggested tracks are properly ranked) to obtain the NDCG.

$$DCG = rel_1 + \sum_{i=2}^{|R|} \frac{rel_i}{\log_2(i+1)}$$

The equation of IDCG is following:

$$IDCG = 1 + \sum_{i=2}^{|G|} \frac{1}{\log_2(i+1)}$$

If intersection of two sets G and R is empty then the value of DCG is set to zero. The equation of NDCG is given as:

$$\text{NDCG} = \frac{DCG}{IDCG}$$

## 5. Results and Analysis

To assess the effectiveness of the proposed model, we present the outcomes separately for 10 distinct classes of playlists. As previously mentioned, the evaluation dataset comprises 10,000 playlists, with each class consisting of 1,000 playlists as follows: (1) simply with titles and no tracks; (2) with titles and the first five tracks; (3) with only the first five tracks; (4) with titles and the first ten tracks; (5) with only the first ten tracks; (6) with titles and the first twenty-five tracks; (7) with titles and the twenty-five randomly selected tracks; (8) with titles and the first hundred tracks; (9) with titles and the hundred randomly selected tracks; and (10) with titles and only the first track.

This division of sets is intended to mimic the real-world implementation of the playlist continuation model, guaranteeing that the model operates efficiently at different phases of playlist development, ranging from playlist with zero songs to playlists containing many songs.

We employ two evaluation metrics [14] to assess the model performance: R-Precision (RPREC) and Normalized Discounted Cumulative Gain (NDCG). For both metrics, we set the upper threshold value to 500 songs which signifies that model needs to get 500 songs for every playlist.

Table 1 below depicts the results of R-precision calculated across all classes of playlists of the dataset. Here we observe that the hybrid approach outperforms other baseline methods. Fig 2 represents the corresponding clustered column chart. Table 2 depicts the result of NDCG across all classes of playlists. It shows that hybrid approach again outperforms the other baseline algorithms mentioned.

**Table 1**. R-Precision results of all classes

| Class No. | Collaborative | KNN | Hybrid |
|---|---|---|---|
| 1 | 0.068 | 0.062 | 0.071 |
| 2 | 0.174 | 0.172 | 0.182 |
| 3 | 0.191 | 0.185 | 0.194 |
| 4 | 0.194 | 0.191 | 0.198 |
| 5 | 0.188 | 0.186 | 0.193 |
| 6 | 0.212 | 0.209 | 0.221 |
| 7 | 0.273 | 0.271 | 0.286 |
| 8 | 0.201 | 0.197 | 0.219 |
| 9 | 0.321 | 0.315 | 0.324 |
| 10 | 0.112 | 0.114 | 0.137 |

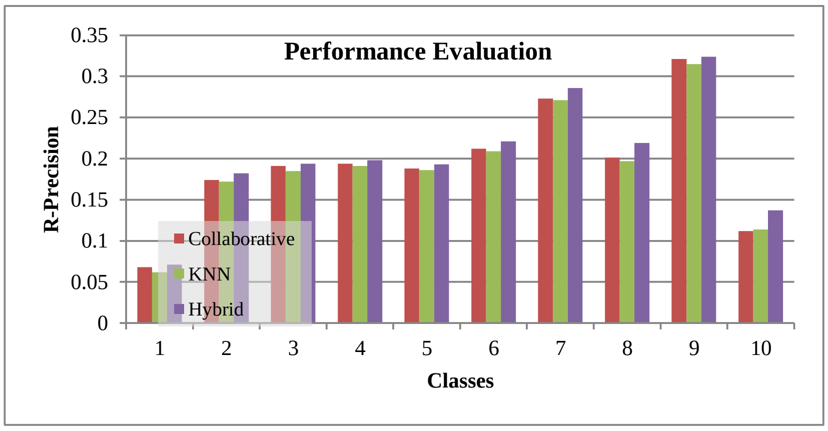

**Fig 2.** R-precision comparisons across all classes

**Table 2:** NDCG results of all classes

| Class No. | Collaborative | KNN | Hybrid |
|---|---|---|---|
| 1 | 0.171 | 0.169 | 0.175 |
| 2 | 0.305 | 0.301 | 0.318 |
| 3 | 0.325 | 0.322 | 0.327 |
| 4 | 0.331 | 0.328 | 0.332 |
| 5 | 0.339 | 0.337 | 0.341 |
| 6 | 0.349 | 0.351 | 0.358 |
| 7 | 0.409 | 0.407 | 0.416 |
| 8 | 0.363 | 0.359 | 0.372 |
| 9 | 0.475 | 0.471 | 0.482 |
| 10 | 0.274 | 0.271 | 0.287 |

