



# Exploring the effectiveness of user-driven intent-based recommendation models implemented in a real-world music web service

Kosetsu Tsukuda<sup>1</sup> · Keisuke Ishida<sup>1</sup> · Kento Watanabe<sup>1</sup> · Masahiro Hamasaki<sup>1</sup> · Masataka Goto<sup>1</sup>

Received: 28 August 2024 / Revised: 17 April 2025 / Accepted: 27 June 2025  
© The Author(s) 2025

## Abstract

This paper explores the effectiveness of a flexible song recommendation function implemented in a music web service. The function allows users to create recommendation models, which we refer to as intent-based recommendation models (IBRMs), according to their intents. For example, a user can develop IBRMs for “cool songs,” “songs for concentrating on work,” and so on, and receive recommendations from each of the IBRMs according to her intents. The key novelty of this work lies in the architecture that enables users to explicitly construct and maintain multiple personalized recommendation models in parallel, each specialized for a particular intent. This user-driven approach contrasts with conventional systems that rely on a single, system-controlled recommendation model per user. To develop an IBRM, the user first initializes it by choosing seed songs and then repeatedly updates it by giving feedback based on whether recommended songs are relevant to the user’s intent. In the case study using the real-world web service “Kiite,” we analyze 1,116 IBRMs created by 417 users and show key characteristics of those IBRMs (e.g., it is meaningful to enable users to create their own IBRMs, because the created IBRMs generate largely different recommendation results from one another). These findings demonstrate the effectiveness and practical value of enabling users to control intent-specific recommendation behavior through the proposed IBRM framework.

**Keywords** Recommender systems · Song recommendation · Web service · User intent

## 1 Introduction

Listening to music is an essential activity in people’s everyday lives (not only during personal time [1–3] but also during work hours [4]). As music streaming has become commonplace and enabled people to access a vast amount of music [5, 6], song recommendation has become essential for people to efficiently find new songs. To estimate people’s musical preferences, a variety of methods, from typical content-based and collaborative filtering [7–16]

---

Extended author information available on the last page of the article

to recent deep-learning-based methods [17–20], has been proposed. Most of these methods estimate users' preferences from songs listened to previously.

The recommendation accuracy on evaluation datasets has improved with the advent of more sophisticated methods. When such methods are applied to a real web service or smartphone application, it is assumed that only one recommendation model (a.k.a. a user profile [21, 22]) is created for each user. However, this assumption can cause the following problems.

- On a music streaming service, Olivia usually listens to intense music such as rock and metal. One day, she wants to listen to music for concentrating on work, and she plays a mellow song. According to her play history, which includes many intense songs, even after the mellow song, the service tends to recommend intense songs that are not suitable for concentration. That is, even if appropriate recommendations are usually generated by the personalized recommendation model, there is a problem of not being able to receive recommendations that meet a user's various intents at different times.
- James usually does not listen to jazz. One day, a friend recommends a jazz song to him, so he listens to it on his regular music streaming service and enjoys it. After listening to it, however, the service starts recommending other jazz songs related to that song. This unintended change in the recommendation model is inappropriate because he only wants to occasionally receive jazz-related recommendations when that is his intent. In other words, when a user does not want to change the regular recommendations that the user typically receives, there is a problem of recommendations becoming *contaminated* by temporary interests.

To solve these problems, in this paper, we propose the concept of an intent-based recommendation model (IBRM), which enables users<sup>1</sup> to develop multiple song recommendation models based on their intents and receive recommendation results corresponding to those intents. Moreover, as a case study, we implemented and released the proposed concept in a web service called *Kiite*<sup>2</sup> for the music domain. A *Kiite* user can develop an IBRM without any expert knowledge of recommendations. To develop an IBRM, the user first inputs a name for the model and chooses at least one seed song. For example, suppose that she creates an IBRM named “songs for concentrating on work.” The created model then returns recommendation results to her. In addition to listening to the recommended songs, she can give feedback based on whether each of the recommended songs is relevant to the model's intent. Note that this feedback is independent of the user's usual musical preferences. Therefore, even if a recommended song matches her usual musical preferences, she may give negative feedback on the song if it is not suitable for concentrating on work. By incorporating such feedback, the model is updated and returns new recommendation results. Then, by repeating the cycle of giving feedback and updating the model, the user can further develop the model so that it correctly reflects her intent. Moreover, when multiple IBRMs are created, the user can receive recommendation results from each model independently.

By creating and switching between multiple IBRMs, a user can receive recommendation results according to her intent at the moment, which solves the first problem. Moreover,

<sup>1</sup>Throughout this paper, the term “user” indicates the “end user” of a service.

<sup>2</sup>“*Kiite*” means “listen” in Japanese. <https://kiite.jp>

when the user is temporarily interested in a particular song, she can create an IBRM by choosing that song as a seed. Because each IBRM generates recommendations independently, the user interaction with an IBRM does not affect either the other models' recommendations or the regular recommendations that the user usually receives, thus solving the second problem. On Kiite, users can also delete created IBRMs; hence, by deleting a created IBRM, a user can receive no more recommendations from that model.

Our contributions and innovations in this paper can be summarized as follows.

- We propose a novel user-driven architecture for intent-based recommendation models (IBRMs) that enables users to explicitly create, manage, and update multiple recommendation models based on their individual intents. Unlike conventional systems that maintain a single, system-controlled recommendation model, our approach allows users to create multiple recommendation models according to various intents.
- We implemented and publicly released the proposed user-driven IBRM architecture in a real-world music web service, Kiite. This allows non-expert users to interactively build personalized, intent-specific models through intuitive interfaces and feedback mechanisms.
- We conducted a large-scale empirical study based on long-term interaction logs from 417 users and 1,116 IBRMs, collected over more than two years. This analysis reveals key characteristics of how users use IBRMs in practice, confirming the feasibility and value of user-driven control in recommendation systems.
- We showed that IBRMs produced significantly different recommendation results even for models with same names, and that recommendation results by a user's IBRM were greatly different from those by the recommendation model reflecting the user's usual musical preferences. These findings empirically validate the necessity of supporting multiple user-driven recommendation models to capture diverse information needs.

## 2 Related work

### 2.1 Context-aware recommender systems

Context-aware recommender systems (CARS) incorporate context information such as emotion, time, and location to improve the quality of recommendations [23–31]. For example, CARS for music that use time information can recommend different songs on weekends from those recommended on weekdays. CARS have been implemented in not only music domain [23–27, 31] but also other domains including movies [31, 32], points of interest [33–35], and e-commerce [34, 36].

Despite the usefulness of considering context information, CARS have the following three limitations. First, CARS can only consider contexts that have been pre-determined for use by the system. For example, a user cannot receive recommendations for a rainy day if the system does not incorporate weather information as context. Second, CARS can certainly deal with various contexts by sensing as many kinds of contexts as possible. However, it is not always easy to measure contexts such as a user's location, activity, and mood because of the unavailability of physical sensing infrastructures and privacy problems [37]. Third, when a user is in a specific context, she does not always listen to songs that are relevant to

that context. For example, even if a user typically likes listening to rock on weekends, she may listen to jazz music introduced by a friend on a particular weekend. In such a case, the recommendation results for the context of “weekend” become contaminated, as in one of the examples given in section 1.

In contrast, IBRMs can overcome these limitations because they enable users to create recommendation models for any context (i.e., according to their intent) just by choosing at least one seed song, without requiring any sensing devices or private information. In addition, users can easily and explicitly switch between IBRMs so that their recommendation results are not contaminated. We believe that such interactive switching by users is important for user-centric music recommendation, though automatic context-dependent recommendation by CARS is helpful.

## 2.2 Playlist generation and radio stations

Because it is time consuming to manually create a playlist [38], two approaches have been studied to ease the process: automatic playlist generation and assisted playlist creation. In both approaches, a user typically chooses at least one seed song [39]. In the former approach, a list of songs is automatically generated by considering multiple constraints such as the playlist length and the continuity between songs [9, 40–43]. Recently, approaches have been proposed that generate playlists using reinforcement learning, where the characteristics of songs included in the playlist [44] and user satisfaction with the generated playlist [45] are treated as rewards. In the latter approach, the user gives feedback by means such as liking and skipping songs [46, 47]; the playlist is then modified according to the feedback. In other systems, the user manually adds songs to a playlist from songs that are recommended according to tags selected by the user [48] or similarity to a seed song(s) [49]. When users create playlists based on recommended songs, different playlist topics lead to the adoption of recommendations with different audio characteristics [50]. These previous approaches aim to generate a fixed list of songs so that users can repeatedly listen to generated static playlists. In contrast, although our IBRM approach also lets users choose seed songs and give feedback, it aims to keep updating recommendation results over a long period of time and thus introduce users to various songs that are chosen dynamically according to their created IBRMs.

On YouTube Music<sup>3</sup>, a user can start a “radio station” from a seed song, album, artist, or playlist. The generated radio station continuously plays songs and updates them according to the user’s feedback (thumbs-up/thumbs-down for each song). Unlike our IBRM architecture, the feedback on YouTube Music is *global*: a song receiving a thumbs-up in one radio station also receives a thumbs-up in other radio stations where it appears. Therefore, generated radio stations influence each other. In contrast, a user’s feedback in an IBRM does not influence the user’s other IBRMs, and each IBRM is updated independently. This enables users to flexibly develop IBRMs even for largely different intents.

## 2.3 Multi-faceted user profiles

In the context of recommendations, the term “user profile” is defined as a user model that represents a user’s interests or preferences [21, 22]. Because a user’s interests may be

<sup>3</sup><https://music.youtube.com>

diverse even in a specific domain, methods have been proposed to detect multi-faceted user profiles and recommend items relevant to each profile [30, 51–54], where a user’s profiles are estimated from the interactions between the user and items. In those studies, the number and granularity of the profiles were limited and depended on the method. On the other hand, in our approach a user can create multiple profiles, which correspond to IBRMs, by herself. This enables creation of as many IBRMs as desired, at an arbitrary granularity. Although diversifying recommended items is effective to cover as many intents as possible in recommendation results [55–58], our proposed IBRMs aim to generate recommendation results specialized for a user’s specific intent. Moreover, unlike the studies for solving the problem of mixed intentions of multiple users in a shared account [59–61], we focus on various intents of each user.

Note that it is impractical for a user to create and manage multiple user accounts solely to receive different recommendations according to her intents (e.g., one account for “concentration on work” and another one for “cool songs”), because it is burdensome to switch the accounts every time her intents change. In addition, playlists and favorite songs are not shared between her accounts. Our proposed IBRM architecture does not face such problems.

## 2.4 Music recommender system applications

In the research field of recommender systems, in addition to proposal for recommendation methods and approaches, the main research focus has recently been to implement recommender system applications and then obtain insights from users’ behaviors on those applications [62, 63]. In music recommendations, too, there have been various studies that proposed recommender system applications for displaying explanations [64], providing group recommendations [65], concentrating on work [66], recommending Internet radio stations [67], recommending songs based on sentiment analysis of songs [68], recommending music for an input video [69], and so on. Implementation of such applications is a worthwhile contribution because it can provide guidelines for other researchers and companies on how to practically apply proposed concepts or methods to real recommender systems [70]. Inspired by the importance of implementation, for this paper, we not only proposed the concept of an IBRM but also implemented it and released Kiite that applies the concept.

## 3 Intent-based recommendation model

For this paper, we implemented the proposed concept of an IBRM as a function on Kiite, which is a web service for exploring and discovering music. In this section, we first give an overview of the key functions of Kiite that are related to our concept. We then describe the user operations for developing IBRMs.

### 3.1 Overview of Kiite

Song data on Kiite are routinely collected from *Nico Nico Douga*<sup>4</sup>, which is one of the most popular video sharing services in Japan. On Nico Nico Douga, it is quite common for both amateur and professional musicians to upload songs created with singing voice synthesizer

---

<sup>4</sup><https://www.nicovideo.jp>

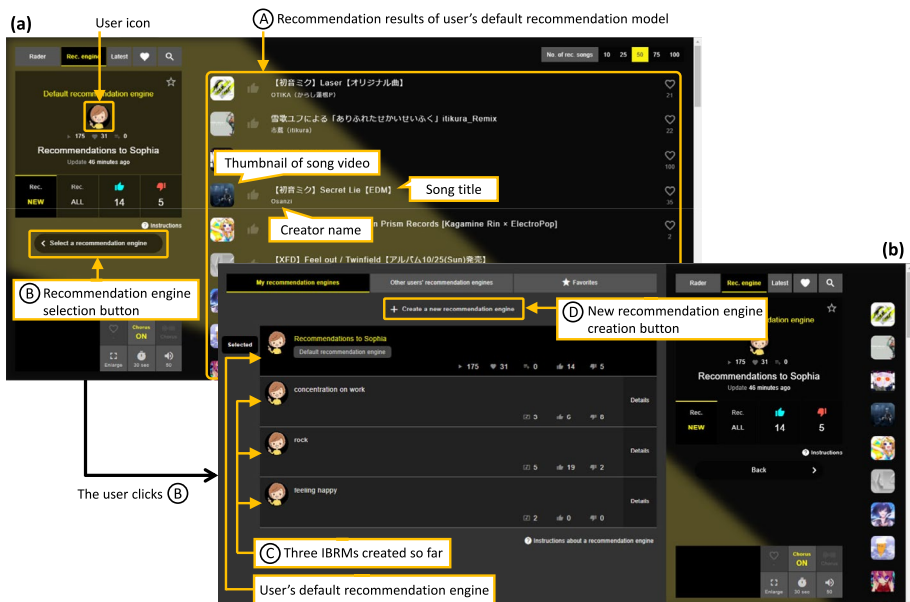
software called *VOCALOID* [71]. As of the end of July 2024, more than 490,000 songs are available on Kiite. When a Kiite user listens to a song, its music video clip is played on Kiite by an embedded video player<sup>5</sup>.

Kiite enables users to effectively find new favorite songs by providing novel functions such as exploration of songs according to the vocal timbre and continuous listening to only the choruses of multiple songs. A registered user can also set her own icon image, add songs to her list of favorite songs, create playlists, and listen to other users' playlists.

## 3.2 Operations for developing IBRMs

### 3.2.1 Initialization

The Kiite interface has a link to a “recommendation engine” page<sup>6</sup>. When a user accesses the page for the first time, it displays song recommendation results that are generated according to her play history and the songs added to her list of favorite songs and her playlists, as shown in Fig. 1(a). Because the recommendation model reflects the user's usual musical preferences, we call it a “default recommendation model.” One default recommendation model is automatically created for a user when she creates a Kiite account.



**Fig. 1** (a) Top page of the “recommendation engine” function. (b) List of the user’s default recommendation model and created IBRMs. (Interfaces are translated into English.)

<sup>5</sup> On Nico Nico Douga, all songs are uploaded as music videos.

<sup>6</sup> On Kiite itself, we use the term “recommendation engine” instead of “recommendation model” so that users can more easily understand the concept. In this paper, we use both terms interchangeably.

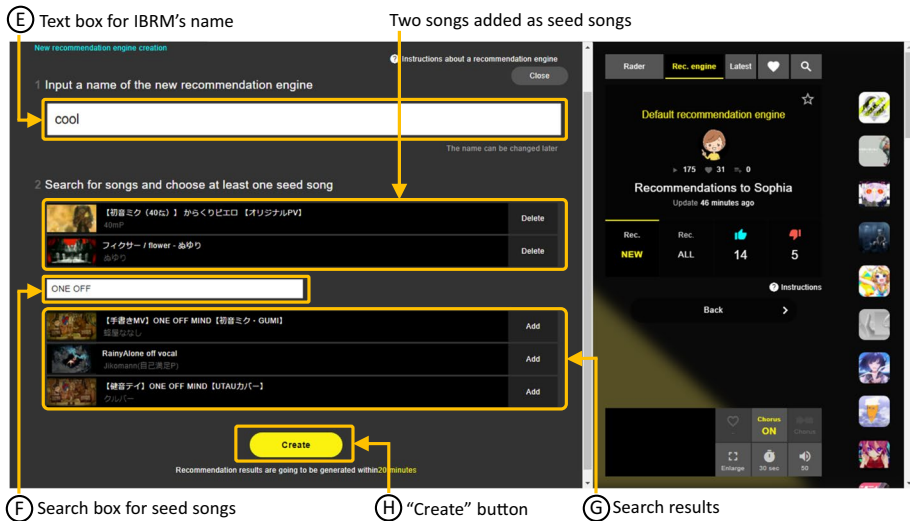


Fig. 2 Interface for creating a new IBRM. (Interface is translated into English.)