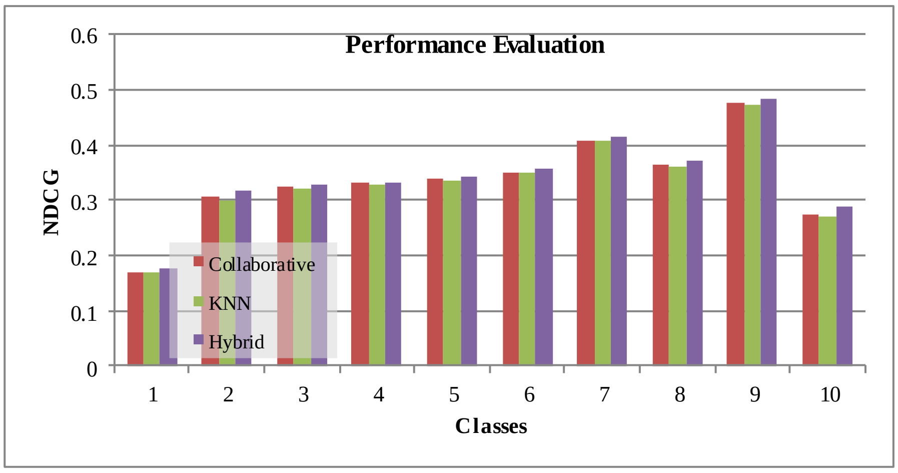

**Fig 3.** NDCG comparisons across all classes

## 6. Conclusion

In this research, we proposed and evaluated an integrated recommender system for automatic playlist expansion. The system combines multiple techniques to provide personalized, diverse, and interesting song recommendations to users. Through a thorough experimental evaluation, we demonstrated the superiority of the system over standalone collaborative filtering and content-based filtering models in terms of R-precision and Normalized Discounted Cumulative Gain (NDCG). We received around 22% and 20% higher R-precision compare to collaborative filtering and KNN respectively. Similarly we obtained 5% and 6% rise in NDCG compare to collaborative filtering and KNN approaches respectively.

In the first phase, we extract a high recall set of candidates by combining temporal, neighbor-based, and latent models. Then, in the next phase, these candidates are re-ordered based on extensive pairwise data. This approach makes it possible to apply sophisticated feature engineering in subsequent stage without affecting runtime speed, which makes it easier to provide end-to-end playlist suggestions.

Overall, our research contributes to the advancement of automatic playlist expansion and reinforces the potential of hybrid recommender systems in the field of music recommendation. By leveraging multiple model techniques, we can deliver more accurate, diverse and engaging music recommendations, enhancing the music listening experiences of users worldwide.

## References

[1] Schedl M, Knees P, Gouyon F (2017) New paths in music recommender systems research. In: Proceedings of the 11th ACM conference on recommender systems (RecSys 2017), Como, Italy

[2] Aggarwal CC (2016) Ensemble-based and hybrid recommender systems. Inss: Recommender systems. Springer, pp 199–224.

[3] Aggarwal CC (2016) Content-based recommender systems. In: Recommender systems. Springer, pp 139–166.

[4] Schedl, Markus, et al. "Current challenges and visions in music recommender systems research." International Journal of Multimedia Information Retrieval 7 (2018): 95-116

[5] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In IEEE International Conference on Data Mining

[6] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International Conference on World Wide Web (WWW 2001), 285-295.

[7] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In International Conference On Knowledge Discovery and Data Mining.

[8] Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, and Mehdi Elahi. [n. d.]. RecSys Challenge 2018: Automatic Playlist Continuation.

[9] Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 426-434.

[10] Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. Advances in Artificial Intelligence, 2009, Article ID 421425.

[11] Lee, J., & Lee, J. H. (2017). Hybrid Music Recommendation: A Combination of Collaborative Filtering and Content-Based Filtering. PLoS ONE, 12(1), e0168985.

[12] Desrosiers, C., & Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In Recommender Systems Handbook (pp. 107-144). Springer.

[13] Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems, 22(1), 5-53.

[14] Cremonesi, P., Koren, Y., & Turrin, R. (2010). Performance of recommender algorithms on top-n recommendation tasks. ACM Transactions on Intelligent Systems and Technology, 1(3), 1-22.

[15] A. Gatzioura, J. Vinagre, A. M. Jorge and M. Sànchez-Marrè, "A Hybrid Recommender System for Improving Automatic Playlist Continuation," in IEEE Transactions on Knowledge and Data Engineering, vol. 33, no. 5, pp. 1819-1830, 1 May 2021, doi: 10.1109/TKDE.2019.2952099.