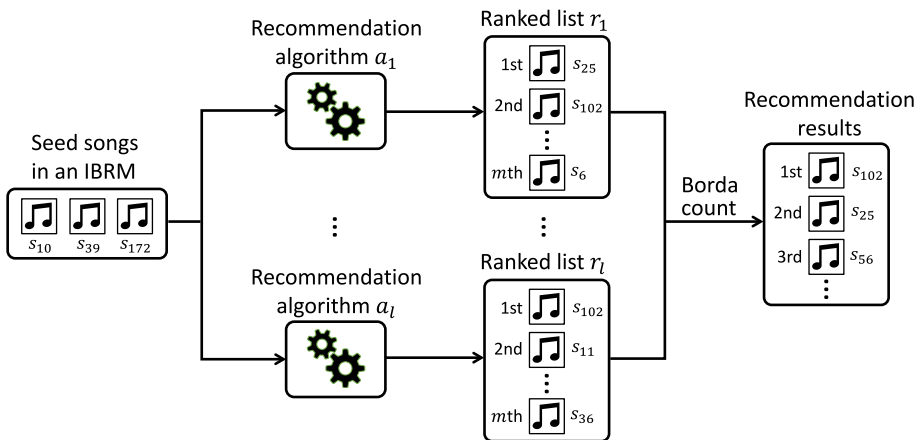


Fig. 3 Process of generating recommendation results for an IBRM

When the “recommendation engine selection” button (Fig. 1Ⓐ) is clicked, a list of available recommendation models, including the default one, is displayed. In the example shown in Fig. 1Ⓒ, the user has already created three IBRMs. To create a new IBRM, she clicks the “new recommendation engine creation” button (Fig. 1Ⓓ). She then inputs the name of the new IBRM (e.g., “cool”), as shown in Fig. 2Ⓔ, and chooses at least one seed song. Seed songs can be searched for via metadata such as titles and creator names (Fig. 2Ⓕ and Ⓖ). Finally, the user completes the initialization process by clicking the “create” button (Fig. 2Ⓖ).

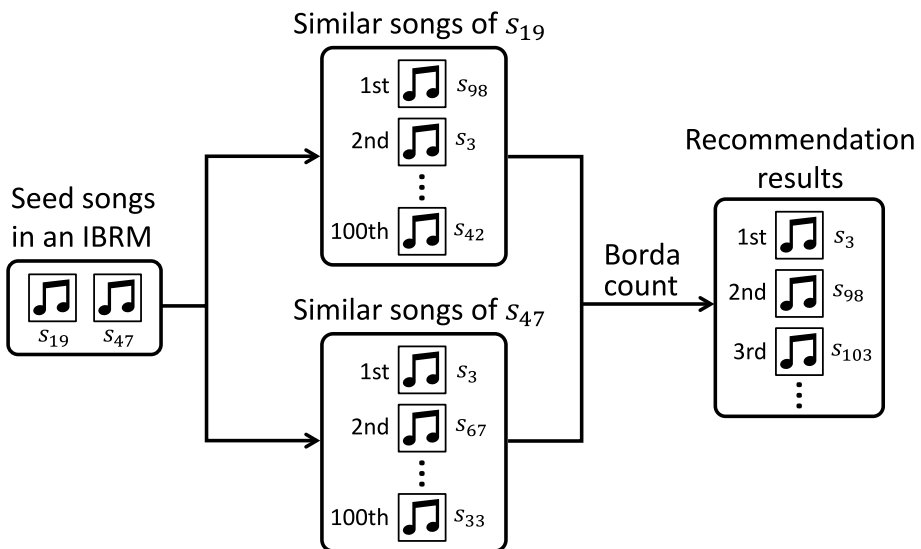
To generate an IBRM’s recommendation results, we use  $l \geq 1$  recommendation algorithms  $A = \{a_1, \dots, a_l\}$ . The input of each algorithm  $a_i$  ( $1 \leq i \leq l$ ) is a set of seed songs.

According to the input,  $a_i$  outputs a ranked list  $r_i$  of the top  $m$  recommended songs. Let  $R$  denote the set of  $l$  recommendation results, i.e.,  $R = \{r_1, \dots, r_l\}$ . Finally, we merge the results in  $R$  by using the Borda count and then display the merged results to the user (Fig. 3). For example, suppose that  $l = 3$  and  $m = 300$  for  $r_1$ ,  $r_2$ , and  $r_3$ . The songs ranked at 1st, 2nd,  $\dots$ , 300th in  $r_i \in R$  obtain the scores of 300, 299,  $\dots$ , 1, respectively. If a song  $s$  is ranked at 3rd in  $r_1$  and 105th in  $r_3$ , and is not included in  $r_2$ , the total score of  $s$  computed by using the Borda count is  $298 + 196 = 494$ . More formally, the score of a song  $s$  is computed by the following equation.

$$f_{\text{Borda}}(s, m, R) = \sum_{1 \leq i \leq l} (m + 1 - \text{rank}(s, r_i)). \quad (1)$$

Here,  $\text{rank}(s, r_i)$  denotes the rank of song  $s$  in the recommendation result  $r_i$ . If  $s$  is not included in  $r_i$ ,  $\text{rank}(s, r_i)$  returns  $m + 1$ . In the same manner, the total scores of each song that appears at least one of  $r_1$ ,  $r_2$ , or  $r_3$  are computed. Then, the songs are ranked in descending order of their total scores in the merged recommendation results. Note that this recommendation process is not our contribution because it is a general approach to merge multiple recommendation results by using the Borda count [72]. Rather, we describe the process here to provide guidelines on how to apply the proposed concept of an IBRM to real recommender systems.

Each of the  $a_i$  can be any recommendation algorithm, from typical ones to recent deep-learning-based ones, if it satisfies the aforementioned input and output requirements. In our implementation for the Kiite service, we use two algorithms (i.e.,  $l = 2$ ). The first one is content-based filtering (Fig. 4), in which the similarity between songs is computed via the Euclidean distance between audio features [73]. A song that is more similar to more seed songs is ranked higher. We used this algorithm to deal with a cold-start problem [74] where only a few seed songs are chosen by a user. Specifically, the recommendation score of a song



**Fig. 4** Process of generating recommendation results through content-based filtering



$s$  is computed based on the Borda count. Let  $S_{seed}$  and  $r_{s'}^{sim}$  denote a set of the seed songs and a ranked list of the top 100 songs in terms of the audio-based similarity with  $s' \in S_{seed}$ , respectively. The score of  $s$  is computed as follows.

$$f_{con}(s, S_{seed}) = \sum_{s' \in S_{seed}} (101 - \text{rank}_{sim}(s, r_{s'}^{sim})), \quad (2)$$

where  $\text{rank}_{sim}(s, r_{s'}^{sim})$  denotes the rank of  $s$  in  $r_{s'}^{sim}$ . It returns 101 if  $s$  is not included in  $r_{s'}^{sim}$ .

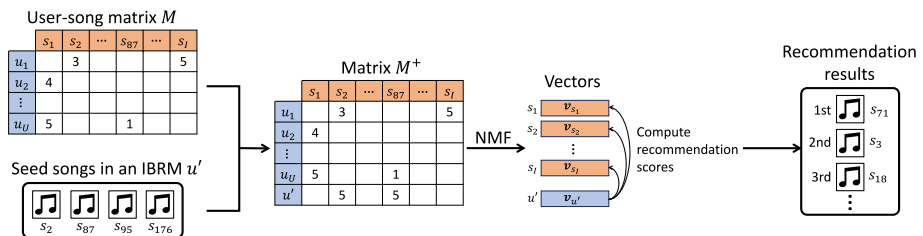
The second algorithm is collaborative filtering (Fig. 5). Here, we first create a user-song matrix  $M \in \mathbb{R}^{U \times I}$  where  $U$  and  $I$  are the numbers of users and songs on Kiite respectively, and an entry  $(x, y)$  is the score of song  $y$  for user  $x$ . The score ranging from 1 to 5 is computed based on the play count of  $y$  by  $x$  and whether  $y$  has been added to  $x$ 's list of favorite songs and playlists. We then regard the initialized IBRM as a user, add it to  $M$ , and create a new matrix  $M^+ \in \mathbb{R}^{(U+1) \times I}$ . In the added row of  $M^+$ , each seed song has a score of 5. Then, we apply Non-negative Matrix Factorization (NMF) [75] to  $M^+$ . In NMF, each user  $u$  and each song  $s$  are represented by  $d$ -dimensional vectors  $\mathbf{v}_u \in \mathbb{R}^d$  and  $\mathbf{v}_s \in \mathbb{R}^d$ , respectively. Finally, we obtain the IBRM's recommendation results by calculating recommendation scores of each song  $s$  for the IBRM  $u'$  based on the inner product:

$$f_{collab}(u', s) = \mathbf{v}_{u'}^T \mathbf{v}_s. \quad (3)$$

In both algorithms, we set  $m = 300$ . Note that the algorithms described above for our service are just examples. In fact, when a particular platform applies the proposed concept of an IBRM, it can use its own recommendation algorithms that are suitable for the platform's domain or service. That is, our contribution is independent from the recommendation algorithms.

### 3.2.2 Update

We generate recommendation results for all users' new IBRMs through a batch operation that is executed every 20 minutes. As a result, within 20 minutes after clicking the "creation" button described above, the user can see the new IBRM's recommendation results by selecting it. In the results, songs are ranked according to their recommendation scores. When a ranked song is clicked, it is played in the video player at the bottom left of the screen, as shown in Fig. 6①. Note that the user can enlarge the video player. When the user finds a song interesting, she can add it to her list of favorite songs or to a playlist. More-



**Fig. 5** Process of generating recommendation results through collaborative filtering



**Fig. 6** Interface for updating a created IBRM. (Interface is translated into English.)

over, by hovering the mouse cursor over the transparent thumbs-up icon displayed next to each song's thumbnail and clicking on either the thumbs-up or thumbs-down icon, the user can give positive or negative feedback (thumbs-up or thumbs-down, as shown in Fig. 6 [J](#)). According to this feedback, the IBRM is updated and new recommendation results are displayed<sup>7</sup>.

Here, note that the feedback is not based on the user's usual musical preferences; rather, it is based on whether the song is relevant to the intent of the IBRM. For example, suppose that a user who usually listens to rock music creates an IBRM named "concentration on work," and a rock song is included in its initial recommendation results. Because the user often listens to rock, the song actually matches her musical preferences. However, if she thinks the rock song is not suitable for the intent of concentration, she can give it a thumbs-down for this IBRM. On the other hand, even if she does not usually listen to jazz, if she thinks a recommended jazz song is suitable for concentration, she can give it a thumbs-up. Thus, as these examples show, the feedback is independent of the user's usual musical preferences. Therefore, a user can give a thumbs-up to a song in one IBRM (e.g., a pop song in a "feeling happy" IBRM) and a thumbs-down to the same song in another IBRM (e.g., the same pop song in a "feeling sad" IBRM). Because each IBRM generates its recommendation results independently, a user can give such conflicting feedback to the song, depending on the IBRM. This novel architecture achieves the desired flexibility for our proposed IBRM.

In our implementation for the Kiite service, we use feedback to generate updated recommendation results by using the aforementioned two algorithms and the Borda count. In both algorithms, songs with thumbs-up are used in the same manner as seed songs. In the content-based filtering, songs that are similar to songs with thumbs-down are removed from the recommendation results; in the collaborative filtering, a score of 1 is assigned to songs with thumbs-down.

<sup>7</sup> Even after initializing an IBRM, the user can add new seed songs, and the IBRM will be updated when they are added.

For a particular IBRM, the interface displays the numbers of thumbs-up and thumbs-down that the user has given, as shown in Fig. 6<sup>Ⓚ</sup>. By clicking the thumbs-up or thumbs-down icon, she can see the list of songs for which she has given such feedback. The model can be repeatedly updated by giving feedback so that the IBRM can correctly reflect the user's intent and generate appropriate recommendations<sup>8</sup>.

## 4 Experiment

We launched the Kiite music recommendation service on the web on Aug. 30, 2019. In this section, by analyzing the user interaction logs stored on Kiite<sup>9</sup>, we first report basic statistics about the usage of IBRMs. We then answer four research questions to reveal key characteristics of IBRMs through the case study.

### 4.1 Basic findings

In this experiment, we analyzed data from users who had created more than one IBRM by Apr. 27, 2022. This gave us 1,116 models created by 417 users; let  $U$  denote the set of these users. The minimum, maximum, and average numbers of IBRMs created by a user were 2, 16, and 2.68, respectively. On average, a user chose 8.87 seed songs to create an IBRM. The total numbers of positive and negative feedback instances (i.e., thumbs-up and thumbs-down) given to the models were 5,521 and 14,215, respectively. However, among the models that received at least one thumbs-up or thumbs-down, 52.74% received more positive feedback than negative feedback. In addition, 56.48% of users gave more positive than negative feedback. Therefore, although positive feedback was slightly more common on average, the type of feedback used to develop an IBRM depended on the user.

As we mentioned in section 3.2.2, one characteristic of an IBRM is to enable a user to give conflicting feedback to a song according to the intents of IBRMs. In our data, we observed that some users behaved that way. For example, a user gave positive feedback to a pop song in an IBRM with a name related to “cheer,” while the same user gave negative feedback to it in an IBRM with a name related to “calm.” Another user gave conflicting feedback to as many as 14 songs between two IBRMs. These findings indicated that users did give feedback on the recommended songs according to their relevance to the IBRM's intent rather than their usual musical preferences.

### 4.2 Difference from playlists

#### 4.2.1 Research question

Although we discussed the difference between our proposed IBRM and a playlist in section 2.2, they have a certain thing in common: they are both named according to a user's

<sup>8</sup>The Terms of Use of the web service state that user interaction logs will be used for research purposes.

<sup>9</sup>In a recommender system evaluation, it is common to adopt a shallow rank such as top 10 or 20. However, the overlap between such shallow recommendation results happens to be small even when they are generated from similar recommendation models. Hence, to evaluate more rigorously, we adopted deep ranks such as top 50, 75, and 100. The same goes for the subsequent experiments.

intent. By comparing the names given to IBRMs with those given to playlists, we can understand how users use them differently, which motivates the following research question.

*RQ1* What are the respective characteristics of the names given by users to IBRMs and playlists?

## 4.2.2 Experimental setup

Because the names of the 1,116 IBRMs provided by the users are too diverse, we categorized them by manually labeling category names for each IBRM. For example, an IBRM named “rock” was labeled with the “genre” category. An IBRM could be labeled with more than one category: for example, an IBRM named “cool jazz” was labeled with both the “mood” and “genre” categories. Next, we collected the 1,379 playlists created by the users in *U* and labeled each one with category names in the same manner as for the IBRMs.

## 4.2.3 Results

Table 1 lists the distributions of the labeled category names. For the IBRMs, “mood” had the highest frequency (22.49%) besides “other.” Because mood information does not usually appear in a song’s title or tags, it is difficult to search for songs that match a user’s desired

**Table 1** Category distributions of the IBRM and playlist names

Category	Examples	No. of IBRMs	No. of playlists
mood	cute, cool, easygoing	251 (22.49%)	96 (6.96%)
genre	rock, jazz, trance	104 (9.32%)	64 (4.64%)
favorite	my favorite songs	57 (5.11%)	114 (8.27%)
creator	BIGHEAD, kz	37 (3.32%)	51 (3.70%)
song title	Mellow Yellow	29 (2.60%)	0
instrument	piano, guitar	12 (1.08%)	12 (0.87%)
situation	before bed, background music for work	9 (0.81%)	35 (2.54%)
time	May 2020, fall 2019	0	319 (23.13%)
selection	30 recommendation songs in 2008	0	259 (18.78%)
VOCALOID	Miku Hatsune	0	111 (8.05%)
tentative	listen later, tentative	0	38 (2.76%)
recommendation	my recommendation	0	28 (2.03%)
novelty	songs found recently	0	26 (1.89%)
other	test, my engine, aaa	633 (56.72%)	496 (33.66%)

mood. In addition, the mood often depends on the user's subjective judgment. For example, the nuances of "cool" could differ from one user to another, and thus, even songs that have a "cool" tag might not always match a user's desired mood. In contrast, a user can develop an IBRM by choosing seed songs and giving feedback so that the model generates recommendation results that match the desired nuances. As a result, many users created IBRMs based on mood. As for "genre," it is relatively easy to search for songs that belong to a desired genre because tags usually include genre information. Nonetheless, "genre" had the second highest frequency among the categories (9.32%). This result also indicates that just filtering songs by a genre name was not enough for users to find desired songs. Instead, they expected that a created IBRM would understand the subtle nuances of their desired songs in a genre from the seed songs and the given feedback.

For the playlists, "time" had the highest frequency (23.13%). This means that playlists were often created to revisit songs that were released during a certain period. Sharing these playlists with other users was also useful because then they did not need to create similar playlists by themselves. The category with the second highest frequency (18.78%) was "selection." Playlists in this category were usually created to recommend the songs in a playlist to other users. Note that our proposed IBRM does not decrease the value of such manually created playlists for recommendation, and vice versa. The reason is that such playlists consist of songs that are chosen according to a creator's unique viewpoint. We believe that we can enrich a user's music listening experiences by offering both playlists created from other users' viewpoints and IBRMs created from the user's own viewpoint.

In summary, the results showed that IBRMs were mainly used to find songs that had desired nuances, while playlists were used to record, share, or recommend songs.

### 4.3 Diversity of IBRMs

#### 4.3.1 Research question

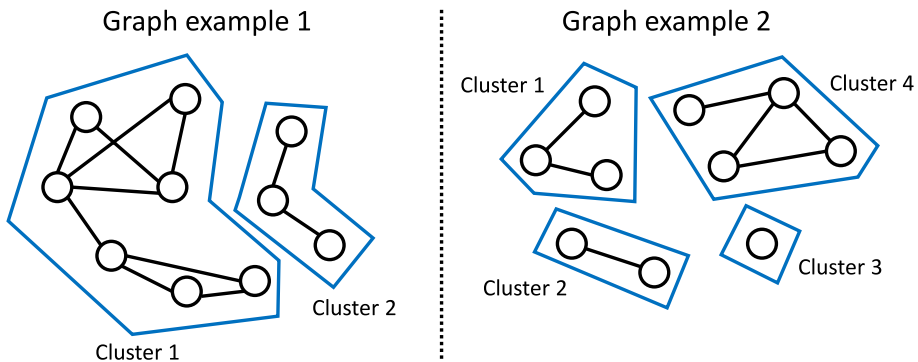
If most of the IBRMs created by users generated similar recommendation results to each other, the users would not need to create models independently. In that case, it might be better if the service platform simply created several (e.g., 10) IBRMs with typical intents and provided them to users. On the other hand, if even models having the same name (e.g., "rock") generated largely different results from one another, it would be more meaningful to enable users to create their own IBRMs. Given these considerations, we address the following two research questions here.

*RQ2 How diverse are the recommendation results generated by users' IBRMs?*

*RQ3 Do even IBRMs having the same name generate different recommendation results from each other?*

#### 4.3.2 Experimental setup

First, to answer **RQ2**, we constructed a graph  $G$  in which each node corresponded to an IBRM (i.e.,  $G$  had 1,116 nodes). When the overlap of the top  $n$  recommendation results generated by two IBRMs was greater than or equal to a threshold  $\theta$ , their corresponding nodes were connected by an undirected edge. Then, we counted the number of connected components in  $G$ . Here, we regarded each connected component as a cluster [76] and assumed that



**Fig. 7** Examples of graphs constructed based on the overlap of recommendation results between IBRMs. Each node represents an IBRM. An edge between two nodes indicates that the corresponding IBRMs share at least  $\theta$  songs in their recommendation results. Both graphs contain 10 nodes, but the left graph has two clusters, while the right graph has four clusters, indicating that the IBRMs in the right graph exhibit greater diversity

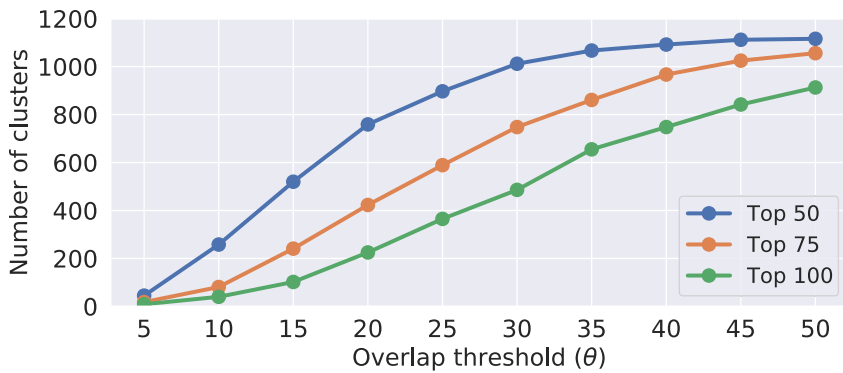
IBRMs in the same cluster generated similar recommendation results. Therefore, the larger the number of clusters was, the more diverse the created IBRMs were, because the IBRMs in different clusters generate dissimilar recommendation results (Fig. 7).

Next, to answer **RQ3**, we collected IBRMs having the same name. In this experiment, we considered the following six names (the number in parentheses represents the number of corresponding IBRMs): “rock” (15), “stylish” (10), “EDM” (8), “up-tempo” (7), “cool” (7), and “cute” (7). Then, we counted the number of overlapping results among the top 100 recommendation results between any two models having the same name.

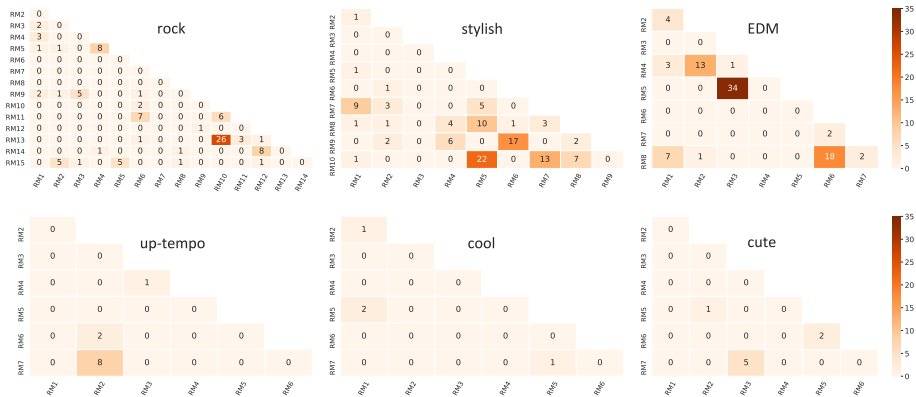
### 4.3.3 Results

Figure 8 shows the results for **RQ2** with  $\theta$  ranging from 5 to 50 for the top 50, 75, or 100 recommendation results. Note that our clustering rule was *loose* because nodes that were not directly connected could belong to the same cluster. Nonetheless, for  $\theta = 20$ , there were 225 clusters for the top 100 recommendation results. This desirable diversity was enabled by our novel architecture. It would be unrealistic for a service platform to prepare and provide such diverse IBRMs in advance. In the cases of the top 50 and 75 results, although the numbers of clusters were smaller than in the case of the top 100, as naturally expected, the created IBRMs were still diverse, which confirmed the significance of enabling users to create their own IBRMs.

Figure 9 shows the results for **RQ3**. For all model names, the most frequent number of overlaps was 0. This does not imply that our recommendation method is inaccurate: if our method was inaccurate and generated recommendations at random, for example, then all the values in the figure would have been close to 0. However, some IBRM pairs had relatively high numbers of overlaps (e.g., 26 for “rock,” 22 for “stylish,” and 34 for “EDM”). This means that our method could generate recommendations when users’ desired nuances were similar. In light of these results, we conclude that even when IBRMs had the same name, they often generated largely different recommendation results from one another.



**Fig. 8** Number of clusters of IBRMs for different overlap thresholds  $\theta$



**Fig. 9** Numbers of overlapping recommendation results between IBRMs having the same name. “RM” stands for “recommendation model.” The value range of the color bars is the same for all the heat maps

## 4.4 Difference from user’s default recommendation model

### 4.4.1 Research question

As mentioned in section 3.2.1, a user’s default recommendation model recommends songs according to her usual musical preferences. Similar to the IBRMs, the DRM is also implemented based on a hybrid recommendation model. In addition, as mentioned in section 3.2.2, when a user develops an IBRM, we assume that she gives feedback on the recommended songs according to their relevance to the IBRM’s intent rather than to her usual musical preferences. If the recommendation results given by a user’s IBRM were largely different from those given by her default recommendation model, it would mean that users actually give feedback in the way we assume. This hypothesis motivates the following research question.

*RQ4 How different are the recommendation results given by a user’s IBRM and the user’s default recommendation model?*

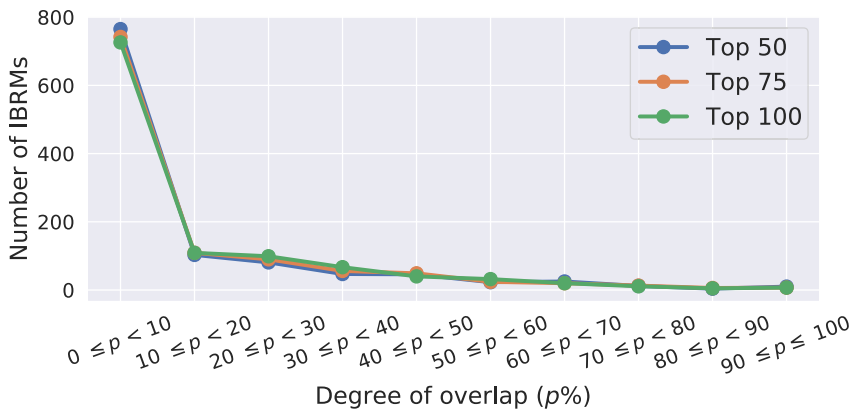


Fig. 10 Overlaps between a user's IBRM and default recommendation model

#### 4.4.2 Experimental setup

Let  $S_{u,n}^d$  denote the set of top  $n$  recommended songs given by the default recommendation model of a user  $u \in U$ . Moreover, let  $C_u$  denote the set of IBRMs created by  $u$ . Similarly to  $S_{u,n}^d$ ,  $S_{u,n}^c$  represents the set of top  $n$  recommended songs given by an IBRM  $c \in C_u$ . To answer **RQ4**, for each IBRM of each user, we computed the percentage of overlapping songs between  $S_{u,n}^d$  and  $S_{u,n}^c$ , i.e.,  $p(u, c, n) = \frac{100 \times |S_{u,n}^d \cap S_{u,n}^c|}{n}$ . For example, when calculating  $p(u, c, n)$  for an IBRM  $c$  created by user  $u$  with  $n = 50$ , if there are 4 overlapping songs between  $S_{u,n}^d$  and  $S_{u,n}^c$ , then  $p(u, c, n) = \frac{100 \times 4}{50} = 8$ .

#### 4.4.3 Results

Figure 10 shows the distribution of  $p(u, c, n)$  over the 1,116 IBRMs for  $n = 50, 75$ , and 100. Regardless of  $n$ , more than half of the IBRMs had  $p(u, c, n)$  values below 10. In particular, for  $n = 100$ , 34.4% of them had  $p(u, c, n) = 0$ . Therefore, we can say that, in general, the recommendation results given by a user's IBRM were largely different from those given by the user's default recommendation model. These results confirm that users tended to give feedback to IBRMs not according to their usual musical preferences but according to the relevance to an IBRM's intent.

### 5 Discussion and conclusion

In this paper, we introduced the concept of intent-based recommendation models (IBRMs), a user-driven framework that allows individuals to create and manage multiple recommendation models according to their specific intents. We implemented this concept in the music web service Kiite and conducted a large-scale analysis of 1,116 IBRMs created by 417 users. Our findings demonstrate that users construct diverse, intent-specific models that differ significantly from playlists and from each other, confirming the value of enabling user-



driven control over recommendation behavior. These results highlight the practical potential of IBRMs as a flexible and personalized approach to recommendation, and they offer a foundation for future research and system design focused on intent-aware personalization. The reusable insights obtained through this case study can be summarized as follows.

- Our proposed IBRM architecture solves two problems that were introduced in section 1. By creating IBRMs that reflect her intents, a user can choose an appropriate IBRM to receive desired recommendations; this solves the first problem of not being able to receive recommendations that meet a user's needs at the moment. Moreover, because each IBRM is created and updated independently, an IBRM's recommendation results are not influenced by other IBRMs. This solves the second problem of recommendations becoming contaminated by a user's temporary interests: instead, the user simply switches IBRMs by herself.
- Our experimental analysis revealed that users utilize IBRMs for purposes clearly distinct from those of playlists, confirming that IBRMs serve as a unique mechanism for expressing and managing intent-specific preferences. Therefore, it is meaningful to enable users to create their own IBRMs. We also found that the vast majority of IBRMs produce significantly different recommendation results from one another, even when they share same names. Furthermore, as for the top 100 recommendation results, over 34% of IBRMs produced zero overlap with the user's default recommendation model, which demonstrates that users give feedback based on intent-specific criteria, not just general musical preferences. These findings underscore the importance of supporting user-driven model creation, as they demonstrate that intent-specific modeling leads to tangible diversity in recommendations. A key strength of this study is that these results were obtained from the long-term, in-the-wild behavior of real users interacting with a deployed web service over a period of more than two years, rather than from a short-term, controlled laboratory setting. This provides strong validity and reinforces the practical significance of the proposed IBRM framework.
- One promising future direction is to generalize the concept of IBRMs beyond the music domain. While this study applied the proposed architecture to a music listening service, the underlying idea is applicable to various multimedia domains such as movie streaming services, video sharing services, and e-book services. Future work could explore how user-IBRM interactions vary across these domains and how domain-specific features can be leveraged to support intent-based personalization more effectively. We believe that the concept can guide other researchers and companies in designing research and systems for recommendation based on user's intents.

Finally, we stress that the goal of this paper was to explore the impact of IBRMs in terms of user interaction logs on the web service. Therefore, it was beyond the scope of this paper to compare the recommendation accuracy between the IBRM and state-of-the-art recommendation models. Another important future direction is to enhance the flexibility of IBRM creation by supporting derivative and hybrid IBRMs. For example, we could enable users to copy an existing IBRM created by themselves or by other users, and then customize it through feedback. A more ambitious direction is to support combining two or more IBRMs into a new hybrid model. While copying is relatively straightforward, merging existing IBRMs involves challenges such as preserving the consistency of user feedback

and reconciling potentially conflicting intents. Although most users passively receive recommendations on general music streaming services, our proposed IBRM enables users to actively interact with recommendation models and experience the usefulness and satisfaction of developing them. We believe such IBRM's characteristics can diversify and enrich users' music listening experiences.

**Acknowledgements** We thank Crypton Future Media, INC. for developing Kiite with us and Nico Nico Douga for initially tacitly (later explicitly) encouraging us to develop Kiite. We also would like to extend our appreciation to users of Kiite and Nico Nico Douga, creators of VOCALOID songs, creators of popularity rankings of VOCALOID songs, and all people who have created, supported, and enjoyed the VOCALOID culture.

**Funding** This work was supported in part by JST CREST Grant Number JPMJCR20D4 and JST ACCEL Grant Number JPMJAC1602, Japan.

**Availability of Data and Materials** Although Kiite's Terms of Use allow us to utilize the data, they do not permit us to share it with third parties, so the data used in this paper will not be shared.

## Declarations

**Competing Interests** The authors declare that they have no competing interests.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References


1. Lonsdale AJ, North AC (2011) Why do we listen to music? A uses and gratifications analysis. *Br J Psychol* 102(1):108–134
2. Randall WM, Rickard NS (2017) Reasons for personal music listening: A mobile experience sampling study of emotional outcomes. *Psychol Music* 45(4):479–495
3. North AC, Hargreaves DJ, Hargreaves JJ (2004) Uses of music in everyday life. *Music Percept Interdiscip J* 22(1):41–77
4. Haake AB (2011) Individual music listening in workplace settings: An exploratory survey of offices in the UK. *Music Sci* 15(1):107–129
5. Lee JH, Waterman NM (2012) Understanding user requirements for music information services. In: *Proceedings of the 13th international society for music information retrieval conference. ISMIR 2012*, pp 253–258
6. Lee JH, Wishkoski R, Aase L, Meas P, Hubbles C (2017) Understanding users of cloud music services: Selection factors, management and access behavior, and perceptions. *J Assoc Inf Sci Technol* 68(5):1186–1200
7. Hoashi K, Matsumoto K, Inoue N (2017) Personalization of user profiles for content-based music retrieval based on relevance feedback. In: *Proceedings of the 11th ACM international conference on multimedia. MM 2003*, pp 110–119
8. Yoshii K, Goto M, Komatani K, Ogata T, Okuno HG (2006) Hybrid collaborative and content-based music recommendation using probabilistic model with latent user preferences. In: *Proceedings of the 7th international conference on music information retrieval. ISMIR 2006*, pp 296–301

9. Flexer A, Schnitzer D, Gasser M, Widmer G (2008) Playlist generation using start and end songs. In: Proceedings of the 9th international conference on music information retrieval. ISMIR 2008, pp 173–178
10. Bu J, Tan S, Chen C, Wang C, Wu H, Zhang L, He X (2010) Music recommendation by unified hypergraph: Combining social media information and music content. In: Proceedings of the 18th ACM international conference on multimedia. MM 2010, pp 391–400
11. Xing Z, Wang X, Wang Y (2014) Enhancing collaborative filtering music recommendation by balancing exploration and exploitation. In: Proceedings of the 15th international society for music information retrieval conference. ISMIR 2014, pp 445–450
12. Vall A, Skowron M, Knees P, Schedl M (2015) Improving music recommendations with a weighted factorization of the tagging activity. In: Proceedings of the 16th international society for music information retrieval conference. ISMIR 2015, pp 65–71
13. Gouvert O, Oberlin T, Févotte C (2018) Matrix co-factorization for cold-start recommendation. In: Proceedings of the 19th international society for music information retrieval conference. ISMIR 2018, pp 792–798
14. Cohen WW, Fan W (2000) Web-collaborative filtering: Recommending music by crawling the web. *Comput Netw* 33(1):685–698
15. Gasser M, Flexer A (2009) FM4 Soundpark: Audio-based music recommendation in everyday use. In: Proceedings of the 6th sound and music computing conference. SMC 2009, pp 23–25
16. Flexer A, Stevens J (2018) Mutual proximity graphs for improved reachability in music recommendation. *J New Music Res* 47(1):17–28
17. Wang X, Wang Y (2014) Improving content-based and hybrid music recommendation using deep learning. In: Proceedings of the 22nd ACM international conference on multimedia. MM 2014, pp 627–636
18. Liang D, Zhan M, Ellis DPW (2015) Content-aware collaborative music recommendation using pre-trained neural networks. In: Proceedings of the 16th international society for music information retrieval conference. ISMIR 2015, pp 295–301
19. Schedl M (2019) Deep learning in music recommendation systems. *Front Appl Math Stat* 5(4):1–9
20. Jin X, Zhou W, Wang J, XU D, Zheng Y An order-complexity aesthetic assessment model for aesthetic-aware music recommendation. In: Proceedings of the 31st ACM international conference on multimedia. MM 2023, pp 6938–6947 (2023)
21. Schiaffino S, Amandi A (2009) Intelligent user profiling. *Artificial Intelligence: An International Perspective*. Springer, Berlin, Heidelberg, pp 193–216
22. Gao M, Liu K, Wu Z (2010) Personalisation in web computing and informatics: Theories, techniques, applications, and future research. *Inf Syst Front* 12(5):607–629
23. Rho S, Han B-j, Hwang E (2009) SVR-based music mood classification and context-based music recommendation. In: Proceedings of the 17th ACM international conference on multimedia. MM 2009, pp 713–716
24. Hariri N, Mobasher B, Burke R (2012) Context-aware music recommendation based on latent topic sequential patterns. In: Proceedings of the 6th ACM conference on recommender systems. RecSys 2012, pp 131–138
25. Wang X, Rosenblum D, Wang Y (2012) Context-aware mobile music recommendation for daily activities. In: Proceedings of the 20th ACM international conference on multimedia. MM 2012, pp 99–108
26. Chen C-M, Tsai M-F, Liu J-Y, Yang Y-H (2013) Using emotional context from article for contextual music recommendation. In: Proceedings of the 21st ACM international conference on multimedia. MM 2013, pp 649–652
27. Cheng Z, Shen J (2014) Just-for-Me: An adaptive personalization system for location-aware social music recommendation. In: Proceedings of the 4th ACM international conference on multimedia retrieval. ICMR 2014, pp 185–192
28. Adomavicius G, Tuzhilin A (2015) Context-aware recommender systems. *Recommender systems handbook*. Springer, Boston, MA, pp 191–226
29. Pichl M, Zangerle E, Specht G (2015) Towards a context-aware music recommendation approach: What is hidden in the playlist name? In: Proceedings of the 2015 IEEE international conference on data mining workshop. ICDMW 2015, pp 1360–1365
30. Volokhin S, Agichtein E (2018) Towards intent-aware contextual music recommendation: Initial experiments. In: Proceedings of the 41st international ACM SIGIR conference on research & development in information retrieval. SIGIR 2018, pp 1045–1048
31. Casillo M, Gupta BB, Lombardi M, Lorusso A, Santaniello D, Valentino C (2022) Context aware recommender systems: A novel approach based on matrix factorization and contextual bias. *Electronics* 11(7):1–19

32. Colombo-Mendoza LO, Valencia-García R, Rodríguez-González A, Alor-Hernández G, Samper-Zapater JJ (2015) RecomMetz: A context-aware knowledge-based mobile recommender system for movie showtimes. *Expert Syst Appl* 42(3):1202–1222
33. Yang D, Zhang D, Yu Z, Wang Z (2013) A sentiment-enhanced personalized location recommendation system. In: *Proceedings of the 24th ACM conference on hypertext and social media*. HT 2013, pp 119–128
34. Wu J, He X, Wang X, Wang Q, Chen W, Lian J, Xie X (2022) Graph convolution machine for context-aware recommender system. *Front Comput Sci* 16(6):166614
35. Afzal I, Yilmazel B, Kaleli C (2024) An approach for multi-context-aware multi-criteria recommender systems based on deep learning. *IEEE Access* 12:99936–99948
36. Shi Y, Karatzoglou A, Baltrunas L, Larson M, Hanjalic A, Oliver N (2012) TFMAP: Optimizing MAP for top-n context-aware recommendation. *SIGIR* 2012:155–164
37. Villegas NM, Sánchez C, Díaz-Cely J, Tamura G (2018) Characterizing context-aware recommender systems: A systematic literature review. *Knowl-Based Syst* 140(C):173–200
38. Bonnin G, Jannach D (2014) Automated generation of music playlists: Survey and experiments. *ACM Comput Surv* 47(2):26–12635
39. Dias R, Gonçalves D, Fonseca MJ (2017) From manual to assisted playlist creation: A survey. *Multimed Tools Appl* 76(12):14375–14403
40. Jannach D, Lerche L, Kamehkhosh I (2015) Beyond “hitting the hits”: Generating coherent music playlist continuations with the right tracks. In: *Proceedings of the 9th ACM conference on recommender systems*. RecSys 2015, pp 187–194
41. Nakano T, Kato J, Hamasaki M, Goto M (2016) PlaylistPlayer: An interface using multiple criteria to change the playback order of a music playlist. In: *Proceedings of the 21st international conference on intelligent user interfaces*. IUI 2016, pp 186–190
42. Shih S, Chi H (2018) Automatic, personalized, and flexible playlist generation using reinforcement learning. In: *Proceedings of the 19th international society for music information retrieval conference*. ISMIR 2018, pp 168–174
43. Kim J, Won M, Liem CCS, Hanjalic A (2018) Towards seed-free music playlist generation: Enhancing collaborative filtering with playlist title information. In: *Proceedings of the ACM recommender systems challenge 2018*. RecSys Challenge 2018, pp 1–6
44. Sakurai K, Togo R, Ogawa T, Haseyama M (2022) Controllable music playlist generation based on knowledge graph and reinforcement learning. *Sensors* 22(10):3722
45. Tomasi F, Cauteruccio J, Kanoria S, Ciosek K, Rinaldi M, Dai Z (2023) Automatic music playlist generation via simulation-based reinforcement learning. In: *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*. KDD 2023, pp 4948–4957
46. Pampalk E, Pohle T, Widmer G (2005) Dynamic playlist generation based on skipping behavior. In: *Proceedings of the 6th international conference on music information retrieval*. ISMIR 2005, pp 634–637
47. Pontello LF, Holanda PHF, Guilherme B, JaPV Cardoso, Goussevskaia O, Silva APCD (2017) Mixtape: Using real-time user feedback to navigate large media collections. *ACM Trans Multimed Comput Commun Appl* 13(4):1–22
48. Kamalzadeh M, Kralj C, Möller T, Sedlmair M (2016) TagFlip: Active mobile music discovery with social tags. *IUI '16*, pp 19–30
49. Baur D, Hering B, Boring S, Butz A (2011) Who needs interaction anyway: Exploring mobile playlist creation from manual to automatic. *IUI* 2011:291–294
50. Kamehkhosh I, Bonnin G, Jannach D (2020) Effects of recommendations on the playlist creation behavior of users. *User Model User-Adap* 30(2):285–322
51. Narducci F, Musto C, Semeraro G, Lops P, Gemmis M (2013) Exploiting big data for enhanced representations in content-based recommender systems. In: *Proceedings of the 14th international conference on electronic commerce and web technologies*. EC-Web 2013, pp 182–193
52. Mehrotra R, Lalmas M, Kenney D, Lim-Meng T, Hashemian G (2019) Jointly leveraging intent and interaction signals to predict user satisfaction with slate recommendations. In: *Proceedings of the world wide web conference 2019*. WWW 2019, pp 1256–1267
53. de Campos LM, Fernández-Luna JM, Huete JF, Redondo-Expósito L (2020) Automatic construction of multi-faceted user profiles using text clustering and its application to expert recommendation and filtering problems. *Knowl-Based Syst* 190:1–18
54. Zheng Z, Hu X, Gao S, Zhu H, Xiong H (2024) MIRROR: A multi-view reciprocal recommender system for online recruitment. In: *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*. SIGIR 2024, pp 543–552
55. Agrawal R, Gollapudi S, Halverson A, Ieong S (2009) Diversifying search results. In: *Proceedings of the 2nd ACM international conference on web search and data mining*. WSDM 2009, pp 5–14

56. Noia TD, Ostuni VC, Rosati J, Tomeo P, Di Sciascio E (2014) An analysis of users' propensity toward diversity in recommendations. In: Proceedings of the 8th ACM conference on recommender systems. RecSys 2014, pp 285–288
57. Kaya M, Bridge D (2018) Accurate and diverse recommendations using item-based subprofiles. In: Proceedings of the 31st international florida artificial intelligence research society conference. FLAIRS-31, pp 462–467
58. Yang L, Wang S, Tao Y, Sun J, Liu X, Yu PS, Wang T (2023) DGRec: Graph neural network for recommendation with diversified embedding generation. In: Proceedings of the 16th ACM international conference on web search and data mining. WSDM 2023, pp 661–669
59. Verstrepen K, Goethals B (2015) Top-N recommendation for shared accounts. In: Proceedings of the 9th ACM conference on recommender systems. RecSys 2015, pp 59–66
60. Jiang J-Y, Li C-T, Chen Y, Wang W (2018) Identifying users behind shared accounts in online streaming services. In: Proceedings of the 41st international ACM SIGIR conference on research & development in information retrieval. SIGIR 2018, pp 65–74
61. Qin J, Zhu J, Liu Y, Gao J, Ying J, Liu C, Wang D, Feng J, Deng C, Wang X, Jiang J, Liu C, Yu Y, Zeng H, Zhang W (2023) Learning to distinguish multi-user coupling behaviors for TV recommendation. In: Proceedings of the 16th ACM international conference on web search and data mining. WSDM 2023, pp 204–212
62. Bobadilla J, Ortega F, Hernando A, Gutiérrez A (2013) Recommender systems survey. Knowl-Based Syst 46:109–132
63. Park DH, Kim HK, Choi IY, Kim JK (2012) A literature review and classification of recommender systems research. Expert Syst Appl 39(11):10059–10072
64. Millecamp M, Htun NN, Conati C, Verbert K (2019) To explain or not to explain: The effects of personal characteristics when explaining music recommendations. In: Proceedings of the 24th international conference on intelligent user interfaces. IUI 2019, pp 397–407
65. Htun NN, Lecluse E, Verbert K (2021) Perception of fairness in group music recommender systems. In: Proceedings of the 26th international conference on intelligent user interfaces. IUI 2021, pp 302–306
66. Yakura H, Nakano T, Goto M (2018) FocusMusicRecommender: A system for recommending music to listen to while working. In: Proceedings of the 23rd international conference on intelligent user interfaces. IUI 2018, pp 7–17
67. Grant M, Ekanayake A, Turnbull D (2013) MeUse: Recommending internet radio stations. In: Proceedings of the 14th international society for music information retrieval conference. ISMIR 2013, pp 281–286
68. Sarin E, Vashishtha S, Megha, Kaur S (2022) SentiSpotMusic: a music recommendation system based on sentiment analysis. In: Proceedings of the 4th international conference on recent trends in computer science and technology. ICRTCST 2022, pp 373–378
69. Dong Z, Liu X, Chen B, Polak P, Zhang P (2024) MuseChat: A conversational music recommendation system for videos. In: Proceedings of the 2024 IEEE/CVF conference on computer vision and pattern recognition. CVPR 2024, pp 12775–12785
70. Lu J, Wu D, Mao M, Wang W, Zhang G (2015) Recommender system application developments: A survey. Decis Support Syst 74:12–32
71. Kenmochi H, Ohshita H (2007) VOCALOID - commercial singing synthesizer based on sample concatenation. In: Proceedings of the 8th annual conference of the international speech communication association. INTERSPEECH 2007, pp 4009–4010
72. Kompan M, Bielikova M (2014) Group recommendations: Survey and perspectives. Comput Inform 33(2):446–476
73. Cramer J, Wu H-H, Salamon J, Bello JP (2019) Look, listen, and learn more: Design choices for deep audio embeddings. In: Proceedings of the 2019 IEEE international conference on acoustics, speech and signal processing. ICASSP 2019, pp 3852–3856
74. Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17(6):734–749
75. Lee DD, Seung HS (1999) Learning the parts of objects by non-negative matrix factorization. Nature 401:788–791
76. Conover M, Ratkiewicz J, Francisco M, Gonçalves B, Menczer F, Flammini A (2011) Political polarization on Twitter. In: Proceedings of the 15th international AAAI conference on weblogs and social media. ICWSM 2011, pp 89–96

## Authors and Affiliations

**Kosetsu Tsukuda<sup>1</sup>  · Keisuke Ishida<sup>1</sup> · Kento Watanabe<sup>1</sup> · Masahiro Hamasaki<sup>1</sup> · Masataka Goto<sup>1</sup>**

✉ Kosetsu Tsukuda  
k.tsukuda@aist.go.jp

Keisuke Ishida  
ksuke-ishida@aist.go.jp

Kento Watanabe  
kento.watanabe@aist.go.jp

Masahiro Hamasaki  
masahiro.hamasaki@aist.go.jp

Masataka Goto  
m.goto@aist.go.jp

<sup>1</sup> National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan