Advances in Next-Track Music Recommendation

Dissertation

zur Erlangung des Grades eines

D o k t o r s d e r N a t u r w i s s e n s c h a f t e n

der Technischen Universität Dortmund
an der Fakultät für Informatik

von

Iman Kamehkhosh

Dortmund

2017

Tag der mündlichen Prüfung: 26.02.2018

Dekan: Prof. Dr.-Ing. Gernot A. Fink

Gutachter:
Prof. Dr. Dietmar Jannach
Prof. Dr. Günter Rudolph

Abstract

Technological advances in the music industry have dramatically changed how people access and listen to music. Today, online music stores and streaming services offer easy and immediate means to buy or listen to a huge number of songs. One traditional way to find interesting items in such cases when a vast amount of choices are available is to ask others for recommendations. Music providers utilize correspondingly *music recommender systems* as a software solution to the problem of *music overload* to provide a better *user experience* for their customers. At the same time, an enhanced user experience can lead to higher customer retention and higher business value for music providers.

Different types of music recommendations can be found on today's music platforms, such as Spotify or Deezer. Providing a list of currently trending music, finding similar tracks to the user's favorite ones, helping users discover new artists, or recommending curated playlists for a certain mood (e.g., romantic) or activity (e.g., driving) are examples of common music recommendation scenarios. "Next-track music recommendation" is a specific form of music recommendation that relies mainly on the user's recently played tracks to create a list of tracks to be played next. Next-track music recommendations are used, for instance, to support users during playlist creation or to provide personalized radio stations. A particular challenge in this context is that the recommended tracks should not only match the general taste of the listener but should also match the characteristics of the most recently played tracks.

This *thesis by publication* focuses on the next-track music recommendation problem and explores some challenges and questions that have not been addressed in previous research. In the first part of this thesis, various next-track music recommendation algorithms as well as approaches to evaluate them from the research literature are reviewed. The recommendation techniques are categorized into the four groups of content-based filtering, collaborative filtering, co-occurrence-based,

and sequence-aware algorithms. Moreover, a number of challenges, such as personalizing next-track music recommendations and generating recommendations that are coherent with the user's listening history are discussed. Furthermore, some common approaches in the literature to determine relevant quality criteria for next-track music recommendations and to evaluate the quality of such recommendations are presented.

The second part of the thesis contains a selection of the author's publications on next-track music recommendation as follows.

1. The results of comprehensive analyses of the musical characteristics of manually created playlists for music recommendation;
2. the results of a multi-dimensional comparison of different academic and commercial next-track recommending techniques;
3. the results of a multi-faceted comparison of different session-based recommenders, among others, for the next-track music recommendation problem with respect to their accuracy, popularity bias, catalog coverage as well as computational complexity;
4. a two-phase approach to recommend accurate next-track recommendations that also match the characteristics of the most recent listening history;
5. a personalization approach based on multi-dimensional user models that are extracted from the users' long-term preferences;
6. a user study with the aim of determining the quality perception of next-track music recommendations generated by different algorithms.

Contents

1 Introduction

"Without music, life would be a mistake."

Friedrich Nietzsche, Twilight of the Idols

1.1 Motivation

No one knows how the story of music began but there is evidence of our caveman ancestors making flutes and whistles out of animal bones. Through its ongoing progress, music has become a massive global phenomenon. Today, it is hard for us to imagine a time – in the days before music could be recorded – when people could go weeks without hearing any music at all [Goo13].

The invention of recording and playback devices in the late 19^{th} century changed the music listening from a "live-only" event in concert halls or churches to a more intimate experience. Portable cassette players introduced another turning point in how people listened to music by making music mobile. With the creation of compact disc and the invention of the MP3 format, music entered the digital era. The launch of the first websites for downloading and sharing music, e.g., eMusic[1] and Napster[2], changed how people accessed music yet again.

Almost a century after the first radio music was broadcast in 1906, Last.fm[3] launched the first ad-funded Internet radio platform offering personalized music. In recent years, music streaming has become the dominant way of consuming music and the most profitable source in the music industry, as in the first half of 2017, 75% of those who consumed music used streaming services and 62% of the U.S. music industry revenues came from streaming [Fri16].

[1] https://www.emusic.com/
[2] https://www.napster.com/
[3] https://www.last.fm/

A remarkable impact of digitalization and the Internet on music is the ease of immediate access to a huge number of songs. Major music streaming services, e.g., Spotify[4], and online music stores like iTunes[5] have over 30 million songs [Pre17], adding thousands of new songs every month. All this music can be accessed anytime through an online application or an app on a mobile device. Besides its potential for discovering new songs and artists, this vast amount of data can easily lead to *information anxiety* [Wur90] for music consumers and make it difficult for them to come to a decision.

Numerous on-demand streaming services, such as Spotify, Last.fm, Pandora[6], Deezer[7], Google Play Music[8], and Apple Music[9], with almost similar music libraries, prices, platforms, and license models, try to differentiate themselves by how well they help listeners go through all that choices [Hog15] and by offering a better user experience. In this regard, *music recommender systems* were introduced to tackle the problem of *music overload* and are supposed to provide listeners with *interesting* and *novel* recommendations from available music collections.

One of the first music recommender systems was an email-based service called *Ringo* [Sha+95]. Users should first rate a list of artists and state how much they like to listen to them. Based on that, users could ask Ringo to suggest new artists or albums that they will like or dislike and to also get a prediction of how much they will like each one. Technological progress in the music domain together with changes in our listening habits have, however, opened new opportunities for other recommendation scenarios. For instance, the *discovery* feature of Spotify provides users with *personalized* recommendations through *weekly playlists* and a playlist of *newly released tracks* that might be interesting to them. Furthermore, *non-personalized* recommendations of trending tracks and curated playlists, are common on most music platforms.

One specific recommendation scenario for today's music recommender system is to recommend a track that could be played next, e.g., when someone is listening to a radio station, or to recommend an additional track that could be included in the playlist a user is creating. The problem of determining suitable tracks to be played next or to be added to a playlist based on a set of *seed tracks* is referred to as the "next-track music recommendation" or the "playlist continuation" problem in the research literature. It is the main focus of this thesis.

[4] https://www.spotify.com
[5] https://www.apple.com/itunes/
[6] https://www.pandora.com/
[7] https://www.deezer.com
[8] https://play.google.com/music/listen
[9] https://www.apple.com/music/

The remainder of this chapter is mainly dedicated to the music recommendation problem in general (Section 1.2) and a short description of next-track music recommendation as a specific form of music recommendation (Section 1.3). Next, the research questions that are addressed in this thesis are discussed (Section 1.4). At the end of this chapter, an outline of the remainder of this thesis (Section 1.5) and a list of the publications that are included in it are presented (Section 1.6).

1.2 The Music Recommendation Problem

This section starts with a general characterization of the music recommendation problem. Next, different music recommendation scenarios along with specific challenges of music recommendation are presented.

1.2.1 Characterization of the Music Recommendation Problem

Like in the general field of recommender systems [Voz+03], the basic *entities* of a music recommender system are (1) the *user*, i.e., the *music listener* or *consumer* who interacts with a streaming service, a music player, or an online music store and (2) the *item*, i.e., the *music item*, such as track, artist, or playlist.

Figure 1.1 illustrates the components of a music recommender system. Schedl et al. [Sch+17c] categorize the *input* of a recommender system into two groups of *user inputs* and *item inputs*. The user inputs consist of (i) the *listener's background* like her demographic data, music experience and preferences, (ii) the *listener's intent*, e.g., changing mood or finding a motivation, and (iii) the *listener's context* like her mood, the time of the day or the current social environment of the listener. Schedl et al. [Sch+17c] also introduce three components for the item inputs. (i) The *content of music* that is the musical characteristics of a track like its rhythm, timbre, or melody, (ii) the *purpose of music*, i.e., the intention of the author of music that could be political, social, etc., and (iii) the *context of music* that can be determined, for example, through cover artwork, video clips, or social tags.

The *goal* of music recommender systems is then to predict the preferences of a user for music items, using the input data, and to generate recommendations based on these predictions. The generated music recommendations could be either about novel tracks, artists, or albums that are new to the user, or items that the user already knows but might have forgotten about them or that might match her current context or listening intention.

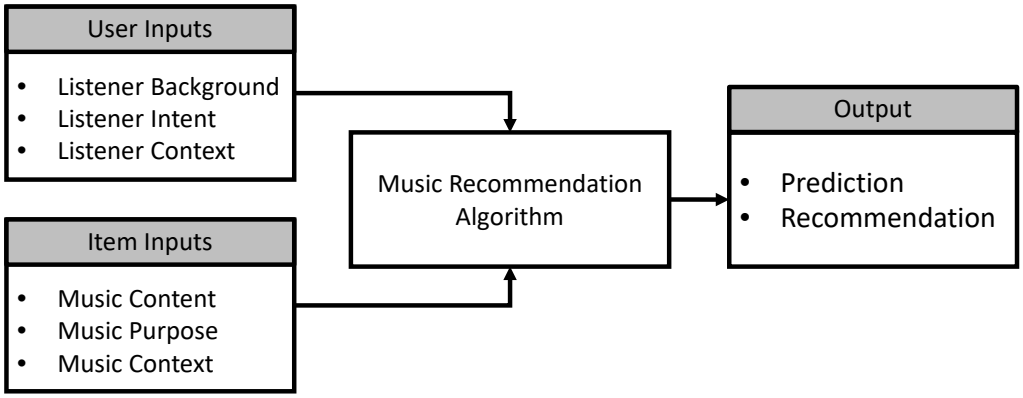


Figure 1.1: Components of a music recommender system.

The *output* of a music recommender system can be either a *prediction* or a *recommendation*.

- A *prediction* expresses the estimated preference of a user for a music item in the catalog. The scale of the predicted value corresponds to the input of the recommendation algorithm. If user preferences are collected in the form of numerical values, for example from a five-star rating system, the predicted value will also be a numerical value between 0 and 5. Similarly, if a recommendation algorithm relies on binary user feedback like thumbs ups/downs, the prediction will be a binary value.
- A *recommendation* is a list of items that a user will probably like. One common way to create the recommendation list is to use the prediction values, for example, by including top n items with the highest predicted values.

Note that in many traditional recommendation scenarios, it makes sense not to recommend items that the user has already consumed [Lin+03]. For instance, recommending a digital camera, only a few days after the user has bought one, will probably not be a helpful recommendation. However, some recent works have explored the value of *reminding* users of items that they have inspected in the past [Jan+17b] or *repeating* favorite tracks of a user, e.g., when it is assumed that the user prefers to listen to known tracks [Jan+17a]. In latter scenarios, a recommender system has to decide *whether or not* to recommend known items and *which* items to recommend for *re-consumption*, see Section 2.6.1.

1.2.2 Music Recommendation Scenarios

In addition to different types of music recommendations that can be found on music platforms and apps, some music recommendation scenarios have been explored in the literature in the past. In the following, an overview of these scenarios is presented.

Recommend a trending list. A common recommendation scenario on almost all music-related sites and applications is to provide users with lists (or charts) of currently popular music items. These lists are usually composed of the top-n globally or locally most played tracks or artists, most bought or downloaded albums, or top hits of a specific genre. The non-personalized approach of recommending the most popular items or the greatest hits to everyone is often used in the music research literature as a baseline to investigate the performance of the proposed algorithms, as done, e.g., in [Bon+14; Pál+14; Che+16]; or [Vas+16].

Recommend new music. This type of recommendation includes newly released music and can include personalized and non-personalized recommendations. Spotify's "Release Radar" or Apple Music's "My New Music Mix" are examples of features that implement such a scenario.

Recommend similar items. A list of similar tracks to the currently playing track or similar artists to the user's favorite artist can also be found on many music services. The similarity of tracks is often determined through audio content analysis of features such as tempo, timbre, or pitch and is mainly addressed in the Music Information Retrieval (MIR) literature, see, e.g., [Cas+08] or [Mül15]. For artist similarity, collaborative filtering approaches and text analysis of user-generated content and lyrics have been applied in the literature [Kne+13].

Discover novel items. In this scenario, personalized music recommendations help users discover unknown and interesting music items such as artists, tracks, albums, concerts, video clips, or radio stations [Mol+12] that match their general preferences. The "Discover Weekly" feature of Spotify [Joh+15], the "Flow" feature of Deezer, and personalized album recommendations on the iTunes store or Amazon.com are commercial implementations of this scenario.

Recommend music playlists. This recommendation scenario consists in recommending a list of tracks that is usually meant to be listened to in a specific order determined by the playlist creator [Lon+16]. This can be

- recommending non-personalized editorial or curatorial playlists like “Staff Picks” on 8tracks.com;
- recommending hand-made playlists to users based on their taste like the “Playlists you may like” recommendations of Deezer;
- recommending a curated playlist based, e.g., on the time of the day, day of the week, or different moods and activities that can be either non-personalized like the “Genres & Moods” playlists of Spotify or personalized like “My Mixes” of Apple Music;
- recommending a personalized playlist from user’s favorite music like Spotify’s “Daily Mixes”.

Support playlist construction. Some music recommenders, such as Spotify or Pandora support users to create new playlists. In this scenario, the recommendation tool uses the title of the playlist or the set of tracks that are already in the playlist to generate a list of tracks, from which the user can select the next track to be added to the playlist. This process can also be personalized based on, e.g., the user’s past preferences [Jan+17a].

Create radio stations. Making music recommendations for radio stations is another scenario in this domain that can, again, be either personalized or non-personalized. In contrast to the *playlist-recommendation* scenario, in which recommendations are presented “in batch”, i.e., as a whole list, radio station recommendations are sequential and usually presented one after the other [Sch+17c]. One application area of such recommendations are *broadcasting radios*, which often use playlists made by disc jockeys containing popular tracks and targeting certain audiences [Eke+00]. In this case, users have no interaction with the system and cannot influence the recommendations. A more recent application area for such recommendations are *virtual radio stations*, in which a virtually endless playlist given a seed track or artist is created [Cli06; Moe+10; Jyl+12]. The process of creating virtual radio stations can be personalized based on the music taste and the immediate feedbacks, e.g., “thumbs-ups/downs” and “skip” actions of the user.

1.2.3 Particularities and Challenges of Music Recommendation

In principle, some of the scenarios discussed in the previous section can be addressed with approaches from other recommendation domains like e-commerce. For instance, collaborative filtering approaches can be utilized to generate a list of *relevant* tracks for a user based on her previously liked tracks. Or, *session-based* recommending techniques from e-commerce can be applied for generating radio stations given the

user's recently played tracks. However, there are specific challenges or aspects that are at least more relevant in music recommendation scenarios. In this section, some particularities and challenges of the music recommendation problem are discussed, which are partly adopted from [Sch+17c].

Consumption aspects. The major difference between the music recommendation problem and other recommendation domains relates to *how* music is usually *consumed* and, among others, includes the following.

- *Consumption time of music is short*. Compared to, e.g., watching a movie, listening to a song needs less time. This comparably short consumption time requires lower user commitment and makes music items more "disposable". At the same time, this reduces the cost of bad recommendations (*false positives*) for a music recommender system [Sch+17b].
- *Music is consumed sequentially*. Be it a curated playlist or a radio station, we usually listen to music in sequence. Recommending a sequence of tracks requires special attention, e.g., to the transition between the tracks or to the evaluation of the list as a whole [Tin+17].
- *Re-consumption of the same music is common*. It is common in the music domain to repeatedly listen to *favorite* tracks. If recommendation algorithms should support repeated consumptions, it is important to decide *which* tracks to recommend repeatedly and *when* to recommend them [Jan+17a].
- *Music is often consumed passively*. Playing music in the background (e.g., at the work place or at a party), which could constitute the major share of a user's daily consumption of music, can make it harder for a recommender system to collect feedback and to learn user preferences from them. For instance, listening to a track can not be interpreted with absolute certainty as liking the music, because the user may not be paying attention to the music or simply may not have access to the music player at the moment [Sch+17b].
- *Music is consumed immediately*. Recommended tracks of, e.g., a personalized radio stations are immediately consumed by the listeners. A main challenge in this context is that the system should be *reactive*, which means that the listeners should be able to express their feedbacks to the currently playing track – by means of, e.g., a like/dislike button – and the system should consider this feedback for its subsequent recommendations.

Music preferences are context-dependent and intent-oriented. The music that we listen to can depend substantially on the situation that we are in (our *context*) and why we are listening to music (our *intention*). For example, if we are preparing a road trip playlist, we might select more familiar songs to sing along with them.

Or, to be more focused for studying, we might prefer music pieces with no vocals. We might even like different music in the morning than in the evening. In all of these examples, a recommender should be able to capture and consider the contextual factors and to recommend tracks that fit the intention of the listener.

Social component influences musical preferences. In addition to the context and intention, the social environment as well as trending music can also affect the musical preferences of the listeners. Therefore, making social-based user models by means of the users' online profiles and relationships can be helpful in some recommendation scenarios. However, some studies show that what people publicly share does not necessarily correspond to their private listening behavior [Jan+14].

Available music catalog is comparably large. While movie streaming services can typically have up to tens of thousands of movies, the number of tracks on Spotify, for instance, is more than 30 million, while new items are constantly added to the catalog, too. The main challenge in this context, especially for academic experiments, is the scalability of the recommendation approaches.

Acquisition of information about tracks is difficult. A number of recommendation algorithms rely on musical features, metadata of tracks and artists, or social tags. Acquiring such information for a large number of tracks is, however, time consuming and computationally expensive. Furthermore, the main challenge of utilizing community-provided social tag collections for the recommendation purposes is that they are noisy and often only available for popular tracks. Expert-provided annotations are also usually incomplete and strongly based on subjective evaluations [Cel10; Bon+14].

1.3 Next-Track Music Recommendation

As mentioned previously, the focus of this thesis is on a specific type of music recommendation, i.e., the recommendation of a list of tracks to be played next, given a user's most recent listening behavior. This is referred to as the "next-track music recommendation" or "playlist continuation" problem in the research literature. Considering the music recommendation scenarios discussed in Section 1.2.2, *supporting playlist creation* and *creating virtual radio stations* are the scenarios where such recommendations are applied. Various algorithmic approaches to this problem have been proposed in recent years [Log04; Har+12; Kam+12b; Moo+12; Wu+13; Bon+14; Vas+16]. A detailed discussion on next-track recommendation algorithms will be presented in Chapter 2.

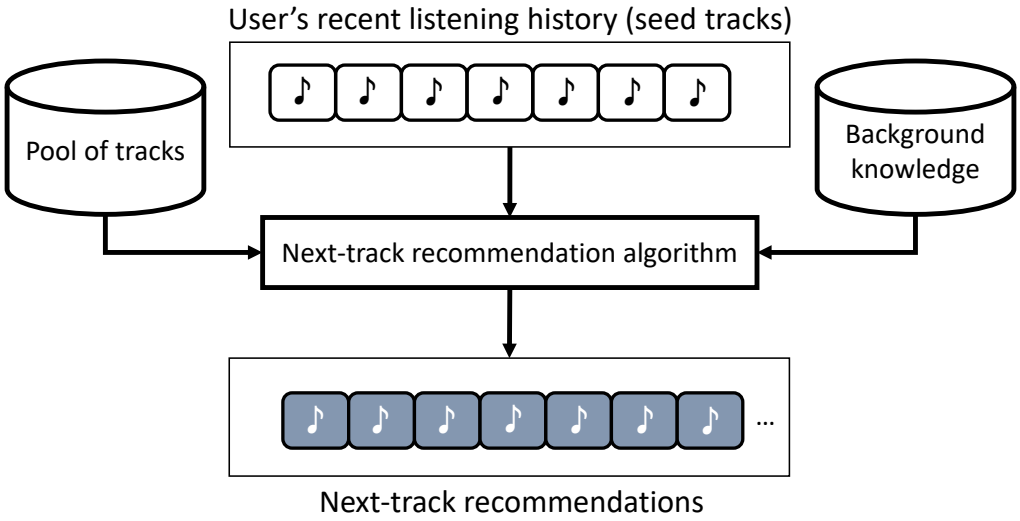


Figure 1.2: Illustration of the next-track recommendation process.

Figure 1.2 illustrates the next-track recommendation process. Bonnin et al. [Bon+14] define "automatic playlist generation" as selecting a sequence of tracks that fulfill the *target characteristics* of the playlist from an available *pool of tracks* using a *background knowledge* database. This definition can be adopted for the next-track recommendation process in both of the above mentioned scenarios.

- The *target characteristics* of a playlist or a radio station for which next-track recommendations should be generated are estimated mainly by *seed tracks*, i.e., the tracks that are already in the playlist or the recently played tracks on the radio station.
- *Background knowledge* about the available tracks in the catalog like musical features, metadata information about artist, genre, or release year, as well as associated social tags or explicit ratings are then used to determine whether or not the next-track recommendations satisfy the desired characteristics.

As will be discussed later in this thesis, additional information such as general user preferences or contextual and emotional information about users can be utilized in the recommendation process to improve the quality of next-track recommendations.

1.4 Research Questions

In this thesis, a number of research questions are considered regarding the topics that have not yet been fully investigated in the research field of next-track music recommendation. Details of *how* these questions have been developed and *why* it is important to seek answers for them will be discussed in the following chapters.

1. What do users consider as desirable characteristics in their playlists? What are the driving principles when users create playlists? How can analyzing the musical characteristics of user playlists help us design better next-track music recommendation algorithms and more comprehensive quality measures?
2. How can we combine patterns in the users' listening behavior with metadata features (e.g., artist, genre, release year), audio features (e.g., tempo or loudness), and personal user preferences to achieve higher recommendation accuracy? How can we utilize these input signals to optimize the selection of next tracks to create recommendations that are more coherent with the user's listening history?
3. How can we extract long-term preference signals like favorite tracks, favorite artists, or favorite topics from the users' long-term listening behavior for personalizing the next-track music recommendations? How can these signals be effectively combined with the user's short-term interests?
4. To what extent do the objective quality measures that are largely used in offline experiments to evaluate the quality of next-track recommendations correlate with the quality perception of music listeners?

In the context of this thesis, a number of novel algorithms and approaches were developed to answer the above-mentioned research questions. The effectiveness and usefulness of the proposed algorithms and approaches were explored through several offline and online experiments.

1. We conduct a systematic analysis on a relatively large corpus of manually created playlists shared on three different music platforms. The goal is to obtain insights on the principles that a recommender should consider to deliver better next-track recommendations. Therefore, musical features of the tracks of these playlists such as the tempo, energy, and loudness as well as the play counts and the associated user-provided social tags are queried through the public APIs of different music-related platforms. In particular, we analyze the *popularity*, *novelty* (in terms of release year), and *artist* and *genre diversity* level of user playlists. We also investigate the distribution of these features in the playlists and analyze the importance of *transitions* between the tracks with respect to different musical characteristics in user playlists that are shared on three different music platforms.
2. We propose a two-phase approach which aims to generate accurate next-track recommendations that match the characteristics of the recently played tracks of a user. In the first phase, we determine a set of suitable next tracks with respect to the user's most recent listening history. We base the selection and ranking of these tracks on a multi-faceted scoring method which combines

track co-occurrence patterns in publicly shared playlists, music and metadata features as well as personal user preferences. In the second phase, we optimize the set of next tracks to be played with respect to the user's individual tendency in different dimensions, e.g., artist diversity. Technically, we re-rank the tracks selected in the previous phase in a way that the resulting recommendation list matches the characteristics of the user's recent listening history.

3. We explore the value of including the users' long-term listening preferences into the recommendation process for personalization purposes. A number of approaches are therefore proposed to extract user-specific personalization signals from the listening behavior of users. In particular, *favorite tracks*, *favorite artists*, *favorite topics*, *co-occurred tracks*, and *online relationships* of users are investigated to learn the relevant personalization features. These additional personalization scores are then combined with a baseline recommendation technique, which focuses on the user's short-term interests, using a weighted scoring scheme.

4. We design and conduct an online user study (N=277) to assess to what extent the outcomes of offline experiments in the music domain correlate with the users' quality perception. Based on the insights obtained from our offline experiments, we state four research questions regarding (1) the suitability of manually created playlists for the evaluation of next-track recommending techniques, (2) the effect of considering *additional signals*, e.g., musical features or metadata into the recommendation process on the users' quality perception, (3) the users' perception of popular recommendations, and (4) the effect of familiar recommendations on the subjective quality perception of users.

1.5 Outline of the Thesis

The rest of this thesis is structured as follows. Chapter 2 reviews the next-track recommendation algorithms from the research literature. These algorithms are categorized into the four general groups of *content-based filtering*, *collaborative filtering*, *co-occurrence-based*, and *sequence-aware* algorithms. Afterwards, the results of a multi-dimensional comparison of a number of these algorithms from [Kam+17a] are presented. Next, with respect to the challenges of next-track music recommendation, different approaches for personalizing next-track music recommendations based on the users' long-term listening preferences from [Jan+17a] are presented. Finally, the algorithmic approaches from the literature for balancing the trade-offs between accuracy and other quality criteria like artist diversity are reviewed. In this context, the proposed approach in [Jan+15a] to generate optimized next-track recommendations in terms of different quality dimensions is presented.

Chapter 3 is dedicated to the evaluation of next-track recommendation algorithms. First, approaches to determine quality criteria for next-track recommendations are discussed. In this context, the results of an experimental analysis of user playlists that are published in [Jan+14] and a user study on the user's relevant quality criteria for playlist creation are presented. Next, different approaches to evaluate next-track recommendation algorithms are reviewed. In this regard, the results of a multi-dimensional comparison of different recommending algorithms [Jan+16] and the results of a user study about the perceived quality of different next-track recommending algorithms [Kam+17b] are presented.

Chapter 4 concludes the first part of this thesis by summarizing the discussed topics and by presenting future perspectives for the next-track music recommendation research. The second part of this thesis includes six of the author's publications that are listed in the next section.

1.6 Publications

The individual contributions of the author to the included publications in this thesis are as follows. The complete list of the author's publications can be found in the appendix.

1.6.1 Analyzing the Characteristics of Shared Playlists for Music Recommendation

Dietmar Jannach, Iman Kamehkhosh, and Geoffray Bonnin. "Analyzing the Characteristics of Shared Playlists for Music Recommendation". In: *Proceedings of the 6th Workshop on Recommender Systems and the Social Web at ACM RecSys*. 2014

This paper was a joint work with Dietmar Jannach and Geoffray Bonnin. The author of this thesis was involved in collecting the required music data, designing and implementing the experiments as well as evaluating the results.

1.6.2 Beyond “Hitting the Hits” – Generating Coherent Music Playlist Continuations with the Right Tracks

Dietmar Jannach, Lukas Lerche, and Iman Kamehkhosh. “Beyond "Hitting the Hits": Generating Coherent Music Playlist Continuations with the Right Tracks”. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. RecSys ’15. 2015, pp. 187–194

This work was written together with Dietmar Jannach and Lukas Lerche. The proposed “recommendation-optimization” approach was designed and developed by the author of this thesis in collaboration with the other authors of the paper. The author of this thesis was responsible for conducting the experiments and evaluating the results and wrote parts of the text.

1.6.3 Biases in Automated Music Playlist Generation: A Comparison of Next-Track Recommending Techniques

Dietmar Jannach, Iman Kamehkhosh, and Geoffray Bonnin. “Biases in Automated Music Playlist Generation: A Comparison of Next-Track Recommending Techniques”. In: *Proceedings of the 24th Conference on User Modeling, Adaptation and Personalization*. UMAP ’16. 2016, pp. 281–285

This study was a joint effort with Dietmar Jannach and Geoffray Bonnin. The author of this thesis contributed to the data collection, experimental design and implementation, and analyzing the results. He also wrote parts of the text.

1.6.4 Leveraging Multi-Dimensional User Models for Personalized Next-Track Music Recommendation

Dietmar Jannach, Iman Kamehkhosh, and Lukas Lerche. “Leveraging Multi-dimensional User Models for Personalized Next-track Music Recommendation”. In: *Proceedings of the 32nd ACM SIGAPP Symposium on Applied Computing*. SAC ’17. 2017, pp. 1635–1642

The paper was written with Dietmar Jannach and Lukas Lerche. The author of this thesis contributed to all parts of the paper and wrote parts of the text. The first version of this paper was presented at a workshop [Kam+16].

1.6.5 User Perception of Next-Track Music Recommendations

Iman Kamehkhosh and Dietmar Jannach. "User Perception of Next-Track Music Recommendations". In: *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. UMAP '17. 2017, pp. 113–121

This paper was written together with Dietmar Jannach. The author of this thesis contributed to all parts of the paper (including the design of the experiment, the implementation of the application, and the evaluation of the collected data) and wrote the major part of the text.

1.6.6 A Comparison of Frequent Pattern Techniques and a Deep Learning Method for Session-Based Recommendation

Iman Kamehkhosh, Dietmar Jannach, and Malte Ludewig. "A Comparison of Frequent Pattern Techniques and a Deep Learning Method for Session-Based Recommendation". In: *Proceedings of the Workshop on Temporal Reasoning in Recommender Systems at ACM RecSys*. 2017, pp. 50–56

The paper is the result of a joint work with Dietmar Jannach and Malte Ludewig. The experiments and the evaluation of the results were performed by the author of this thesis who also wrote the major part of the text.

Next-Track Recommendation Algorithms

2

A variety of algorithmic approaches have been proposed in the literature for the next-track recommendation task or the playlist continuation problem. In this chapter, these approaches are organized in the four categories of content-based filtering, collaborative filtering, co-occurrence-based, and sequence-aware. After a brief review of the research literature on next-track music recommendation algorithms and presenting the results of a multi-faceted comparison of a number of these approaches, two key challenges in this context along with our proposed approaches to deal with them will be introduced at the end of this chapter. These challenges relate to personalizing next-track recommendations and to balancing the possible trade-offs between accuracy and other quality factors, e.g., diversity.

2.1 Content-Based Filtering Algorithms

An intuitive strategy to select candidate tracks to be played next on a radio station or to be added to a playlist is to look for tracks that have similar *content* to the recently played (selected) ones, or to the user's favorite tracks.

Typically, in *content-based filtering* methods each track is represented as a musical feature vector and the *recommendable* next track is selected based on its similarity or *distance* to the user's profile. A user profile is also built based on the content of the recently played tracks and/or the favorite tracks of the user. The content information that is used in such methods is extracted either from the *audio signal* [Poh+05] or from the *metadata* of the tracks [Sla11].[10]

In the context of content-based approaches, typical *distance functions*, such as the Euclidean distance [Che+12; Moo+12], the Earth-Mover's distance [Log04], the Kullback-Leibler divergence [Vig+05], or the Pearson correlation [Bog+10] can be applied to determine the similarity (distance) between two tracks.

[10] For a detailed discussion on different types of content information see [Bog13].

2.1.1 Audio-Based Approaches

Extracting and processing audio content such as pitch, loudness [Blu+99], chord changes [Tza02], and mel-frequency cepstral coefficients (MFCC) [Tza+02; Bog+10] from a music file, using, e.g., machine learning approaches [Sch+11], is the main focus of the music information retrieval (MIR) research, see, e.g., [Cas+08; Mül15]; or [Wei+16].

More recent approaches based on audio content utilize *deep learning* techniques [Ben09] for both the feature extraction task [Hum+12; Die+14] and the recommendation problem [Oor+13; Wan+14]. For instance, Humphrey et al. [Hum+12] reviewed deep architectures and feature learning as alternative approaches for traditional feature engineering in content-based MIR tasks. Moreover, Dieleman et al. [Die+14] investigated the capability of convolutional neural networks (CNNs) [LeC+98] to learn features from raw audio for the tag prediction task. Their results showed that CNNs are able to autonomously discover frequency decompositions as well as phase and translation-invariant features.

One of the earliest deep learning content-based approaches for music recommendation was proposed by Oord et al. [Oor+13]. They compared a traditional approach using a "bag-of-words" representation of the audio signals with deep convolutional neural networks in predicting latent factors from music audio. They evaluated the predictions by using them for music recommendation and concluded that using CNNs can lead to novel music recommendations and reduce the popularity bias. In a similar work, Wang et al. [Wan+14] introduced a content-based recommendation model using a probabilistic graphical model and a deep belief network (DBN) [Hin+06] which unified feature learning and recommendation phases. Their experiments on The Echo Nest Taste Profile Subset [McF+12b] showed that the proposed deep learning method outperforms traditional content-based approaches in terms of predictive performance in both *cold* and *warm* start situations.

2.1.2 Metadata-Based Approaches

In contrast to audio content, metadata is not extracted from the music file. In fact, *metadata-based* approaches rely primarily on information, such as artist, album and genre [Bog+11; Aiz+12], release year [VG+05], or lyrics [Coe+13] of the tracks.

A common type of metadata-based algorithms exploit the *textual representation of musical knowledge* [Kne+13] that can be obtained from web pages [Poh+07], users' annotations [Lev+07], or lyrics [Log+04]. These algorithms apply typical text analysis methods, such as Term Frequency-Inverse Document Frequency (TF-IDF)

weighting [Whi+02] or latent semantic analysis (LSA) [Kne+08], to determine the similarity between music items, e.g., tracks, artists, or genres. For instance, in [Jan+15a], the selection of next tracks is based on the cosine similarity of the averaged TF-IDF vector of the recent listening history of a user and the TF-IDF vector of the target track. In this work, the TF-IDF vectors are computed based on the social tags assigned to the tracks by Last.fm users.

Content-based recommendation approaches are mainly favorable when *usage data*, i.e., information about the listening behavior of users like listening logs is not available [Zhe+10]. Furthermore, content-based methods do not suffer from the cold start problem, which enables them to find and recommend niche or new tracks from the catalog that have not or have rarely been listened to before. This is of more value in the music domain with its comparably longer tail [Cel08]. Another advantage of content-based recommendations is transparency, i.e., a recommended item can, in most cases, be explained easily by listing the content features or descriptions that were used to select that item.

A major limitation of content-based techniques, however, is that the generated next-track recommendations might be too similar to the seed tracks. This makes such recommendations sometimes not suitable for discovering new music. In addition, extracting musical features and determining the similarity of tracks can be computationally expensive [Bud+12].

2.2 Collaborative Filtering Approaches

Collaborative filtering (CF) approaches utilize community-provided feedback to determine the similarity between items. Similar to the general field of recommender systems, both *implicit* and *explicit* feedback have been used by CF methods for next-track music recommendation.

The “thumbs-up/thumbs-down” approach, for instance, is one of the most popular ways of collecting explicit user feedback on a track or recommendation. Moreover, *play* events – as a positive feedback – and *skip* actions – as a negative feedback – are also the most commonly used implicit feedback in the music domain. In the context of commercial applications, public presentations such as [Joh14] or [Joh+15] show that Spotify applies collaborative filtering, among other techniques, for certain recommendation tasks.

CF methods, in contrast to content-based approaches, do not need any item description and are therefore domain-independent. However, CF algorithms rely heavily on community feedback and are therefore sensitive to data sparsity and also suffer

from the cold-start problem in which no or only few interactions with new items are available in the system [Kne+13], or not enough information is available about new users to make any accurate recommendations.

Matrix factorization (MF) methods [Pan+08; Kor+09], which aim to find latent features that determine how a user interacts with an item, have been developed to alleviate the data sparsity problem of CF. In the music domain, for instance, Spotify uses a distributed MF method based on listening logs for its discovery feature [Joh14; Joh+15]. In this particular implementation of matrix factorization, the entries in the user-item matrix are not necessarily the explicit item ratings, but they correspond to the number of times a user has played each track.[11] Moreover, a distributed computing architecture based on the *map-reduce* scheme was utilized by Spotify to cope with the huge amount of required computations.

To overcome the cold-start problem of CF methods, some platforms, such as Microsoft Groove[12] or Deezer, ask users to provide some initial set of preferences for certain artists or genres. Furthermore, content information is usually combined with CF using a hybridization approach [Bur02]. For instance, Vasile et al. [Vas+16] propose a hybridization of an embedding-based method [Grb+15], a CF method, and track metadata for the next-track recommendation task. Their results show that the hybrid method outperforms all other tested methods.

Another drawback of CF techniques is their bias towards popular items [Jan+16]. The proposed methods in the literature to tackle this problem rely mainly on filtering techniques with respect to available popularity information in the system (e.g., play counts and/or release year) [Cel08].

2.3 Co-Occurrence-Based Algorithms

Another category of next-track music recommendation techniques approximate the similarity of music items based on their *co-occurrence*, e.g., on web pages [Coh+00], peer-to-peer networks [Sha+09], microblogs [Zan+12], or in playlists [Can+04]. The main assumption here is that two tracks or artists that appear together, for example within the same context or situation, may bear some similarity [Kne+13]. Co-occurrence-based methods can, in fact, be considered a subset of collaborative filtering which utilize co-occurrence information as implicit feedback to find similar music items or "similar-minded" listeners. In this thesis, however, as done historically, we will organize co-occurrence-based algorithms in a separate group.

[11] The advantages of using play counts instead of explicit ratings are also explored in the research literature, see for instance [Jaw+10].

[12] `https://music.microsoft.com/`

In an early work in this context, Pachet et al. [Pac+01] evaluated the co-occurrence function as a similarity measure using an expert-based ground truth. Their results showed the better performance of the co-occurrence function in comparison with the correlation function in approximating the artist and track similarity in radio station playlists. Similarly, Bonnin et al. [Bon+14] proposed a next-track recommending method called "collocated artists – greatest hits" (CAGH) which recommends the greatest hits of artists of the recently played tracks or of artists that are similar to them. The similarity of two artists corresponded to the number of times they appeared together in the listening sessions or playlists of the training data.

More elaborate co-occurrence-based approaches aim to determine *frequent patterns* in the data (playlists or listening logs of users).[13] Association rules (AR) are one type of such patterns. AR are often applied for market basket analysis with the goal to find sets of items that are bought together with some probability [Bon+14]. A well-known example of applying association rules for the recommendation problem are Amazon-like "Customers who bought this item also bought these items" recommendations.

Frequent patterns can, however, be identified more effectively by applying a *session-based* k-nearest-neighbor (kNN) approach as discussed in [Har+12] or [Bon+14]. For instance, the kNN method proposed in [Bon+14] looks for past listening sessions which contain the same tracks as the current session for which a next track should be recommended. Figure 2.1 illustrates the general idea of this approach. The assumption is that the additional tracks in a very *similar* listening session match well to the tracks of the current session. A binary cosine similarity measure based on track co-occurrence is used to compute the similarity of playlists.

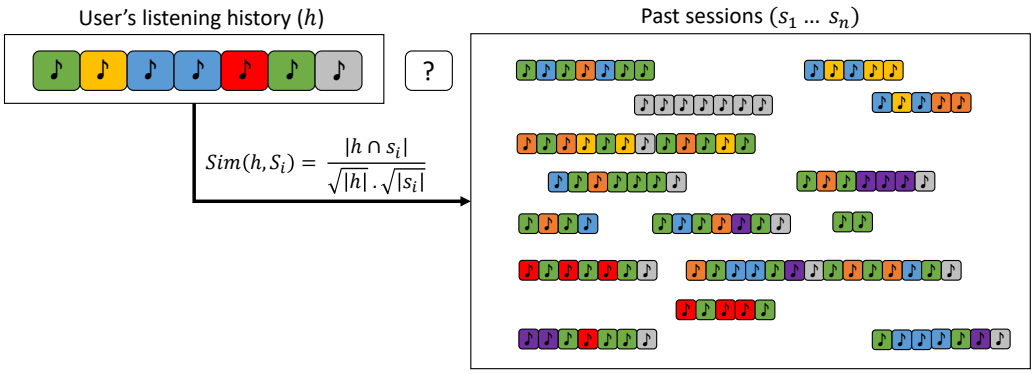


Figure 2.1: The proposed kNN approach in [Bon+14]. The k nearest neighbors of the recent listening history of the user are computed based on the cosine similarity of the tracks in the listening history and the tracks in the past listening sessions in the training data.

[13] Co-occurrence-based methods are also known as *frequent pattern approaches* in the literature.

2.4 Sequence-Aware Algorithms

The presented algorithms in the previous section do not consider the sequence of the tracks in a listening session or playlist for which the next track should be recommended. To address this problem, a number of *sequence-aware* techniques were also investigated in the literature.

Early academic sequence-aware approaches relied on the detection of sequential patterns [Agr+95] in the listening sessions or the created playlists of users. In a related approach, Hariri et al. [Har+12] proposed to mine sequential patterns of latent topics based on the tags attached to the tracks to predict the context of the next song. Their results showed that considering these topic transitions leads to a performance improvement over a kNN technique. In another work, Park et al. [Par+11] proposed to capture sequence patterns from session information. These patterns were then integrated in the recommendation process based on collaborative filtering to predict the next song. They showed that their proposed sequence-aware CF approach can outperform a basic CF method.

A possible drawback of sequential pattern mining techniques – or any rule mining approach in general – is that the quality of the generated recommendations depends on the number and quality of the listening sessions or playlists used for rule extraction [Bon+14].

More elaborate sequence-aware approaches to the next-track recommendation problem are based on *sequence modeling* using, e.g., Markov models. The main assumption of Markov-model-based approaches in this context is that the selection of the next item in a listening session or playlist is dependent on a limited number of previous listening actions. McFee et al. [McF+11], for instance, proposed a music playlist generation algorithm based on Markov chains that – given a song – selects the next track from uniform and weighted distributions as well as from k-nearest-neighbor graphs. Generally, the main challenge of applying Markov chains for the recommendation problem is the scalability of the state space when the number of possible sequences of user selections increases [Gra+14; Hid+15].

More recent sequence-modeling-based approaches leverage recurrent neural networks (RNN) to predict the next item in a sequence [Ber14]. RNNs process sequential data one element at a time and are able to selectively pass information across sequence steps [Lip15]. Often, such networks are used to learn embeddings of content features in compact fixed-size latent vectors, e.g., for music, images, video data, or documents or to represent the user [Elk+15; Ban+16; Cov+16]. While several recent works like [Hid+15; Tan+16; Hid+17] indicate that RNN can be successfully

applied for sequence-aware recommendation, the scalability issues (e.g., hyper-parameter optimization for the complex networks) hamper the application of such methods in real-world scenarios.

2.5 A Comparison of Session-based Approaches for Next-Track Music Recommendation

In recommendation scenarios where a recommender system has no access to the user's long-term preferences, making recommendations is solely based on the user's last interactions. For example, on an e-commerce website, when a visitor is new (or not logged in), there are no long-term user models that can be applied to determine suitable recommendations for this user. Such types of problems are not only common on e-commerce sites, but also can be observed for other application domains such as news [Özg+14] and music recommendation [Jan+15a].

The problem of predicting the next actions of users based solely on their sequence of actions in the current session is referred to in the literature as "session-based recommendation". Many of the algorithms that have been reviewed in this chapter such as frequent pattern mining algorithms or approaches that are based on sequence modeling can be employed to address the session-based recommendation problem.

In [Kam+17a], which is one of the papers included in this thesis by publication, we compared a number of these algorithms, among others, for next-track recommendation. These comparisons should help us obtain a better understanding of the effectiveness and the practicability of the proposed algorithms for the session-based next-item recommendation problem. An interesting question in this context relates to the performance of more recent RNN-based approaches in comparison with computationally less expensive pattern mining approaches.

The comparisons done in [Kam+17a] included the proposed RNN-based algorithm in [Hid+15] named GRU4REC, two co-occurrence-based approaches (including one session-based kNN algorithm and a sequence-aware extension of it), and two rule mining approaches (including an association rules and a sequential rules technique). These algorithms were evaluated in terms of accuracy, popularity and concentration biases and their computational complexity on six different datasets from e-commerce and music domains including two listening logs and two playlists datasets.

Algorithms. The RNN model applied in [Hid+15] uses Gated Recurrent Units to deal with the vanishing or exploding gradient problem. Figure 2.2 depicts the general architecture of this model. The network takes the current item of the session

as the input and predicts the likelihood of being the next in the session for each item. The GRU layer(s)[14] are the core of the network and additional embedding layers as well as feedforward layers can be added before and after the GRU layers, respectively.

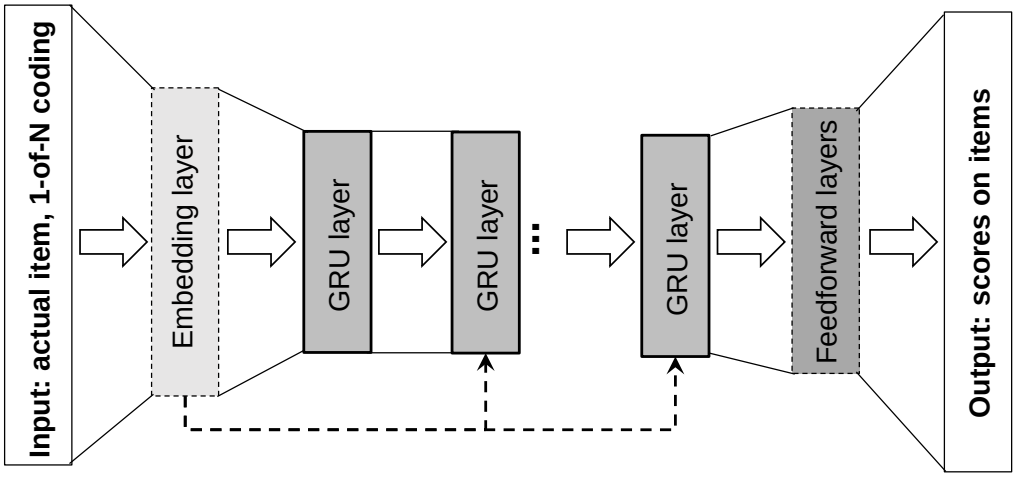


Figure 2.2: General architecture of the GRU-based RNN model, adapted from [Hid+15].

The session-based kNN method used in [Kam+17a] is similar to the kNN approach from [Bon+14] described above (see Figure 2.1). It looks for the k most similar past sessions (*neighbors*) in the training data based on the set of items in the current session. However, to reduce the computational complexity, only a limited number of the most recent sessions are considered in this process. The other kNN-based approach in this work – in addition to the item co-occurrence patterns in sessions – considers the sequence of the items in a session. More precisely, a track will be considered *recommendable*, only if it appears in the neighbor listening session directly after the last item of the current listening history.

Both of the rule-mining approaches deployed in [Kam+17a] define a rule for every two items that appear together in the training sessions (rule size of two). For the association rules method the *weight* of each rule is the number of co-occurrences of the two items, while the sequential rules technique in addition considers the sequence of the items in a session.

Datasets. For the e-commerce domain, we chose the *ACM RecSys 2015 Challenge* dataset (of 8 million shopping sessions) and a public dataset published for the *TMall competition* (of 4.6 million shopping logs from the Tmall.com website). For the music datasets, we used two subsets of listening logs collections from the *#nowplaying* dataset [Zan+12] (with 95,000 listening sessions) and the *30Music* dataset [Tur+15]

[14] In the implemented version of this algorithm by Hidasi et al. [Hid+15] that is publicly available at `https://github.com/hidasib/GRU4Rec`, only one GRU layer is used.

(with 170,000 listening sessions), and two collections of hand-crafted playlists from the *Art-of-the-Mix* dataset [McF+12a] (with 83,000 playlists) and the *8tracks* dataset that was shared with us by the 8tracks.com platform (including 500,000 playlists). In general, the total number of sessions was larger for the e-commerce datasets. However, the number of unique items was higher in the music datasets than in the e-commerce datasets.

Evaluation protocol. The task of the recommendation techniques was to predict the next item in a (shopping or listening) session. As done in [Hid+15], events are incrementally added to the sessions in the test set. The average *hit rate*, which corresponds to recall in this evaluation setting, and the *mean reciprocal rank*, which takes the position of the hit into account are then measured. As done in [Jan+17c], a five-fold sliding window validation protocol was applied to avoid random effects.

Furthermore, the average *popularity* of the top-20 recommendations of the algorithms was measured to assess possible recommendation biases. The *catalog coverage* of each algorithm was analyzed by counting the number of different items that appear in the top-20 recommendation lists of all sessions in the test set. The results of the experiments can be summarized as follows.

Accuracy results. Overall, the results indicated that simple co-occurrence-based approaches deliver competitive accuracy results and at the end, at least one pattern-based approach in every investigated recommendation task could be found that was significantly better than the recent RNN-based method in terms of accuracy.

- On the *e-commerce* datasets, the frequent pattern methods led to higher or at least similar accuracy values as the RNN-based GRU4REC method.
- For the *listening logs* datasets, the sequence-aware methods worked in general better than the sequence-agnostic approaches. In particular, the sequential-rules approach outperformed GRU4REC.
- On the *playlists* datasets, the sequence-aware version of the kNN method performed significantly better than the RNN-based approach.

Biases of the algorithms. With respect to popularity bias and catalog coverage, the results showed that

- the *sequence-agnostic* methods (e.g., kNN and association rules) are more biased towards popular items and focus their recommendations on a smaller set of items;

- the *sequence-aware* methods (e.g., the sequential-rules and RNN-based approaches) recommend less popular items and cover more different items in their recommendations.

Computational complexity. Table 2.1 summarizes the results of comparing the algorithms in terms of their computational complexity. The results showed that the simple sequential rules method, which performed better or as good as the RNN-based method in every experiment in terms of accuracy, is also much faster in the "training" phase (which corresponds to learning the rules for this algorithm) and requires less memory and less time for generating recommendations.

Table 2.1: Computational complexity of the algorithms used in the experiments of [Kam+17a]. The measurements are done on one split of the ACM RecSys 2015 challenge dataset [BS+15] with about 4 million sessions on a desktop computer with an Intel i7-4790k processor and Nvidia GeForce GTX 960 graphics card.

	"Training" time	**Creating one recommendation list**	**Memory used for data structures**
GRU4REC	12h (using CPU), 4h (using GPU)	12ms	510MB
k**NN**	27s	26ms	3.2GB
Sequential rules	48s	3ms	50MB

2.6 Challenges

In the introduction of this thesis, the particularities and challenges of music recommendation were discussed. In general, the subjectiveness and context-dependency of music make the recommendation task more challenging. In this section, we present two challenging aspects of next-track recommendation that have not been investigated to a large extent in the music recommendation literature, even though they have effects on the user's quality perception of music recommendations.

2.6.1 Personalizing Next-Track Recommendations

Most of the reviewed algorithms in the previous sections focus solely on the user's recent listening behavior or current context and do not consider the *long-term* preferences of the user. In one of the few attempts in the literature where the next-track recommendation is personalized, Wu et al. [Wu+13] proposed personalized Markov embedding (PME) for the next-track recommendation problem in online karaokes. Technically, they first embed songs and users into an Euclidean space in which distances between songs and users reflect the strength of their relationships. This

embedding could efficiently combine the users' long-term and short-term preferences. Then, given each user's last song, recommendations can be generated by ranking the candidate songs according to the embedding. Their results show that PME performs better than a non-personalized Markov embedding model [Che+12].

In [Jan+17a], which is one of the papers included in this thesis by publication, we also aimed to personalize next-track music recommendations. In this context, we explored the value of including different types of information that reflect long-term preferences into the next-track recommendation process. For this reason, a multi-faceted scoring scheme, as illustrated in Figure 2.3, is utilized to combine a baseline algorithm (e.g., a kNN method as described above) that focuses on the short-term profile of the user with additional personalization components extracted from the user's long-term listening history. The goal is to determine a relevance score for each possible next track using different personalization signals.

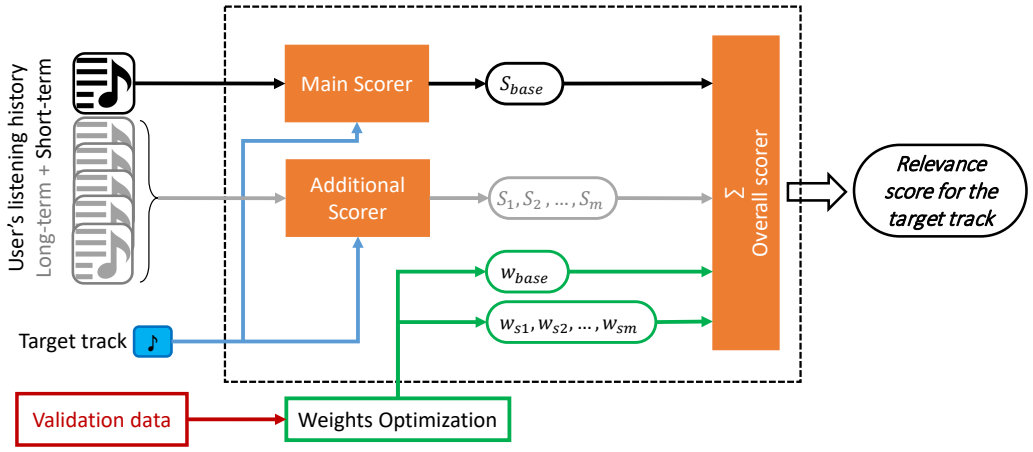


Figure 2.3: Illustration of the multi-faceted scoring scheme to combine a baseline algorithm with personalization components in a weighted approach [Jan+17a].

Technically, the overall relevance score $score_{overall}$ for a possible next track t^*, given the current listening history h, is computed as follows [Jan+17a],

$$score_{overall}(h, t^*) = w_{base} \cdot score_{base}(h, t^*) + \sum_{pers \in P} w_{pers} \cdot score_{pers}(h, t^*) \tag{2.1}$$

where P is a set of personalization strategies, each with a different weight w_{pers}, and w_{base} is the weight of the baseline. The functions $score_{base}$ and $score_{pers}$ compute the baseline score and the scores of the individual personalization components, respectively.

The personalization approaches proposed in this work consider the following signals, including *favorite tracks*, *favorite artists*, *topic similarity*, *extended neighborhoods*, and *online social friends*.

Favorite tracks. Users usually listen to their favorite tracks over and over again. This simple pattern in the listening behavior of users can be adopted by recommender systems to generate personalized recommendations. We examined different strategies to determine which tracks from the user's previously played tracks to recommend. For instance, we selected generally popular tracks from her listening history, the tracks which have been played at the same time of the day in the past, or the tracks from the same artists as in the current session. Since users tend to re-consume more recent items [And+14], we assigned more weights to tracks that were more recently listened to by the user.

In addition to determining *which* tracks to recommend, another challenging task in this context is to decide *when* it is a good time to recommend an already known (favorite) track. In a simple strategy, we assumed that recommending a favorite track might be interesting for a user when a minimum proportion of the played tracks in her recent listening history (e.g., 50%) were tracks that she has played in the past. In general, the correct detection of the user mode (*exploration* vs. *repetition*) can improve the quality perception of recommendations, as discussed in [Kap+15].

Favorite artists. Just like favorite tracks, music enthusiasts also have their favorite artists. The idea here is to recommend tracks of not only the artists that user liked (played) in the past, but also to consider the popular tracks of similar artists in the recommendation process. The relevance of a candidate track then depends on its general popularity and the similarity between its artist and the user's favorite artists. As a measure of similarity between two artists, one can, e.g., simply count how often two artists appear together in the users' listening sessions or playlists, see Section 2.3.

Topic similarity. The assumption of this personalization score is that some users listen only to certain types of music, for example, mostly romantic songs or instrumental music. One way to determine the topic of a track is to use social tags that are assigned to them on music platforms like Last.fm. In this context, a higher score is assigned to tracks that are annotated with similar tags. The similarity can be computed based on, for example, the cosine similarity between two TF-IDF encoded track representations. In general, however, other musical features can also be used to determine a content-based similarity of the tracks, see Section 2.1.2.

Extended neighborhood. To personalize the recommendations of a neighborhood-based method (see Figure 2.1), the idea is to consider not only the current listening session of the user for finding similar sessions (neighbors), but also her past sessions, maybe with a lower weight.

Online social friends. Finally, the last personalization approach that we considered in [Jan+17a] takes the musical preferences of the user's online social friends into account, as their listening behavior can influence the user's preferences. In our experiments, we explored the value of recommending the favorite tracks of the user's Twitter friends and gave more weight to (popular) friends with more followers.

The performance of these personalizing approaches were then evaluated using hand-crafted playlist collections and listening logs datasets. To measure accuracy, we computed the track *hit rate* of the approaches as follows. The data was split into training and test sets, and the last track of each playlist or listening session in the test set was hidden. The goal was to predict this last hidden track. A "hit" was counted when the hidden track was in the top-n recommendation list of an algorithm. In addition to accuracy, the *diversity* and *coherence* of the resulting recommendations based on the artists – and where applicable based on the tags – were also assessed.

Overall, the experimental evaluations in [Jan+17a] on different datasets of listening logs and hand-crafted playlists showed that all these personalization features can have a positive effect on the quality of the resulting recommendations. For instance, the results on the *#nowplaying* dataset [Pic+15] showed that repeating appropriate candidate tracks during a listening session increased the accuracy of recommendations up to 30% and made the recommendations more connected to the listening history. The best results were achieved when multiple signals were combined. A successful combination of multiple signals, however, requires the fine-tuning of the importance weights in the scoring scheme (see Equation 2.1).

2.6.2 Beyond Accuracy – How to Find the Right Next Tracks

Apart from numerous algorithmic contributions, a side effect of the Netflix prize[15] [Ben+07] was an enormous focus of researchers on accuracy measures like the mean absolute error (MAE) or root mean squared error (RMSE) for evaluating the quality of recommendation algorithms [Jan+12]. However, several recent works indicate that optimizing accuracy could be insufficient in many real-world recommendation scenarios and there are other quality criteria that could affect the user's quality perception of the recommendations [Cre+12; Jan+15c].

[15] In 2006, Netflix released a dataset containing 100 million anonymous movie ratings of its costumers for a public challenge on improving the accuracy of it recommender system, Cinematch, by 10%.

In the music domain, in particular, the proposed approaches in the research literature are most often evaluated based on historical data and are mainly aimed to identify tracks that users actually listened to, using performance measures like the mean reciprocal rank (MRR), precision, or recall. Although the relevant quality criteria for a playlist or listening session might vary based on the context or intent of the listeners, some works have attempted to determine additional quality criteria that are relevant to find the *right* next tracks. Examples of such quality factors are artist *diversity*, *homogeneity* of musical features, or the *transitions* between the tracks. Common approaches to determine such factors are to conduct user studies [Kam+12a] or to analyze the characteristics of published user playlists [Sla+06; Jan+14], see Section 3.1.

When quality factors other than prediction accuracy are considered in the recommendation process, it can become necessary to find a trade-off as improving one quality factor could impact another one negatively. Some works in the research literature on recommender systems have also proposed approaches to deal with such trade-off situations and to improve recommendations by considering additional quality factors. For instance, the work presented in [Ado+12] tried to re-rank the first n items of an accuracy optimized list in a way to increase or balance diversity across all users. Bradley et al. [Bra+01] and Ziegler et al. [Zie+05] also aimed to optimize diversity, this time in terms of intra-list similarity, using techniques that reorder the recommendations based on their dissimilarity to each other. Finally, Zhang et al. [Zha+08] proposed a binary optimization approach to ensure a balance between accuracy and diversity of the top-n recommendations.

The main shortcomings of the proposed approaches are that (1) they consider only two quality factors, for example, accuracy versus diversity and (2) do not take the tendencies of individual users into account. Some more recent works try to overcome these limitations [Oh+11; Shi+12; Rib+14; Kap+15]. For instance, in [Kap+15] a regression model is proposed to predict the user needs for novel items from her past interactions in each recommendation session individually. These approaches also have their limitations. For instance, the above mentioned approach from [Kap+15] is designed for only a specific quality factor, i.e., novelty. Furthermore, the proposed balancing strategies are often integrated in the specific algorithmic frameworks which makes such approaches difficult to reuse.

To address these problems, in [Jan+15a], which is one of the papers included in this thesis by publication, we proposed a two-phase *recommendation-optimization* approach that can be combined with any existing item-ranking algorithm to generate accurate recommendations that also match the characteristics of the last played tracks. Figure 2.4 shows an overview of this approach.

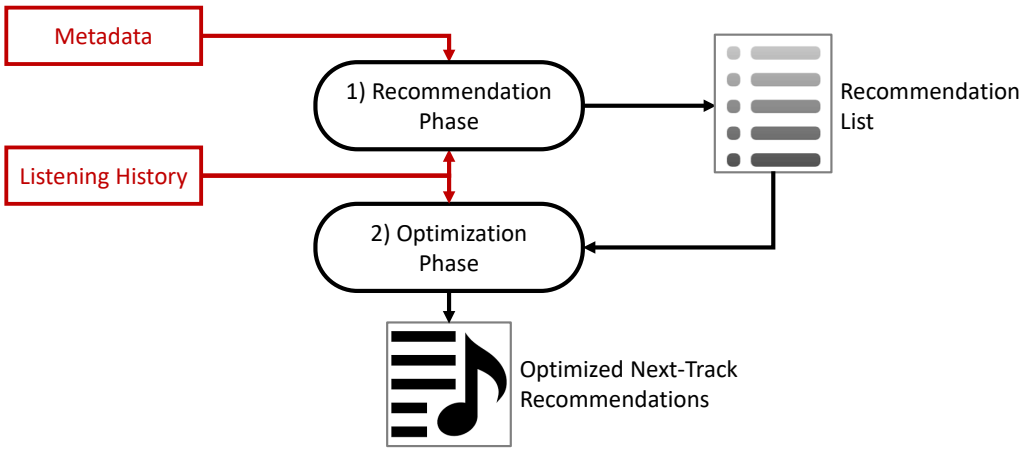


Figure 2.4: Overview of the proposed *recommendation-optimization* approach in [Jan+15a].

Recommendation phase. The goal of the first phase (the recommendation phase) is to determine a relevance score for each possible next track t^* given the current listening history or the list of tracks added so far to a playlist (playlist beginning) h, using different input signals. A weighted scoring scheme – similar to Equation 2.1 – is then used to combine a baseline next-track recommending algorithm (e.g., kNN) with additional suitability scores. In general, the combination of different scores shall serve two purposes, i.e., (1) increasing the hit rate as more relevant tracks receive a higher aggregated score and (2) making the next-track recommendations more homogeneous.

In our experiments, we examined different suitability scores to be combined with the baseline scorer. One of these scores considered the *topic-based similarity* of the tracks based on the social tags assigned to them. The idea here is that if several tracks in the history were annotated by users with certain tags, we should increase the relevance score of tracks with similar tags. The cosine similarity of the TF-IDF vectors of a target track (t^*) and the tracks of the user's listening history (h) was used to compute a topic-based suitability score, see Equation 2.2.

$$score_{\text{topic}}(h, t^*) = sim_{\text{cosine}}\left(\frac{\Sigma_{t_i \in h}\vec{t_i}}{|h|}, \vec{t^*}\right) \tag{2.2}$$

Another suitability score in our experiments was based on *musical features* like the tempo or loudness, and release year or popularity of the tracks. If we, for example, detect that the homogeneity of the tempo of the tracks is most probably a guiding quality criterion, we should give an extra relevance weight to tracks that are similar in tempo to those in the history. We assumed that if a feature is relevant, the spread of the values will be low. For instance, a low spread and variance of the tempo values of the tracks in the listening history – e.g., in the range of 110 to 120 bpm – indicates that the user generally prefers to listen to *moderato* music.

To be able to combine this signal with the baseline recommendation technique, the Gaussian distribution of numerical features like tempo can be used as a suitability score of a target track (t^*). Mean (μ) and standard deviation (σ) are computed based on the distributions of the respective numerical feature in the history (h), see Equation 2.3.

$$score_{\text{feature}}(h, t^*) = \frac{1}{\sigma_h \sqrt{2\pi}} e^{-\frac{(f_{t^*} - \mu_h)^2}{2\sigma_h^2}} \tag{2.3}$$

Optimization phase. After determining a list of *accurate* and *homogeneous* next-track recommendations in phase one, the goal of the second phase (the optimization phase) is to optimize the selection of the next tracks with respect to the listening history of each individual user. The general idea is to take a limited number of the most relevant top recommendations generated by any item-ranking or scoring algorithm and to re-rank them in a way that the top-n items which will be presented to the user match some selected quality factors as much as possible. For example, the top 30 selected tracks of a recommendation service on a radio station can be re-ranked so that the top 10 recommended next tracks have the same level of artist diversity as previously played tracks.

A critical question in this context is how diverse the next-track recommendations should be. There are approaches that determine the desired level of a quality factor (e.g., diversity) globally. However, in reality, the optimal diversity of a recommendation list could depend mainly on the context, intent, and the users' general preferences and may vary for each individual listener.

In the proposed approach by Jannach et al. [Jan+15a], the individual level of diversity is determined based on the last played tracks of each user (*seed tracks*) and the optimization goal is to *minimize the difference* between the characteristics of a feature in the listening history and the recommended next tracks. Different measures like *mean and standard deviation*, *aggregate measures*, and *mixture models* can be used to quantify the characteristics of seed tracks.

In contrast to the work done by Jambor et al. [Jam+10] in which an individualized "per-user" optimization technique for numerical quality factors was introduced, in [Jan+15a], items from the long tail are not promoted and the applied re-ranking technique, as it will be described in the following, can be configured to exchange elements from a comparably small set of items from top of the recommendation list. In this way, the computational complexity of the optimization phase is reduced and too high accuracy losses can also be prevented.

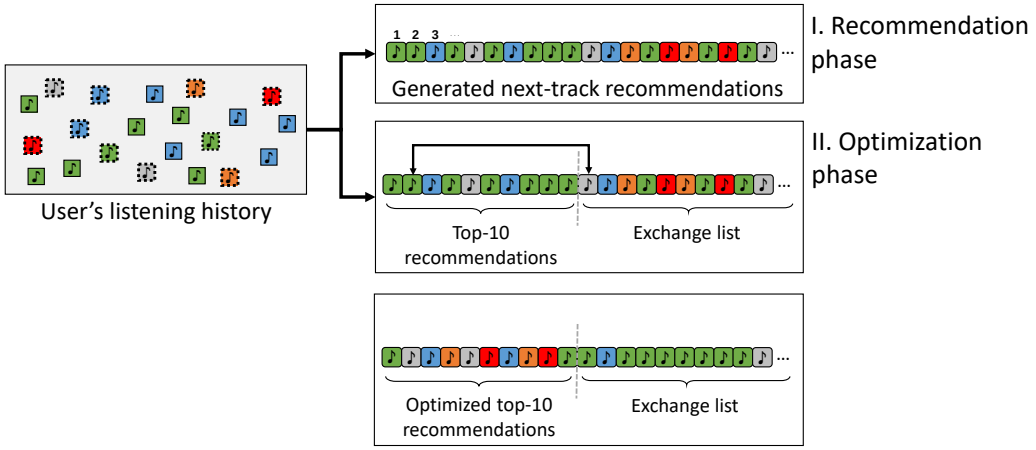


Figure 2.5: Illustration of the re-ranking scheme, adapted from [Jug+17].

Re-ranking scheme. Figure 2.5 illustrates the re-ranking scheme based on the artist diversity problem. The elements in dotted lines in the listening history rectangle represent the selected seed tracks and the different colors represent different artists. The seed tracks can be taken from the set of the user's last played tracks or be a subset of the user's favorite tracks. Based on the seed tracks, the user's individual *tendency* towards artist diversity can be computed. Again, different colors in the generated next-track recommendation list represent different artists. Since the top-10 tracks of the recommendation list have a lower artist diversity than the seed tracks in the listening history, the algorithm then starts exchanging elements from the top of the list with elements from the end of the list ("exchange list"), which probably have a slightly lower predicted relevance but help to improve the diversity of the top-10 list, i.e., to minimize the difference between the diversity level of the top-10 list and the seed tracks. So if a user generally prefers lists with high diversity, the re-ranking will lead to higher diversity. Vice versa, a user who usually listens to various tracks from the same artist in a session will receive recommendations with a lower artist diversity. As a result, the definition of a globally desired artist diversity level can be avoided.

Another advantage of the proposed approach is that multiple optimization goals can be considered in parallel as long as the relative importance of the difference factors can be specified. The acceptable compromises on recommendation accuracy can also be fine-tuned by determining the number of top items which should be considered in the optimization process. For more details on this re-ranking algorithm, see [Jug+17].

Evaluation of Next-Track Recommendations

3

Like in the general field of recommender systems, utilizing information retrieval or machine learning measures is the most common strategy in academia to evaluate the quality of next-track music recommendations. Although the major focus of research in this field has been on accuracy measures, the relevance of other quality factors like the homogeneity of the recommended tracks, artist diversity, coherence with the previous tracks, or the transitions between the tracks has also been discussed in the literature [Jan+16; Jan+17a]. In general, the main goal of any evaluation approach should not only be to assess the relevance of the individual recommendations, but also to consider quality factors that are determined by the characteristics of the recommendation list as a whole. In this chapter, we first discuss approaches to determine quality criteria for next-track recommendations and then review approaches to evaluate the quality of next-track recommendation algorithms.

3.1 How to Determine Quality Criteria for Next-Track Recommendations

Bonnin et al. [Bon+14] discuss two approaches to determine quality criteria for next-track music recommendations (in the context of the playlist continuation problem), (1) analyzing the characteristics of user playlists, and (2) conducting user studies.

3.1.1 Analyzing the Characteristics of User Playlists

An intuitive way to learn how to select a *good* track to be played next is to look at the characteristics of the tracks that have been selected by real users in, e.g., previous listening sessions or playlists. For instance, to determine the general principles for designing a next-track recommender for the task of *automatic playlist continuation*, it will be helpful to analyze playlists that are created and shared by users, assuming that such *hand-crafted* playlists have been created carefully and are of good quality [Bon+14].

In the research literature, Slaney et al. [Sla+06], for example, investigated whether users prefer to create homogeneous or rather diverse playlists based on genre information about the tracks. In the study, they analyzed 887 manually created playlists. The results showed that users' playlists usually contain several genres and concluded therefore that genre diversity is a relevant feature for users. In another work, Sarroff et al. [Sar+12] focused on track transitions and examined the first 5 songs of about 8,500 commercial albums for latent structures. The results of two feature selection experiments using a Gaussian mixture model and a data filtering technique showed that fade durations and the mean timbre of song endings and beginnings are the most discriminative features of consecutive songs in an album.

Similar to [Sla+06] and [Sar+12], in [Jan+14], which is one of the papers included in this thesis by publication, we analyzed a relatively large set of manually created playlists that were shared by users. Our primary goal in this work was to obtain insights on the principles that a next-track recommendation algorithm should consider to deliver *better* or more "natural" playlist continuations. We used samples of hand-crafted playlists from three different sources including *last.fm*, *artofthemix.org* and *8tracks.com*. Overall, we analyzed 10,000 playlists containing about 108,000 distinct tracks of about 40,000 different artists. Using the public APIs of Last.fm, The Echo Nest[16], and the MusicBrainz database[17], we first retrieved additional information about audio features like the tempo, energy, and loudness of the tracks as well as their play counts and social tags assigned to them by users.

The goal of the first analysis in [Jan+14] was to determine the user tendency towards popular tracks. As a measure of popularity, we considered the total number of times a track was played on Last.fm. The results showed that users actually include more popular tracks (in terms of play count) in the beginning of the playlists in all datasets. Moreover, to measure the concentration biases in the user playlists, we calculated the Gini index to measure the inequality among the catalog items. The Gini index revealed that the tracks in the playlist beginnings are selected from smaller sets of tracks and the diversity slightly increases at the end of playlists.

Next, we analyzed to what extent the users' playlists contain recently released tracks. We compared the creation year of each playlist with the average release year of its tracks. The results showed that users include relatively fresh tracks (that were released on average in the last 5 years) in their playlists. Furthermore, our results revealed that the tracks of a user playlist are often homogeneous with respect to the release date.

[16] `http://the.echonest.com/`
[17] `https://musicbrainz.org/`

Analyzing hand-crafted playlists further indicated that the homogeneity and diversity of artists and genres in playlists depend mainly on the characteristics of the users of the platform and its possible limitations. For instance, on 8tracks it is not allowed to include more than two tracks from the same artist in a (public) playlist. Or, the Art-of-the-Mix (AotM) platform is specifically designed for sharing playlists among music enthusiasts who select their tracks more carefully [Cun+06]. In contrast, users of Last.fm often seem to create playlists from their personal favorite tracks to be used by themselves. In general, playlists on Last.fm cover less artists and genres, while on AotM and 8tracks, playlists are on average more diverse. Despite such platform-related particularities, the results indicated that users tend to keep the same level of artist and genre diversity throughout the playlists.

The distributions of musical features like energy, hotness (which corresponds to how famous a track is right now), loudness, danceability, and tempo of tracks (independently of playlists) were then compared with the average feature values of each user playlist. The results implied that users, in general, pay more attention to the energy and hotness of the tracks than the other features, e.g., loudness, danceability, and tempo, when creating playlists.

Furthermore, we analyzed the importance of transitions between the tracks by computing the *coherence* of a playlist which corresponds to the average similarity between its consecutive tracks. An interesting result in this regard is that the coherence values (in terms of artist, genre, energy and hotness) of the first tracks are higher than the last tracks in playlists. This may indicate that users select the first tracks of a playlist more carefully. Finally, comparing the characteristics of public playlists and the private playlists, which are not shared with others, on the 8tracks platform indicated that the private playlists generally contained more popular and more recent tracks than public playlists. These results can be interpreted as the users' attempts for creating a social image on the platform through sharing less popular or less known tracks and artists in their public playlists.

3.1.2 Conducting User Studies

Another possible approach to determine quality criteria for next-track recommendations is to conduct *user studies*. User studies, especially in the music domain have, however, their challenges and limitations.

- *User studies are time consuming and expensive.* The participants have to listen to a number of tracks during such experiments, which can take a long time.

- *Participants' ratings are biased towards familiar items.* The participants rate items that they already know higher than the ones that they do not know [Eks+14; Jan+15b; Kam+17b].
- *Users behave differently in simulated environments.* In general, when users feel being supervised (e.g., in a laboratory study) they might show different behavior than in normal situations.
- *Academic user studies often have a limited size.* One limitation of the conducted user studies in the music domain in academia is that they involve 10 to 20 participants in total, see, e.g., [Swe+02; Cun+06; Bau+10; Lam+10; Stu+11]; or [Tin+17]. This makes it difficult to generalize the findings of such studies.
- *It is difficult to reproduce user studies and/or generalize their findings.* User experiments are often conducted using a specific software that is developed for that study. This specific design limits the reproducibility of the experiment. Furthermore, recruiting the participants from a specific population, e.g., university students, using quite different music collections in different studies, or selecting the music from limited genres or styles, make it usually unclear, to what extent the findings of such studies can be generalized [Bon+14].

Most of the user studies in the music domain focus on how users search for music and on the social or contextual aspects of listening to music [Dow03; Lee+04; Cun+06; Cun+07; Lam+10]. Few studies also analyze the factors that could influence the selection of next-track recommendations by users. For instance, in the context of *playlist generation*, Stumpf et al. [Stu+11] presented the results of a user study on how users create playlists in different listening contexts. Analyzing the interactions of 7 participants with a playlisting tool (iTunes) in *think-aloud* sessions indicated that in more personal use cases like *private travel*, mood was selected as the most relevant feature for users, whereas in more public situations like *large party* or *small gathering* the rhythmic quality of songs was selected as the most important feature. Moreover, tempo and genre were identified as *context-independent* features that were considered equally important in all the examined contexts.

In another work, Kamalzadeh et al. [Kam+12a] conducted a user study involving 222 subjects on music listening and management behavior of users. Their results showed that mood, genre, and artists are the most important factors for users when selecting the tracks of a playlist. In a more recent work, Tintarev et al. [Tin+17] conducted an exploratory study (N=20) of the users' perceptions of diversity and ordering in playlist recommendations of the "Discover Weekly" service of Spotify. Their results indicated that novelty, diversity, and genre familiarity are important aspects for playlist recommendations, while ordering is an aspect that users usually do not pay attention to.

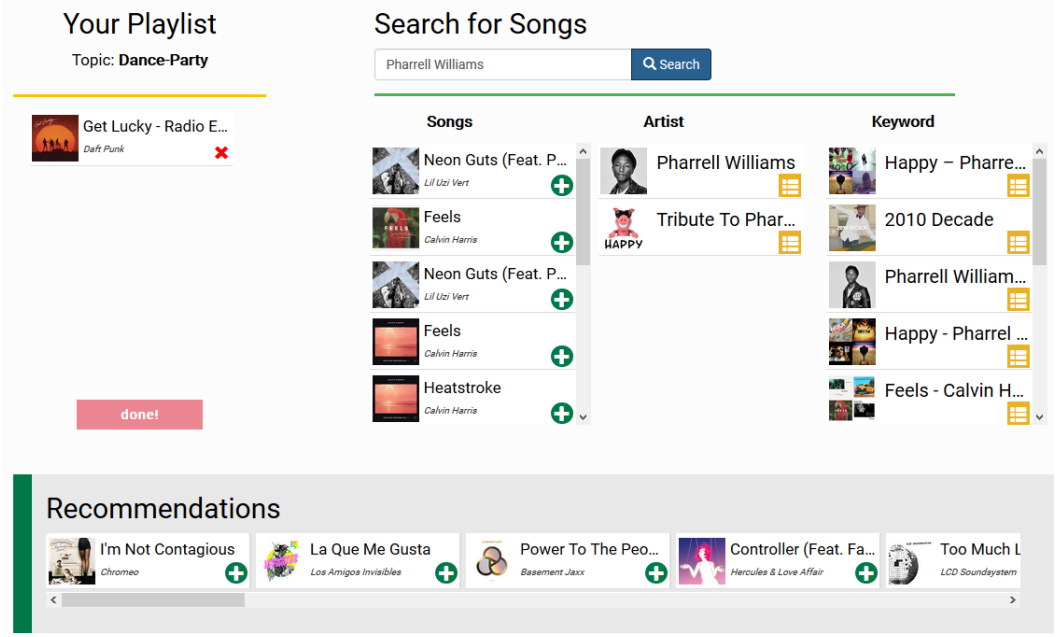


Figure 3.1: Web application used in the study for playlist creation.

In the context of this thesis, we conducted a between-subject user study involving 123 subjects to, among others, determine the relevant quality criteria for playlists. The findings of this study could help us better understand which quality characteristics should be considered when designing next-track recommending algorithms for playlist construction support. In the following, we present this user study in detail.

Study design. We developed a web application for the purpose of this user study. Using this application, the participants were asked to create a playlist with one of the pre-defined themes including *rock night*, *road trip*, *chill out*, *dance party*, and *hip hop club*. After choosing a topic, the participants were forwarded to the playlist creation page. All participants could use the provided search functionality to look for their favorite tracks or artists. However, to analyze the effect of automated next-track recommendations on the playlist creation behavior of users, the participants of the experimental group (`Rec`) received additional recommendations as shown at the bottom of Figure 3.1. The control group (`NoRec`) was shown the same interface but without the recommendations bar at the bottom.

Both the search and the recommendation functionality were implemented using the public API of Spotify[18], which allowed us to rely on industry-strength search and recommendation technology. When the playlist contained at least six tracks, the participants could proceed to the post-task questionnaire, in which they should accomplish the following tasks.

[18] `https://developer.spotify.com/web-api/`

1. In the first part of the questionnaire, the participants were asked to order a list of quality factors based on their relevance for their created playlist. They could also mark individual factors as irrelevant. The quality factors were selected from the respective research literature and related either to individual tracks (e.g., popularity or freshness) or to the list as a whole (e.g., artist homogeneity), see Figure 3.2(a).

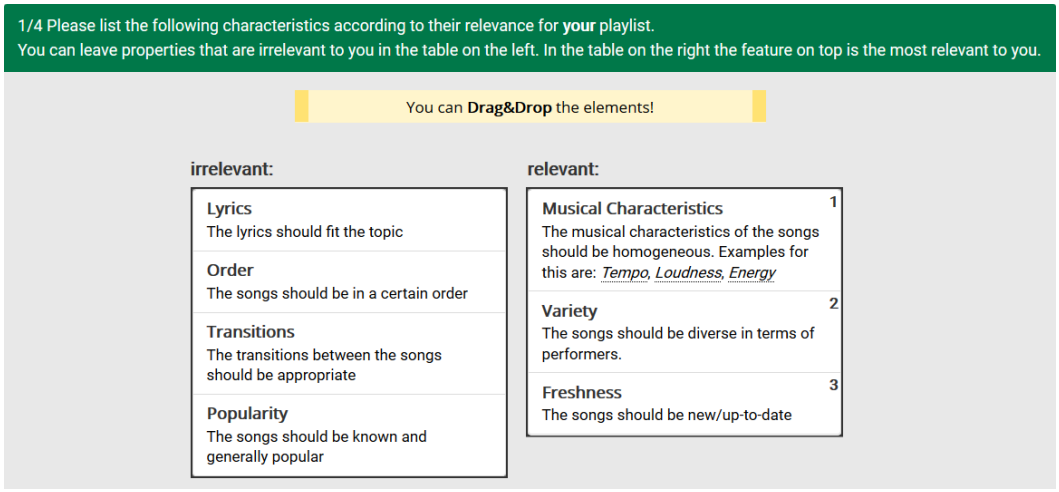
2. In the next step, participants of the experimental group (`Rec`) who were provided with recommendations were asked if they had looked at the recommendations during the task and if so, how they assessed their quality in terms of *relevance*, *novelty*, *accuracy*, *diversity* (in terms of genre and artist), *familiarity*, *popularity*, and *freshness*. Participants could express their agreement with the provided statements, e.g., "The recommendations were novel", on a 7-point Likert item or state that they could not tell, see Figure 3.2(b).

3. In the final step, all participants were asked (1) how often they create playlists, (2) about their musical expertise, and (3) how difficult they found the playlist creation task, again using 7-point Likert items. Free text form fields were provided for users to specify which part of the process they considered the most difficult one and for general comments and feedback, see Figure 3.2(c). The user study ended with questions about the age group and email address of the participants.

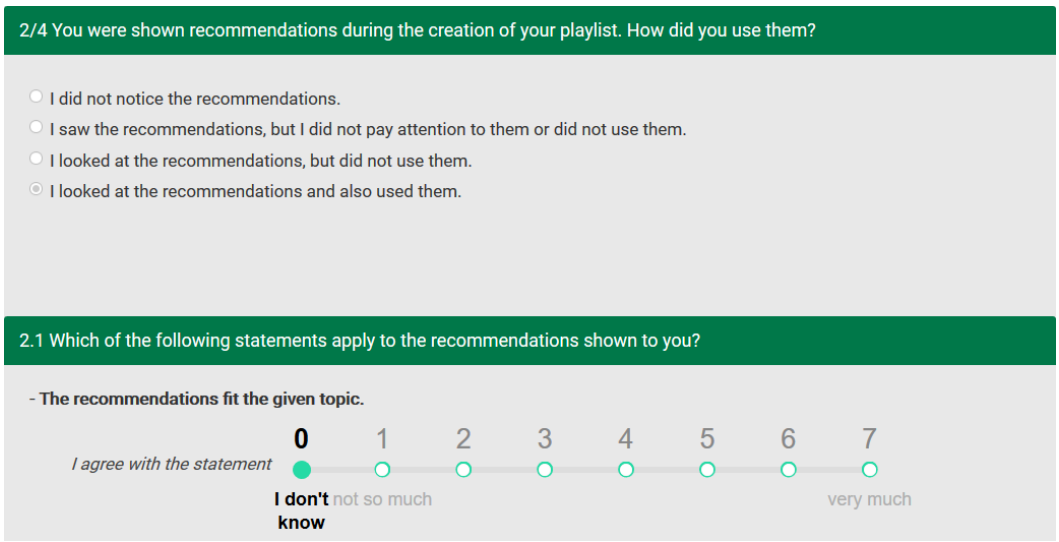
General statistics. At the end, 123 participants (mainly students, aged between 20 and 40) completed the study. Based on the self-reported values, the participants considered themselves as experienced or interested in music. However, they found the playlist creation task comparably difficult. 57% of the participants were assigned to the `Rec` group (with recommendations). Almost half of these participants (49%) drag-and-dropped at least one of the recommended tracks to their playlists. We denote this group as `RecUsed`. The other half will be denoted as `RecNotUsed`.

Study outcomes. Considering the topic of this section, first the relevant quality criteria that were determined by the subjects of the study are introduced. Afterwards, the observations on the effect of next-track recommendations on the playlist creation behavior of users and on the resulting playlists are briefly summarized.

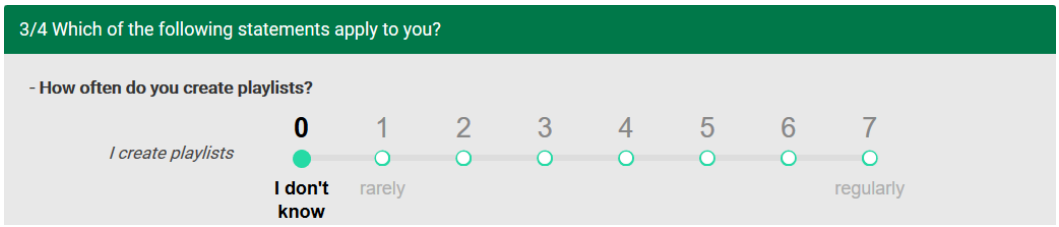
Investigating quality criteria for playlists. To determine the overall ranking of quality criteria based on the responses of the participants, we used a variant of the Borda Count rank aggregation strategy which is designed for the aggregation of partial rankings called Modified Borda Count (MBC) [Eme13], as the criteria could also be marked as irrelevant.



(a) Task 1: Order the list of quality criteria based on their relevance for your created playlist.



(b) Task 2 (only for the participants of the experimental group): Have you looked at the recommendations? If yes, how do you assess their quality?



(c) Task 3: Answer questions regarding your playlisting experience.

Figure 3.2: The questionnaire of the user study. Note that the screen captures in section (b) and (c) illustrate only the beginning (the first question) of the respective tasks.

Table 3.1: Modified Borda Count of quality criteria for playlists.

Criteria	**All**	`RecUsed`	`RecNotUsed`	`NoRec`
Homogeneity of musical features, e.g., tempo	250	68	79	103
Artist diversity	195	55	62	78
Transition	122	30	46	46
Popularity	106	39	34	33
Lyrics	95	32	34	29
Order	74	12	33	29
Freshness	32	12	11	9

The results of the overall ranking are shown in Table 3.1. Some of the interesting observations can be summarized as follows.

- *Homogeneity of musical features* like tempo or loudness along with the *artist diversity* of tracks were considered as the most relevant quality criteria.
- The *order* of the tracks in a playlist and their *freshness* appeared to be less relevant for the participants.
- The participants who used the recommendations considered *transition* as a less relevant criterion than the participants who did not use any recommendations. One explanation might be that using recommendations can reduce the effort that is needed to keep the transitions between the tracks and users therefore pay less attention to that.

Furthermore, some topic-related variations in the relevance of quality criteria for playlists were observed. For instance,

- for road-trip and hip-hop playlists, the *lyrics* aspect was more important, and
- *popularity* was considered a very important criterion only for dance playlists.

We further analyzed the collected logs and the musical features of the resulting playlists to obtain a better understanding of the effects of next-track recommendations on the playlist creation behavior of users. Although the observations that are presented in the following are not directly related to the topic of this section, they contain interesting insights on the perception and adoption of next-track recommendations, which can be relevant for evaluating music recommender systems.

Impact of recommendations on the playlist creation behavior of users.

- *Recommendations are comparably well adopted for creating playlists.* About 38% of the selected tracks by the participants of the `RecUsed` group who were presented with recommendations and actively used them were taken

from recommendations (on average, 3.2 recommendations per playlist). This strongly indicates that such a recommendation component in this domain could be useful. Note that analyzing e-commerce logs in [Jan+17b], for instance, revealed that in general e-commerce settings sometimes only every 100th click of a user is on a recommendation list.

- *Presenting recommendations increases user exploration.* On average, the participants who were presented with recommendations, played almost 1.5 times more tracks than those with no recommendation support (mean value 14.4 and 9.8, respectively). This ratio increases to more than 2.0 times, when we only consider the participants of the `RecUsed` group, who actively used the recommendations in their playlists (with the average number of 20.3 played tracks). Interestingly, the participants of the `Rec` group needed, on average, only 30 seconds more time to create the playlists. Although the number of played tracks of the participants of the `Rec` group is significantly higher than the control group (`NoRec`), the differences between the needed time to accomplish the playlist creation task were not statistically significant.[19] This indicates that the recommendations helped the participants to explore, and potentially discover, many more options in about the same time.

- *Presenting recommendations makes the playlist creation task more complex.* Comparing the self-reported difficulty of the playlist creation task for the participants of the `Rec` group and the `NoRec` group, showed that the recommendation component did not make the task easier for users but even slightly (not significantly) added to the perceived complexity. This could be caused by the more complex UI, as well as by the additional effort that users required to browse the recommendations.

Impact of recommendations on the resulting playlists of users. To analyze the impact of recommendations on the resulting playlists, we queried the musical features of the tracks of the resulting playlists through the Spotify API. Table 3.2 shows the list of these features along with their average values and standard deviations for each of the study groups.[20] The created playlists of the participants who actively used the recommendations (`RecUsed`), those who received recommendations but did not use them (`RecNotUsed`), and those of the control group with no recommendation support (`NoRec`) vary significantly in different dimensions. The most interesting observations in this context can be summarized as follows.

[19] In this study, we used the Mann-Whitney U test and the Student's t-test – both with $p < 0.05$ – to test for statistical significance for the ordinal data and the interval data, respectively.

[20] For a detailed description of the audio features listed in Table 3.2, see `https://developer.spotify.com/web-api/get-audio-features/`

Table 3.2: Average (Avg) and standard deviation (Std) of the musical features of the resulting playlists in different groups. * indicates statistical significance in comparison with the `RecUsed` group.

	`RecUsed`		`RecNotUsed`		`NoRec`	
Feature	Avg	Std	Avg	Std	Avg	Std
Acousticness	0.22	0.28	0.17*	0.23	0.17*	0.26
Danceability	0.56	0.18	0.59*	0.17	0.54	0.17
Energy	0.68	0.24	0.70	0.19	0.73*	0.23
Instrumentalness	0.16	0.32	0.12	0.28	0.12	0.27
Liveness	0.20	0.17	0.21	0.18	0.21	0.17
Loudness (dB)	-7.68	4.59	-7.60	3.67	-7.52	4.72
Popularity	50.7	21.9	55.7*	17.1	54.3*	21.3
Release year	2005	12.47	2002*	15.73	2003*	13.08
Speechiness	0.07	0.07	0.08	0.08	0.08	0.08
Tempo (BPM)	123.0	28.7	122.5	27.9	122.9	28.6
Valence	0.50	0.26	0.53	0.25	0.49	0.24

- *Popularity effect*. Using the recommended tracks significantly reduced the average popularity level of the resulting playlists.[21] This is actually in line with the observations reported in [Jan+16] where the playlist continuations generated by a commercial service were significantly less popular than the tracks users selected manually, see Section 3.2.
- *Recency effect*. Using recommendations also slightly but still significantly increases the average release year of the tracks of the resulting playlists. Accordingly, about 50% of the tracks of the playlists created by the participants who used the recommendations (`RecUsed`) were released in the last five years. This value is 40% for the `RecNotUsed` group and 34% for the `NoRec` group.
- *Mere-presence effect*. Comparing the musical features of the resulting playlists of the participants who were presented with recommendations but did not use them with the recommended tracks shows only significant difference in terms of popularity (like the resulting playlists of the participants who did use the recommendations), while the playlists of the participants without recommendation support show various significant differences with the recommendations. This similarity indicates that the subjects in the `RecNotUsed` group were biased (or inspired) by the presence of the recommendations. This is referred to in the literature as the "mere-presence" effect of recommendations and was previously investigated in a user study [Köc+16], where the participants indicated a tendency to select items with similar content to a (random) recommendation.

[21] The popularity of the tracks was determined using the Spotify API, with values between 0 and 100 (lowest to highest popularity) based on the play counts and recency of the tracks.

3.2 Evaluation Approaches

Having discussed the determination of relevant quality criteria for next-track music recommendations, this section will focus on the assessment of performance of next-track music recommendation algorithms and present the related works that have been done in the context of this thesis.

Various evaluation approaches can be found for music recommendations in general in the research literature. McFee et al. [McF+11] grouped the proposed evaluation approaches for *automated playlist generation* in the literature into the three categories of (1) *human evaluation*, (2) *semantic cohesion*, and (3) *sequence prediction*.

Human evaluation refers to user studies in which participants rate the quality of playlists generated by one or more algorithms in different dimensions, e.g., the perceived quality, diversity, or the transition between the tracks. Direct human evaluations are in principle expensive to conduct and it is also difficult to reproduce their results, see Section 3.1.2.

The evaluation approaches based on the semantic cohesion determine the quality of a generated playlist by measuring how similar the tracks in the playlist are. Different similarity measures like the co-occurrence counts of track metadata, e.g., artists [Log02; Log04], entropy of the distribution of genres within the playlist [Kne+06; Dop+08], or the distance between latent topic models of playlists [Fie+10] have been proposed in the literature. The similarity of the tracks, however, may not always be a good (or at least the only) quality criterion in real-world scenarios [Sla+06; Lee+11; Kam+12a].

The third group of evaluation approaches that were presented in [McF+11] rely on information retrieval (IR) measures. In these approaches, a playlist generation algorithm is evaluated based on its prediction. A prediction is successful, only if the predicted next track matches an observed co-occurrence in the ground truth set, e.g., the available user playlists [Pla+01; Mai+09]. In this evaluation setting, a prediction that might be interesting for the user but does not match the co-occurrences in the ground truth set will be considered as a false positive. This group of evaluation approaches will be discussed later in more detail in Section 3.2.2.

In a more recent work, Bonnin et al. [Bon+14] proposed to categorize the evaluation approaches to the *playlist continuation problem* in the four more general groups of (1) *log analysis*, (2) *objective measures*, (3) *comparison with hand-crafted playlists*, and (4) *user studies*. The following sections discuss these four approaches.

3.2.1 Log Analysis

Music platforms like Spotify analyze the listening logs that they collect, for example, through A/B tests to better understand the listening behavior of their users. Among others, these logs can be used to evaluate the acceptance of the recommendations. Although conducting field tests with real users in academia is usually not possible, there are different data sources from which information about the users' listening behavior can be obtained. For instance, listening logs of Last.fm users can be accessed through the platform's public API[22] [Bon+14].

In addition, there are some public listening logs datasets like the *#nowplaying* dataset [Pic+15], which contains information about the listening sessions collected from music-related tweets on Twitter or the *30Music* dataset [Tur+15], which contains listening sessions retrieved from Internet radio stations through the Last.fm API. In [Jan+17a], we used subsets of these two datasets to explore the value of repeated recommendations of known tracks (see Section 2.6.1). Moreover, in [Kam+17a], we used, among others, the music listening sessions to evaluate the quality of the next-track recommendations of different session-based recommending algorithms (see Section 2.5). One advantage of using listening logs for the evaluation task is the reproducibility of the results [Bon+14].

3.2.2 Objective Measures

Another evaluation approach that was discussed in [Bon+14] and [McF+11] is to use objective measures to assess the quality of next-track recommendations. The goal of such measures is to *approximate* the subjective quality perception of a user [Cre+11]. For example, in [Jan+15a] and [Jan+17a], the *inverse intra-list similarity* [Zie+05] of artists and social tags assigned to the tracks is used to assess the *diversity* level, and the *overlap* of artists and tags in the listening history and the recommended next tracks is used to quantify the *coherence* level of next-track recommendations.

Schedl et al. [Sch+17b] review the most frequently reported evaluation measures in the academic literature. They differentiate between *accuracy-related* and *beyond-accuracy* measures. For example, mean absolute error (MAE) and root mean square error (RMSE), which indicate the prediction error of a recommendation algorithm, or precision and recall, which measure the relevance of the recommendations, have been applied in the field of recommender systems to evaluate accuracy. On the other hand, *novelty*, which measures the ability of recommender systems to help users discover new items, and *serendipity*, which measures how unexpected the novel recommendations are, are examples of beyond-accuracy measures.

[22] `https://www.last.fm/api`

Accuracy-related measures can be further categorized into the metrics that evaluate the ability of recommender systems to *find good items* (e.g., precision, MAE, or RMSE), and the ones that in addition evaluate the *ranking quality* of recommender systems by considering whether or not good recommendations are positioned at the top of the recommendation lists (e.g., mean average precision (MAP), normalized discounted cumulative gain (NDCG), or mean percentile rank (MPR)) [Sch+17b].

As will be discussed in Section 3.2.4, a major limitation of evaluation approaches based on objective measures is that it is not always clear to what extent such computational quality measures correlate with the actual quality perception of music listeners [Kam+17b].

3.2.3 Comparison with Hand-Crafted Playlists

Another way to evaluate next-track recommendations is to compare them with user playlists that are, e.g., shared on music platforms. The assumption is that users, in principle, select the tracks to be added to their playlists with respect to specific quality criteria and a recommending algorithm will therefore achieve a better performance by generating recommendations that are similar to the tracks selected by real users [Bon+14].

A typical evaluation protocol in this context is to take a hand-crafted playlist, hide a subset of its tracks, and let the recommendation algorithms predict the hidden tracks. The "hit rate" of a recommendation algorithm is then computed by counting the number of times that the hidden tracks appear in its top-n recommendation lists [Har+12; Bon+13; Bon+14; Jan+16]. Another measure based on the comparison of next-track recommendations with hand-crafted playlists is "average log-likelihood" which can be used to assess how likely a system is to generate the tracks of given playlists or listening sessions [McF+11; Che+12; Moo+12].

We applied this evaluation approach in [Jan+16], which is one of the papers included in this thesis by publication, to compare a number of academic next-track recommendation approaches and a commercial playlisting service in different dimensions. The academic algorithms that were considered for the experiment are described earlier in the context of next-track recommendation algorithms and included an artist-based approach named CAGH (described in Section 2.3); a kNN-based approach (described in Section 2.3 and Figure 2.1); a content-based technique based on social tags (described in Section 2.1.2); and a weighted hybridization of the kNN method, the content-based method and additional suitability scores (explained in Section 2.6.2). As a commercial service, we used the recommender of *the.echonest.com*, which is now a subsidiary company of Spotify.

More than 10,000 manually created playlists – collected from three music platforms – were used to evaluate these algorithms. Our goal was to compare generated playlist continuations with those made by users. For this experimental setting, hiding only one last track of a playlist would be insufficient. We therefore split the playlists in the test set into two halves as shown in Figure 3.3. The *seed* half is used by the recommender to generate a continuation with the same size as the seed half. The generated continuation will then be compared with the *test* half to measure how good the algorithm was able to select similar tracks to the ones that were selected by the playlist creator.

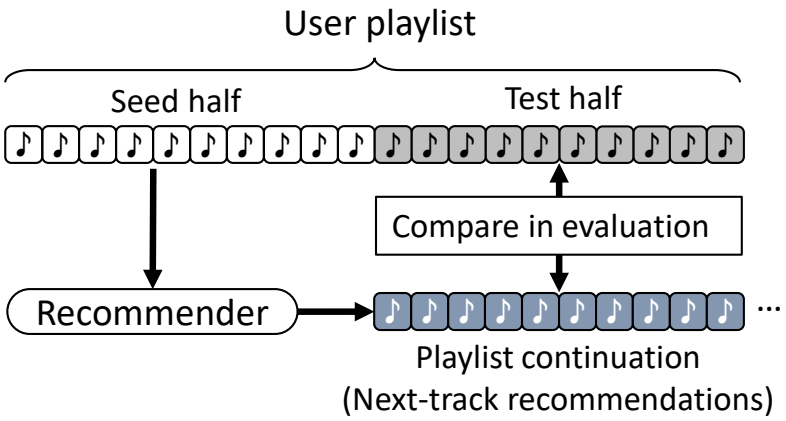


Figure 3.3: The evaluation protocol proposed in [Jan+16].

In our multi-dimensional comparison in [Jan+16], we searched for answers to the following two general questions.

1. To which extent can different algorithms recommend (1) the *right tracks*, (2) the *relevant artists*, (3) the *correct genres*, and (4) the tracks with *suitable tags*?

2. To which extent can the algorithms produce continuations that are coherent with the playlist beginnings in terms of different musical features?

Concerning the first question, which mainly deals with the accuracy performance of the algorithms, an interesting observation was that the commercial recommender led, in most cases, to the lowest precision and recall values. This can be an indication that the playlists that are generated by the commercial service are not necessarily (exclusively) optimized for precision or recall and that also other criteria govern the track selection process. Another observation according to the accuracy results was that the comparably simple CAGH method led to competitive accuracy results especially in cases where the goal is to play music of similar artists, related genres or to find tracks that are similar in terms of their social tags.

Analyses done, for instance, in [Jan+15c], showed that algorithms that lead to high accuracy results often are biased towards popular items. Recommending popular items, however, limits the discovery potential of algorithms and leads to recommendations that are not necessarily coherent with the users' recent listening history and preferences. We, therefore, in addition to accuracy, stated the second question and analyzed the *coherence* of the generated continuations (next-track recommendations) with the playlist beginnings in terms of some selected music features including popularity, loudness, tempo, and release year of the tracks.

To answer the second question regarding the coherence of the generated continuations of the algorithms, we looked at the *mean* and *distributions* of the feature values in the seed halves, test halves, and the generated continuations. In general, our analyses showed that users prefer playlist continuations (test halves) that are coherent with the first halves, however, for some features, like tempo or release years, all recommenders were able to mimic the user's behavior and for some other features, like popularity or loudness, the algorithms showed strong biases. More precisely, the explored academic recommenders focused on more popular tracks and the variability of the generated continuations of these algorithms in terms of popularity was higher than the user-created test halves. In contrast, the commercial service recommended less popular tracks and reduced the loudness and popularity diversity in the generated continuations more than the users do.

3.2.4 User Studies

In Section 3.1.2, we discussed that one reliable way to determine the relevant quality criteria for next-track recommendations is to conduct user studies. User studies can also be applied as an evaluation approach for the quality of next-track recommendations. For instance, Barrington et al. [Bar+09] compared Apple's Genius collaborative filtering based system with a recommendation method based on artist similarity in a user study. In their experimental setting, they hid the artist and track information in one condition and displayed this information in another. An interesting insight was that the recommendations of the Genius system were perceived better when the information was hidden, whereas the artist-based recommendations were selected as the better ones in the other case.

Despite the growing popularity of user-centered evaluation approaches for recommender systems which have led to the development of new evaluation frameworks [Pu+11; Kni+12], laboratory or online studies on music recommendation and in particular on next-track recommendation are comparably rare. For the domain of Music Information Retrieval, Lee et al. [Lee+16] recently discussed the limitations of the current research practice in the field and stressed the importance of user-centered evaluation approaches.

In [Kam+17b], which is one of the papers included in this thesis by publication, we addressed this research gap and conducted a user study involving 277 participants to determine the user's quality perception of different next-track recommendation algorithms. In the remainder of this section, we present the details of this online user experiment.

Lessons learned from offline experiments. Our main goal was to validate some of the insights that were obtained from offline experiments by utilizing an online experiment with real users. Specifically, the following offline observations from [Har+12; Bon+14; Jan+15a; Jan+16]; and [Jan+17a] were selected to be tested in our user study.

1. Hand-crafted playlists can be used as a reference to evaluate the performance of next-track recommending techniques.

2. Methods based on k-nearest-neighbor approach represent a strong baseline in terms of the recall of the next-track recommendations.

3. A simple artist-based approach called CAGH, which recommends the greatest hits of user' favorite artists and similar artists to them, leads to competitive accuracy results.

4. Considering additional signals, e.g., musical features in combination with the track co-occurrences captured by the kNN method, can lead to further accuracy improvements.

Study design. We created a web application to validate these offline observations in an online experiment with users. At the welcome page of the user study application, along with a brief description of the general purpose of music recommender systems, the tasks that should be done by the participants are described, see Figure 3.4.

In the first step, the participants had to listen to four tracks of a selected playlist. To minimize the familiarity bias, information about the artists and tracks were hidden, see Figure 3.5(a). When the participants had listened to the tracks, they had to answer five questions about the *emotion*, *energy*, *topic*, *genre*, and *tempo* of the tracks. Using 7-point Likert items, the participants should state how similar the tracks of the playlists were in any of the these dimensions, see Figure 3.5(b). Next, the participants were presented with four alternative continuations for the given playlist from task 1. The recommended next tracks were also anonymized and displayed in randomized order across participants to avoid any order bias. The participants should state how well each track matches the playlist as its next track and indicate if

they liked the song and if they knew the artist of the track, the track itself, or both, see Figure 3.5(c). Finally, the participants were asked questions regarding their age group and music experience.

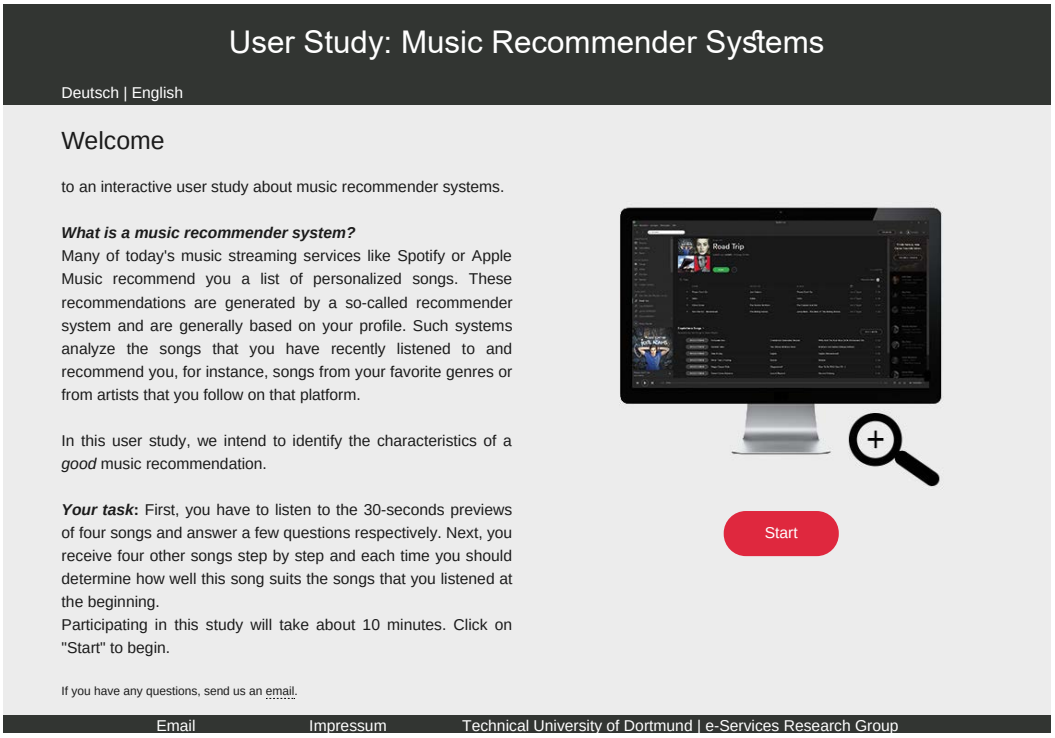


Figure 3.4: Welcome screen of the user study.

In each trial, one playlist was randomly assigned to the participants. One main question when designing the study was how to select the seed playlists for which we wanted to evaluate the next-track recommendations. Since we aimed to analyze whether hand-crafted playlists are suitable for evaluating the recommending techniques, we chose five hand-crafted playlists. To assess if the choice of the most suitable next track is influenced by certain characteristics of the playlists, we selected these playlists in a way that each one was very *homogeneous* in one dimension.

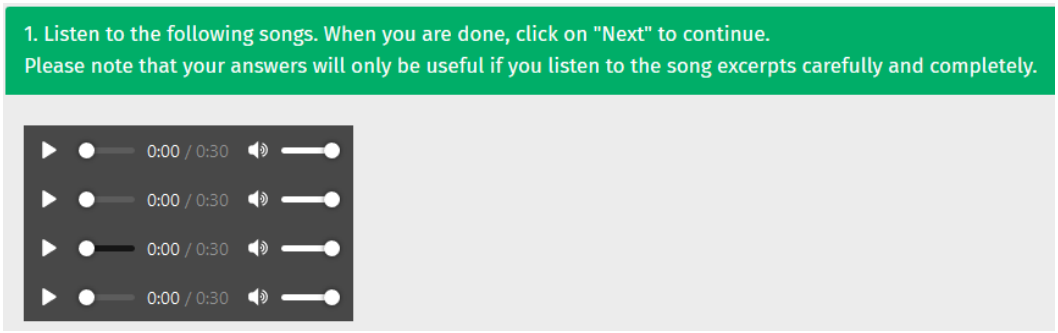
1. *Topic-playlist*. This playlist was organized around the topic *Christmas* with pop songs from the 70s and 80s (Table 3.3, section (1)).

2. *Genre-playlist*. This playlist contained tracks of the genre “soul” (Table 3.3, section (2)).

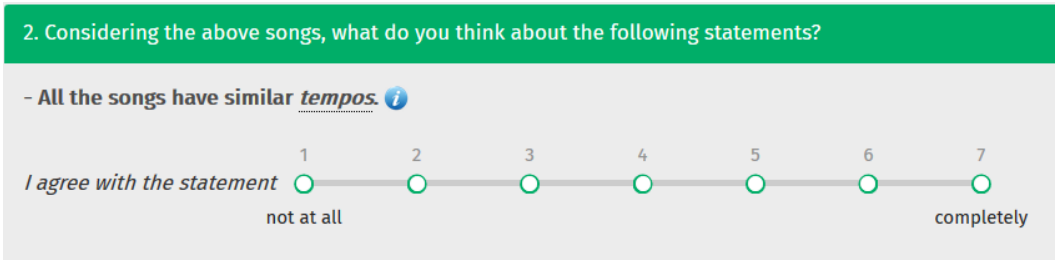
3. *Mood-playlist*. This playlists included tracks with *romantic* lyrics (Table 3.3, section (3)).

4. *Tempo-playlist*. The tracks of this playlists had similar (allegro) *tempo*. The average tempo of the playlist is 125 bpm with a standard deviation of 2 bpm (Table 3.3, section (4)).

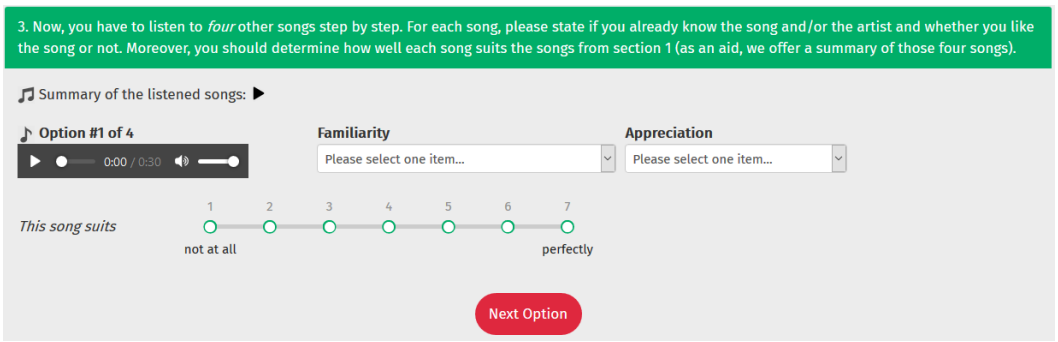
5. *Energy-playlist*. This playlist contained tracks with homogeneous *energy* levels. The average energy value of the tracks was at 0.86 on a scale of 0 to 1, with a standard deviation of 0.02 (Table 3.3, section (5)).



(a) Task 1: Listen to the tracks of a playlist.



(b) Task 2: Determine the similarity of the tracks of the given playlist.



(c) Task 3: Evaluate the suitablity of each alternative next track for the given playlist.

Figure 3.5: The tasks of the user study in [Kam+17b]. Note that sections (b) and (c) of this figure show only the beginning of the respective tasks.

Table 3.3: Selected hand-crafted playlists for the experiments in [Kam+17b]. Each section of the table consists of the tracks of the five chosen playlists (Track #1 to 4) followed by the four presented next-track recommendations. The last column of the table shows the dominating characteristic of each playlists respectively.

(1) Topic-Playlist			
	Title	**Artist**	**Top Tags**
Track #1	*Do They Know It's Christmas*	*Band Aid*	Xmas, 80s, Pop, Rock, ...
Track #2	*Happy Xmas (War Is Over)*	*John Lennon*	Xmas, Rock, Pop, 70s, ...
Track #3	*Thank God It's Christmas*	*Queen*	Xmas, Rock, 80s, ...
Track #4	*Driving Home For Christmas*	*Chris Rea*	Xmas, Rock, Pop, 80s, ...
Hidden Track	*White Christmas*	*Bing Crosby*	Xmas, Oldies, Jazz, ...
CAGH	*Bohemian Rhapsody*	*Queen*	Rock, Epic, British, 70s, ...
kNN	*Santa Baby*	*Eartha Kitt*	Xmas, Jazz, 50s, ...
kNN+X	*Step Into Christmas*	*Elton John*	Xmas, Pop, Piano, 70s, ...
(2) Genre-Playlist			
	Title	**Artist**	**Artist Genres**
Track #1	*The Dark End Of The Street*	*James Carr*	Soul, Motown, Soul Blues, ...
Track #2	*I Can't Stand The Rain*	*Ann Peebles*	Soul, Motown, Soul Blues, ...
Track #3	*Because Of You*	*Jackie Wilson*	Soul, Motown, Soul Blues, ...
Track #4	*Mustang Sally*	*Wilson Pickett*	Soul, Motown, Soul Blues, ...
Hidden Track	*Cigarettes And Coffee*	*Otis Redding*	Soul, Motown, Soul Blues, ...
CAGH	*In The Midnight Hour*	*Wilson Pickett*	Soul, Motown, Soul Blues, ...
kNN	*Ever Fallen In Love*	*Thea Gilmore*	Folk-Pop, New Wave Pop, ...
kNN+X	*I Can't Get Next To You*	*Al Green*	Soul, Motown, Soul Blues, ...
(3) Mood-Playlist			
	Title	**Artist**	**Mood**
Track #1	*Memory Motel*	*The Rolling Stones*	Romantic
Track #2	*Harvest Moon*	*Neil Young*	Romantic
Track #3	*Full Of Grace*	*Sarah McLachlan*	Romantic
Track #4	*Shiver*	*Coldplay*	Romantic
Hidden Track	*Beast Of Burden*	*The Rolling Stones*	Romantic
CAGH	*Yellow*	*Coldplay*	Romantic
kNN	*Here It Comes*	*Doves*	Calm
kNN+X	*Twilight*	*Elliott Smith*	Romantic

Table 3.3: Selected hand-crafted playlists – continued from previous page

(4) Tempo-Playlist			
	Title	**Artist**	**Tempo (bpm)**
Track #1	*Everything In Its Right Place*	*Radiohead*	124.0
Track #2	*The Crystal Lake*	*Grandaddy*	126.0
Track #3	*Imitation Of Life*	*R.E.M.*	128.7
Track #4	*Buggin'*	*The Flaming Lips*	123.2
Hidden Track	*Paper Thin Walls*	*Modest Mouse*	126.7
CAGH	*Fake Plastic Trees*	*Radiohead*	73.5
kNN	*Pyramid Song*	*Radiohead*	77.1
kNN+X	*Brilliant Disguise*	*Bruce Springsteen*	126.3
(5) Energy-Playlist			
	Title	**Artist**	**Energy**
Track #1	*Wild At Heart*	*Gloriana*	0.85
Track #2	*Feel That Fire*	*Dierks Bentley*	0.83
Track #3	*Summer Nights*	*Rascal Flatts*	0.88
Track #4	*My Kinda Party*	*Jason Aldean*	0.87
Hidden Track	*Days Go By*	*Keith Urban*	0.86
CAGH	*What Hurts The Most*	*Rascal Flatts*	0.63
kNN	*Oh It Is Love*	*Hellogoodbye*	0.35
kNN+X	*It Had To Be You*	*Motion City Soundtrack*	0.87

Considering the offline insights that were discussed earlier in this section, the four alternative tracks to be played next in each trial were selected using the following approaches.

1. **Hidden Track**. In each trial, we presented the first four tracks of the chosen hand-crafted playlist to the participants. One alternative to continue this playlist was the actual fifth track of the playlist that was originally chosen by the playlist creator which is referred to as "hidden track" in the experiment.

2. **CAGH**. To assess the effect of recommending popular tracks on the user's quality perception, one of the recommended next track in each trial was selected by the CAGH method which recommends the greatest hits of certain artists, see Section 2.3.

3. k**NN**. To validate the prediction accuracy of nearest-neighbor methods in an online experiment, we included the next-track recommendations of a kNN-based method which takes the playlist beginning and looks for other playlists in the training data that contain the same track, see Figure 2.1.

4. k**NN+X**. To assess the value of incorporating additional features into the recommendation process with respect to the user's quality perception and validate if users prefer more homogeneous playlists, we selected one alternative continuation to the given playlist using a hybrid method from our offline experiments. This hybridization combines the kNN method as a baseline with the dominating feature of the respective playlist like topic, emotion, or genre, see Section 2.6.2.

To determine the ranking of the investigated techniques regarding the suitability of their recommendations, we used two aggregation strategies.

1. **Winning frequency**. We count how often the recommendations of an approach were considered as the most suitable continuation.

2. **Borda count**. We apply the Borda count measure [Eme13] to aggregate the rankings of all four alternatives. The responses provided by the participants are used here as implicit ranking information.

Furthermore, to investigate to what extent familiarity aspects may affect the results, in addition to considering the rankings of all trials, we also reported the results for only those trials in which the participants explicitly indicated that they did not know the track that they selected as the most suitable track, i.e., 70% of all trials. We refer to the former configuration setting as "All Tracks" configuration and to the latter setting as "Novel Tracks" configuration in our experiment. Table 3.4 summarizes the overall ranking results.

Table 3.4: Overall ranking results of the next-track recommending techniques with respect to the users' quality perception based on winning frequency (WF) and Borda count (BC) [Kam+17b].

	All Tracks		**Novel Tracks**	
Strategy	**WF**	**BC**	**WF**	**BC**
Hidden Track	41%	645	43%	580
CAGH	46%	649	32%	403
k**NN**	25%	520	19%	477
k**NN+X**	30%	594	36%	631

Results. Several observations from offline studies were reproduced in this user study. Regarding the insights obtained from offline experiments that were mentioned earlier in this section, we categorized our observations into the following four groups.

1. *Hand-crafted playlists are suitable for the evaluation of playlisting algorithms.* The hidden tracks of the given playlists, i.e., the tracks that were originally picked by the playlist creator, were in a considerable number of trials (41% in the all-tracks setting and 43% in the novel-tracks setting) selected as the most suitable continuation by the participants. Note that the difference between the CAGH method and the hidden track strategy in the all-tracks setting is not statistically significant.

2. *Users prefer recommendations that are more coherent with their recently played tracks.* The perceived quality of the recommendations of the hybrid method, which are more coherent with recently played tracks in terms of the dominating characteristic of the seed playlist, were considered significantly more suitable than the recommendations that are only based on track-co-occurrence patterns in both configurations and on both measures.

3. *Popularity-based approaches can be considered as a* safe *strategy.* We observed that the popularity-based method, CAGH, fared well in terms of the perceived quality, particularly in the all-tracks setting, where familiar tracks are also considered. This is in line with our offline observations in [Jan+16], where the CAGH method led to competitive results in comparison with more complicated recommendation algorithms, see Section 3.2.3. When applying such popularity-based approaches in practice, however, the limited discovery potential of such techniques should be taken into account.

4. *Users consider familiar recommendations as more suitable.* Measurable differences between the rankings of the alternatives can be observed when known or novel tracks are considered (all-tracks versus novel-tracks settings). For instance, the CAGH method that recommends the most familiar tracks among other algorithms, is perceived as the best next-track recommending algorithm when all tracks are considered but its recommendations are not well received anymore when familiar tracks are excluded from the analyses.

It should also be noted that the ranking of the algorithms could vary from the overall ranking results when the trials with a particular playlist are considered. For instance, while the kNN+X recommendations were generally ranked higher than those of the kNN method, for the tempo-oriented playlist, the kNN and CAGH methods were, on average, ranked higher than the kNN+X method. This could be interpreted as less relevance of the tempo than other characteristics like artist homogeneity, which was also observed in [Jan+14], see Section 3.2.3.

4 Conclusion

Next-track music recommendation, defined as the recommendation of a list of tracks to be played next given a user's recently played tracks, is a specific form of music recommendation that can be found on most of today's music applications, such as Spotify or Deezer. In addition to commercial solutions to the next-track recommendation problem, such as playlist creation support tools or personalized virtual radio stations, different academic studies have also addressed this problem in recent years. The algorithmic approaches proposed in the research literature rely mainly on contextual information about users or the musical features and metadata information of tracks. Due to the specific characteristics of music like subjectiveness and context-dependency, researchers in this field encounter particular challenges.

This thesis by publication discussed the advances in next-track music recommendation in both academia and industry. The reviewed research literature provided a general overview of algorithmic approaches to generate next-track recommendations and to evaluate them. Furthermore, the publications included in this thesis addressed several crucial challenges in this domain, such as the personalization and multi-dimensional optimization of next-track recommendations as well as the evaluation of the quality perception of different recommending techniques. This chapter will focus on providing a summary of the discussed issues in this thesis and sketching possible directions for future work.

4.1 Summary

The first chapter of this thesis introduced a brief history of music along with a general characterization of the music recommendation problem. The next-track recommendation scenario was then presented in this chapter. Moreover, the research questions that this thesis aimed to answer were categorized and briefly discussed.

The algorithmic approaches that have been proposed for next-track recommendation in the research literature were reviewed in the second chapter of this thesis. Particularly, content-based filtering approaches, collaborative filtering methods, frequent

pattern mining techniques, and sequence-aware algorithms were discussed and a number of published works on each topic were introduced. Afterwards, the results of comparing a number of these approaches in different dimensions, such as accuracy, popularity bias, and computational complexity that was conducted in the context of this thesis were presented.

With respect to the challenges of next-track music recommendation, two publications of the author of this thesis were presented in this chapter. First, an algorithmic proposal on how to leverage the users' long-term preference signals for personalizing next-track music recommendations. Second, a two-phase recommendation-optimization approach to recommend more accurate recommendations and to optimize the selection of next tracks based on the user's individual tendency towards different quality factors, e.g., artist diversity.

The evaluation of next-track recommendations was the topic of the third chapter of this thesis. A critical question in this regard is how to determine quality criteria for next-track recommendations. One way to do this is to analyze the characteristics of playlists that are created and shared by users based on musical and metadata features of the tracks. An experimental analysis of 10,000 hand-crafted playlists in [Jan+14], for instance, revealed that features like freshness, popularity, and homogeneity of the tracks are relevant for users. The insights from such analyzes should help researchers design algorithms that recommend more *natural* next tracks. Another way to determine the relevant quality criteria is to conduct user studies. As an example, a user study that was conducted recently in the context of this thesis was presented in this chapter. The findings of this study involving 123 subjects indicated that the homogeneity of musical features, such as tempo and energy, along with the artist diversity are important characteristics for playlists and should be considered when recommending next tracks, e.g., for supporting playlist construction.

Finally, different evaluation approaches were reviewed. Among others, comparing next-track recommendations with hand-crafted playlists and conducting user studies were discussed based on two publications that were included in this thesis. Regarding the former approach, the results of a multi-dimensional comparison of next-track recommendations of different academic algorithms and a commercial service were presented. And, for the latter approach, the results of a user study were presented which was designed to investigate the quality perception of playlist continuation proposals generated by different next-track music recommendation techniques.

4.2 Perspectives

Schedl et al. [Sch+17b] identify *the creation of more personalized recommendations*, which was also addressed in this thesis, as the future direction of music recommender systems. In this regard, the authors mention three aspects that could influence the next generation of music recommender systems.

The first aspect relates to *psychological* factors. They argue that despite the indicated effect of *personality* and *emotion* on music tastes [Fer+15; Sch+17a], "psychologically-inspired" music recommender systems have not been investigated to a large extent so far.

Another aspect that can affect the future of personalized music recommender systems is the incorporation of *situational signals* into the recommendation process. Although several academic works have explored the value of situational information like location or time of the day in music recommender systems [Bal+11; Wan+12; Kam+13; Che+14], such signals have not been integrated in large scale commercial systems yet.

The last research perspective for music recommender systems that was discussed in Schedl et al. [Sch+17b] relates to *cultural* aspects like language, religion, or history. The idea is to study the impact of cultural backgrounds and differences on the listening behavior of users, as done for instance in Schedl [Sch17], and to build cultural user models that can be integrated into recommender systems.

The publications included in this thesis utilized different musical features of the tracks to infer the underlying *theme* of a playlist or listening session as a basis for generating or evaluating next-track recommendations. The selection of the musical features was, however, limited to publicly available data. A future work in this regard would be to acquire and exploit additional information about the tracks and artists that could help to reach a better understanding of the desired characteristics of the seed tracks and to enhance the quality of next-track recommendations.

Another aspect of next-track music recommendation that is not fully investigated in the research field is the external validity of insights that are obtained through offline experiments. In this thesis, we aimed to answer some of the open questions with respect to, e.g., the correlation between offline quality measures like precision and recall with the quality perception of music listeners, or the impact of next-track music recommendations on the user's listening behavior. Questions regarding, for example, the perceived quality of *personalized* next-track recommendations are, however, still open.

Bibliography

[Ado+12] Gediminas Adomavicius and YoungOk Kwon. “Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques”. In: *IEEE Transactions on Knowledge and Data Engineering* 24.5 (May 2012), pp. 896–911 (cit. on p. 28).

[Agr+95] Rakesh Agrawal and Ramakrishnan Srikant. “Mining Sequential Patterns”. In: *Proceedings of the Eleventh International Conference on Data Engineering*. ICDE ’95. 1995, pp. 3–14 (cit. on p. 20).

[Aiz+12] Natalie Aizenberg, Yehuda Koren, and Oren Somekh. “Build Your Own Music Recommender by Modeling Internet Radio Streams”. In: *Proceedings of the 21st international conference on World Wide Web*. 2012, pp. 1–10 (cit. on p. 16).

[And+14] Ashton Anderson, Ravi Kumar, Andrew Tomkins, and Sergei Vassilvitskii. “The Dynamics of Repeat Consumption”. In: *Proceedings of the 23rd International Conference on World Wide Web*. WWW ’14. 2014, pp. 419–430 (cit. on p. 26).

[Bal+11] Linas Baltrunas, Marius Kaminskas, Bernd Ludwig, Omar Moling, Francesco Ricci, Aykan Aydin, et al. “InCarMusic: Context-Aware Music Recommendations in a Car”. In: *E-Commerce and Web Technologies* (2011), pp. 89–100 (cit. on p. 57).

[Ban+16] Trapit Bansal, David Belanger, and Andrew McCallum. “Ask the GRU: Multi-task Learning for Deep Text Recommendations”. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys ’16. 2016, pp. 107–114 (cit. on p. 20).

[Bar+09] Luke Barrington, Reid Oda, and Gert R. G. Lanckriet. “Smarter than Genius? Human Evaluation of Music Recommender Systems”. In: *Proceedings of the 10th International Society for Music Information Retrieval Conference*. 2009, pp. 357–362 (cit. on p. 47).

[Bau+10] Dominikus Baur, Sebastian Boring, and Andreas Butz. “Rush: Repeated Recommendations on Mobile Devices”. In: *Proceedings of the 15th International Conference on Intelligent User Interfaces*. IUI ’10. 2010, pp. 91–100 (cit. on p. 36).

[Ben+07] James Bennett, Stan Lanning, et al. “The Netflix Prize”. In: *Proceedings of KDD Cup and Workshop*. 2007, p. 35 (cit. on p. 27).

[Ben09] Yoshua Bengio. “Learning Deep Architectures for AI”. In: *Foundations and trends in Machine Learning* 2.1 (2009), pp. 1–127 (cit. on p. 16).

[Blu+99] T.L. Blum, D.F. Keislar, J.A. Wheaton, and E.H. Wold. *Method and Article of Manufacture for Content-Based Analysis, Storage, Retrieval, and Segmentation of Audio Information*. US Patent 5,918,223. 1999 (cit. on p. 16).

[Bog+10] Dmitry Bogdanov, M. Haro, Ferdinand Fuhrmann, Emilia Gómez, and Perfecto Herrera. “Content-Based Music Recommendation Based on User Preference Examples”. In: *The 4th ACM Conference on Recommender Systems. Workshop on Music Recommendation and Discovery*. 2010 (cit. on pp. 15, 16).

[Bog+11] Dmitry Bogdanov and Perfecto Herrera. “How Much Metadata Do We Need in Music Recommendation? A Subjective Evaluation Using Preference Sets.” In: *Conference of the International Society for Music Information Retrieval*. 2011, pp. 97–102 (cit. on p. 16).

[Bog13] Dmitry Bogdanov. “From Music Similarity to Music Recommendation: Computational Approaches Based on Audio Features and Metadata”. PhD thesis. Barcelona, Spain: Universitat Pompeu Fabra, 2013, p. 227 (cit. on p. 15).

[Bon+13] Geoffray Bonnin and Dietmar Jannach. “Evaluating the Quality of Generated Playlists Based on Hand-Crafted Samples”. In: *Proceedings of the 14th International Society for Music Information Retrieval Conference*. 2013, pp. 263–268 (cit. on p. 45).

[Bon+14] Geoffray Bonnin and Dietmar Jannach. “Automated Generation of Music Playlists: Survey and Experiments”. In: *ACM Computing Surveys* 47.2 (2014), 26:1–26:35 (cit. on pp. 5, 8, 9, 19, 20, 22, 33, 36, 43–45, 48).

[Bra+01] Keith Bradley and Barry Smyth. “Improving Recommendation Diversity”. In: *Proceedings of the Twelfth Irish Conference on Artificial Intelligence and Cognitive Science*. 2001, pp. 85–94 (cit. on p. 28).

[BS+15] David Ben-Shimon, Alexander Tsikinovsky, Michael Friedmann, Bracha Shapira, Lior Rokach, and Johannes Hoerle. “RecSys Challenge 2015 and the YOOCHOOSE Dataset”. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. RecSys ’15. 2015, pp. 357–358 (cit. on p. 24).

[Bud+12] Karan Kumar Budhraja, Ashutosh Singh, Gautav Dubey, and Arun Khosla. “Probability Based Playlist Generation Based on Music Similarity and User Customization”. In: *National Conference on Computing and Communication Systems*. 2012, pp. 1–5 (cit. on p. 17).

[Bur02] Robin Burke. “Hybrid Recommender Systems: Survey and Experiments”. In: *User Modeling and User-Adapted Interaction* 12.4 (Nov. 2002), pp. 331–370 (cit. on p. 18).

[Can+04] Pedro Cano and Markus Koppenberger. “The Emergence of Complex Network Patterns in Music Artist Networks”. In: *Proceedings of the 5th International Symposium on Music Information Retrieval*. 2004, pp. 466–469 (cit. on p. 18).

[Cas+08] Michael A. Casey, Remco Veltkamp, Masataka Goto, Marc Leman, Christophe Rhodes, and Malcom Slaney. “Content-Based Music Information Retrieval: Current Directions and Future Challenges”. In: *Proceedings of the IEEE* 96.4 (2008), pp. 668–696 (cit. on pp. 5, 16).

[Cel08] Òscar Celma. “Music Recommendation and Discovery in the Long Tail”. PhD thesis. Barcelona: Universitat Pompeu Fabra, 2008 (cit. on pp. 17, 18).

[Cel10] Òscar Celma. *Music Recommendation and Discovery - The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010 (cit. on p. 8).

[Che+12] Shuo Chen, Josh L. Moore, Douglas Turnbull, and Thorsten Joachims. “Playlist Prediction via Metric Embedding”. In: *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’12. 2012, pp. 714–722 (cit. on pp. 15, 25, 45).

[Che+14] Zhiyong Cheng and Jialie Shen. “Just-for-Me: An Adaptive Personalization System for Location-Aware Social Music Recommendation”. In: *Proceedings of International Conference on Multimedia Retrieval*. ICMR ’14. 2014, 185:185–185:192 (cit. on p. 57).

[Che+16] Chih-Ming Chen, Ming-Feng Tsai, Yu-Ching Lin, and Yi-Hsuan Yang. “Query-based Music Recommendations via Preference Embedding”. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys ’16. 2016, pp. 79–82 (cit. on p. 5).

[Cli06] Dave Cliff. “hpDJ: An Automated DJ with Floorshow Feedback”. In: *Consuming Music Together: Social and Collaborative Aspects of Music Consumption Technologies*. Ed. by Kenton O’Hara and Barry Brown. Dordrecht: Springer Netherlands, 2006, pp. 241–264 (cit. on p. 6).

[Coe+13] Filipe Coelho, José Devezas, and Cristina Ribeiro. “Large-scale Crossmedia Retrieval for Playlist Generation and Song Discovery”. In: *Proceedings of the 10th Conference on Open Research Areas in Information Retrieval*. OAIR ’13. 2013, pp. 61–64 (cit. on p. 16).

[Coh+00] William W. Cohen and Wei Fan. “Web-collaborative Filtering: Recommending Music by Crawling the Web”. In: *Computer Networks* 33.1-6 (June 2000), pp. 685–698 (cit. on p. 18).

[Cov+16] Paul Covington, Jay Adams, and Emre Sargin. “Deep Neural Networks for YouTube Recommendations”. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys ’16. 2016, pp. 191–198 (cit. on p. 20).

[Cre+11] Paolo Cremonesi, Franca Garzotto, Sara Negro, Alessandro Vittorio Papadopoulos, and Roberto Turrin. “Looking for “Good” Recommendations: A Comparative Evaluation of Recommender Systems”. In: *Human-Computer Interaction – INTERACT 2011: 13th IFIP TC 13 International Conference, Lisbon, Portugal, September 5-9, 2011, Proceedings, Part III*. Ed. by Pedro Campos, Nicholas Graham, Joaquim Jorge, Nuno Nunes, Philippe Palanque, and Marco Winckler. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 152–168 (cit. on p. 44).

[Cre+12] Paolo Cremonesi, Franca Garzotto, and Roberto Turrin. “Investigating the Persuasion Potential of Recommender Systems from a Quality Perspective: An Empirical Study”. In: *ACM Transactions on Interactive Intelligent Systems* 2.2 (June 2012), 11:1–11:41 (cit. on p. 27).

[Cun+06] Sally Jo Cunningham, David Bainbridge, and Annette Falconer. “More of an Art than a Science: Supporting the Creation of Playlists and Mixes”. In: *Proceedings of 7th International Conference on Music Information Retrieval*. 2006, pp. 240–245 (cit. on pp. 35, 36).

[Cun+07] Sally Jo Cunningham, David Bainbridge, and Dana McKay. “Finding New Music: A Diary Study of Everyday Encounters with Novel Songs”. In: *Proceedings of the 8th International Conference on Music Information Retrieval*. 2007, pp. 83–88 (cit. on p. 36).

[Die+14] Sande Dieleman and Benjamin Schrauwen. “End-to-End Learning for Music Audio”. In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2014, pp. 6964–6968 (cit. on p. 16).

[Dop+08] Markus Dopler, Markus Schedl, Tim Pohle, and Peter Knees. “Accessing Music Collections Via Representative Cluster Prototypes in a Hierarchical Organization Scheme”. In: *Conference of the International Society for Music Information Retrieval*. 2008, pp. 179–184 (cit. on p. 43).

[Dow03] J. Stephen Downie. “Music Information Retrieval”. In: *Annual Review of Information Science and Technology* 37.1 (2003), pp. 295–340 (cit. on p. 36).

[Eke+00] Robert B. Ekelund Jr, George S. Ford, and Thomas Koutsky. “Market Power in Radio Markets: An Empirical Analysis of Local and National Concentration”. In: *The Journal of Law and Economics* 43.1 (2000), pp. 157–184 (cit. on p. 6).

[Eks+14] Michael D. Ekstrand, F. Maxwell Harper, Martijn C. Willemsen, and Joseph A. Konstan. “User Perception of Differences in Recommender Algorithms”. In: *Proceedings of the 8th ACM Conference on Recommender Systems*. RecSys ’14. 2014, pp. 161–168 (cit. on p. 36).

[Elk+15] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. “A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems”. In: *Proceedings of the 24th International Conference on World Wide Web*. WWW ’15. 2015, pp. 278–288 (cit. on p. 20).

[Eme13] Peter Emerson. “The Original Borda Count and Partial Voting”. In: *Social Choice and Welfare* 40.2 (2013), pp. 353–358 (cit. on pp. 38, 53).

[Fer+15] Bruce Ferwerda, Markus Schedl, and Marko Tkalcic. “Personality & Emotional States: Understanding Users’ Music Listening Needs”. In: *Posters, Demos, Late-breaking Results and Workshop Proceedings of the 23rd Conference on User Modeling, Adaptation, and Personalization*. 2015 (cit. on p. 57).

[Fie+10] Ben Fields, Christophe Rhodes, Mark d’Inverno, et al. “Using Song Social Tags and Topic Models to Describe and Compare Playlists”. In: *1st Workshop On Music Recommendation And Discovery*. 2010 (cit. on p. 43).

[Gra+14] Alex Graves, Greg Wayne, and Ivo Danihelka. “Neural Turing Machines”. In: *CoRR* abs/1410.5401 (2014) (cit. on p. 20).

[Grb+15] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, et al. “E-commerce in Your Inbox: Product Recommendations at Scale”. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’15. 2015, pp. 1809–1818 (cit. on p. 18).

[Har+12] Negar Hariri, Bamshad Mobasher, and Robin Burke. “Context-aware Music Recommendation Based on Latenttopic Sequential Patterns”. In: *Proceedings of the Sixth ACM Conference on Recommender Systems*. RecSys ’12. 2012, pp. 131–138 (cit. on pp. 8, 19, 20, 45, 48).

[Hid+15] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. “Session-Based Recommendations with Recurrent Neural Networks”. In: *CoRR* abs/1511.06939 (2015) (cit. on pp. 20–23).

[Hid+17] Balázs Hidasi and Alexandros Karatzoglou. “Recurrent Neural Networks with Top-k Gains for Session-Based Recommendations”. In: *CoRR* abs/1706.03847 (2017) (cit. on p. 20).

[Hin+06] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. “A Fast Learning Algorithm for Deep Belief Nets”. In: *Neural Computation* 18.7 (2006), pp. 1527–1554 (cit. on p. 16).

[Hum+12] Eric J. Humphrey, Juan Pablo Bello, and Yann LeCun. “Moving Beyond Feature Design: Deep Architectures and Automatic Feature Learning in Music Informatics.” In: *Proceedings of the 13th International Conference on Music Information Retrieval*. 2012, pp. 403–408 (cit. on p. 16).

[Jam+10] Tamas Jambor and Jun Wang. “Optimizing Multiple Objectives in Collaborative Filtering”. In: *Proceedings of the Fourth ACM Conference on Recommender Systems*. RecSys ’10. 2010, pp. 55–62 (cit. on p. 30).

[Jan+12] Dietmar Jannach, Markus Zanker, Mouzhi Ge, and Marian Gröning. “Recommender Systems in Computer Science and Information Systems - A Landscape of Research”. In: *13th International Conference on Electronic Commerce and Web Technologies*. 2012, pp. 76–87 (cit. on p. 27).

[Jan+14] Dietmar Jannach, Iman Kamehkhosh, and Geoffray Bonnin. “Analyzing the Characteristics of Shared Playlists for Music Recommendation”. In: *Proceedings of the 6th Workshop on Recommender Systems and the Social Web at ACM RecSys*. 2014 (cit. on pp. 8, 12, 28, 34, 54, 56, 75).

[Jan+15a] Dietmar Jannach, Lukas Lerche, and Iman Kamehkhosh. “Beyond "Hitting the Hits": Generating Coherent Music Playlist Continuations with the Right Tracks”. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. RecSys ’15. 2015, pp. 187–194 (cit. on pp. 11, 13, 17, 21, 28–30, 44, 48, 75).

[Jan+15b] Dietmar Jannach, Lukas Lerche, and Michael Jugovac. “Item Familiarity as a Possible Confounding Factor in User-Centric Recommender Systems Evaluation”. In: *i-com Journal of Interactive Media* 14.1 (2015), pp. 29–39 (cit. on p. 36).

[Jan+15c] Dietmar Jannach, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac. “What Recommenders Rrecommend: An Analysis of Recommendation Biases and Possible Countermeasures”. In: *User Modeling and User-Adapted Interaction* 25.5 (2015), pp. 427–491 (cit. on pp. 27, 47, 76).

[Jan+16] Dietmar Jannach, Iman Kamehkhosh, and Geoffray Bonnin. “Biases in Automated Music Playlist Generation: A Comparison of Next-Track Recommending Techniques”. In: *Proceedings of the 24th Conference on User Modeling, Adaptation and Personalization*. UMAP ’16. 2016, pp. 281–285 (cit. on pp. 12, 13, 18, 33, 42, 45, 46, 48, 54, 75).

[Jan+17a] Dietmar Jannach, Iman Kamehkhosh, and Lukas Lerche. “Leveraging Multi-dimensional User Models for Personalized Next-track Music Recommendation”. In: *Proceedings of the 32nd ACM SIGAPP Symposium on Applied Computing*. SAC ’17. 2017, pp. 1635–1642 (cit. on pp. 4, 6, 7, 11, 13, 25, 27, 33, 44, 48, 75).

[Jan+17b] Dietmar Jannach, Malte Ludewig, and Lukas Lerche. “Session-based Item Recommendation in E-Commerce: On Short-Term Intents, Reminders, Trends, and Discounts”. In: *User-Modeling and User-Adapted Interaction* 27.3–5 (2017), pp. 351–392 (cit. on pp. 4, 41).

[Jan+17c] Dietmar Jannach and Malte Ludewig. “When Recurrent Neural Networks Meet the Neighborhood for Session-Based Recommendation”. In: *Proceedings of the Eleventh ACM Conference on Recommender Systems*. RecSys ’17. 2017, pp. 306–310 (cit. on p. 23).

[Jaw+10] Gawesh Jawaheer, Martin Szomszor, and Patty Kostkova. “Comparison of Implicit and Explicit Feedback from an Online Music Recommendation Service”. In: *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems*. HetRec ’10. 2010, pp. 47–51 (cit. on p. 18).

[Jug+17] Michael Jugovac, Dietmar Jannach, and Lukas Lerche. “Efficient Optimization of Multiple Recommendation Quality Factors According to Individual User Tendencies”. In: *Expert Systems With Applications* 81 (2017), pp. 321–331 (cit. on p. 31).

[Jyl+12] Antti Jylhä, Stefania Serafin, and Cumhur Erkut. “Rhythmic Walking Interactions with Auditory Feedback: An Exploratory Study”. In: *Proceedings of the 7th Audio Mostly Conference: A Conference on Interaction with Sound*. AM ’12. 2012, pp. 68–75 (cit. on p. 6).

[Kam+12a] Mohsen Kamalzadeh, Dominikus Baur, and Torsten Möller. “A Survey on Music Listening and Management Behaviours”. In: *Conference of the International Society for Music Information Retrieval*. 2012, pp. 373–378 (cit. on pp. 28, 36, 43).

[Kam+12b] Marius Kaminskas and Francesco Ricci. “Contextual Music Information Retrieval and Recommendation: State of the Art and Challenges”. In: *Computer Science Review* 6.2-3 (2012), pp. 89–119 (cit. on p. 8).

[Kam+13] Marius Kaminskas, Francesco Ricci, and Markus Schedl. “Location-aware Music Recommendation Using Auto-tagging and Hybrid Matching”. In: *Proceedings of the 7th ACM Conference on Recommender Systems*. RecSys ’13. 2013, pp. 17–24 (cit. on p. 57).

[Kam+16] Iman Kamehkhosh, Dietmar Jannach, and Lukas Lerche. “Personalized Next-Track Music Recommendation with Multi-dimensional Long-Term Preference Signals”. In: *Proceedings of the Workshop on Multi-dimensional Information Fusion for User Modeling and Personalization at ACM UMAP*. 2016 (cit. on pp. 13, 76).

[Kam+17a] Iman Kamehkhosh, Dietmar Jannach, and Malte Ludewig. “A Comparison of Frequent Pattern Techniques and a Deep Learning Method for Session-Based Recommendation”. In: *Proceedings of the Workshop on Temporal Reasoning in Recommender Systems at ACM RecSys*. 2017, pp. 50–56 (cit. on pp. 11, 14, 21, 22, 24, 44, 76).

[Kam+17b] Iman Kamehkhosh and Dietmar Jannach. “User Perception of Next-Track Music Recommendations”. In: *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. UMAP ’17. 2017, pp. 113–121 (cit. on pp. 12, 14, 36, 45, 48, 50, 51, 53, 75).

[Kap+15] Komal Kapoor, Vikas Kumar, Loren Terveen, Joseph A. Konstan, and Paul Schrater. “"I Like to Explore Sometimes": Adapting to Dynamic User Novelty Preferences”. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. RecSys ’15. 2015, pp. 19–26 (cit. on pp. 26, 28).

[Kne+06] Peter Knees, Tim Pohle, Markus Schedl, and Gerhard Widmer. “Combining Audio-based Similarity with Web-based Data to Accelerate Automatic Music Playlist Generation”. In: *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*. MIR ’06. 2006, pp. 147–154 (cit. on p. 43).

[Kne+08] Peter Knees, Markus Schedl, and Tim Pohle. “A Deeper Look into Web-Based Classification of Music Artists”. In: *Proceedings of the 2nd Workshop on Learning the Semantics of Audio Signals*. 2008, pp. 31–44 (cit. on p. 17).

[Kne+13] Peter Knees and Markus Schedl. “A Survey of Music Similarity and Recommendation from Music Context Data”. In: *ACM Transactions on Multimedia Computing Communications, and Applications* 10.1 (Dec. 2013), 2:1–2:21 (cit. on pp. 5, 16, 18).

[Kni+12] Bart P. Knijnenburg, Martijn C. Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. “Explaining the User Experience of Recommender Systems”. In: *User Modeling and User-Adapted Interaction* 22.4-5 (Oct. 2012), pp. 441–504 (cit. on p. 47).

[Kor+09] Yehuda Koren, Robert Bell, and Chris Volinsky. “Matrix Factorization Techniques for Recommender Systems”. In: *Computer* 42.8 (Aug. 2009), pp. 30–37 (cit. on p. 18).

[Köc+16] Sören Köcher, Dietmar Jannach, Michael Jugovac, and Hartmut H. Holzmüller. “Investigating Mere-Presence Effects of Recommendations on the Consumer Choice Process”. In: *Proceedings of the Joint Workshop on Interfaces and Human Decision Making for Recommender Systems at RecSys*. 2016 (cit. on p. 42).

[Lam+10] Alexandra Lamont and Rebecca Webb. “Short- and Long-Term Musical Preferences: What Makes a Favourite Piece of Music?” In: *Psychology of Music* 38.2 (2010), pp. 222–241 (cit. on p. 36).

[LeC+98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. “Gradient-Based Learning Applied to Document Recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324 (cit. on p. 16).

[Lee+04] Jin Ha Lee and J. Stephen Downie. “Survey Of Music Information Needs, Uses, And Seeking Behaviours: Preliminary Findings”. In: *5th International Conference on Music Information Retrieval*. 2004 (cit. on p. 36).

[Lee+11] Jin Ha Lee, Bobby Bare, and Gary Meek. “How Similar Is Too Similar?: Exploring Users’ Perceptions of Similarity in Playlist Evaluation”. In: *Conference of the International Society for Music Information Retrieval*. 2011, pp. 109–114 (cit. on p. 43).

[Lee+16] Jin Ha Lee and Rachel Price. “User Experience with Commercial Music Services: An Empirical Exploration”. In: *Journal of the Association for Information Science and Technology* 67.4 (2016), pp. 800–811 (cit. on p. 47).

[Lev+07] Mark Levy and Mark Sandler. “A Semantic Space for Music Derived from Social Tags”. In: *8th International Conference on Music Information Retrieval*. 2007 (cit. on p. 16).

[Lin+03] Greg Linden, Brent Smith, and Jeremy York. “Amazon.Com Recommendations: Item-to-Item Collaborative Filtering”. In: *IEEE Internet Computing* 7.1 (Jan. 2003), pp. 76–80 (cit. on p. 4).

[Lip15] Zachary Chase Lipton. “A Critical Review of Recurrent Neural Networks for Sequence Learning”. In: *CoRR* abs/1506.00019 (2015). arXiv: 1506.00019 (cit. on p. 20).

[Log+04] Beth Logan, Andrew Kositsky, and Pedro Moreno. “Semantic Analysis of Song Lyrics”. In: *IEEE International Conference on Multimedia and Expo*. Vol. 2. 2004, 827–830 Vol.2 (cit. on p. 16).

[Log02] Beth Logan. “Content-Based Playlist Generation: Exploratory Experiments.” In: *Conference of the International Society for Music Information Retrieval*. 2002 (cit. on p. 43).

[Log04] Beth Logan. “Music Recommendation from Song Sets”. In: *Conference of the International Society for Music Information Retrieval*. 2004, pp. 425–428 (cit. on pp. 8, 15, 43).

[Lon+16] Babak Loni, Roberto Pagano, Martha Larson, and Alan Hanjalic. “Bayesian Personalized Ranking with Multi-Channel User Feedback”. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys ’16. 2016, pp. 361–364 (cit. on p. 5).

[Mai+09] François Maillet, Douglas Eck, Guillaume Desjardins, and Paul Lamere. “Steerable Playlist Generation by Learning Song Similarity from Radio Station Playlists”. In: *International Society for Music Information Retrieval Conference*. 2009, pp. 345–350 (cit. on p. 43).

[McF+11] Brian McFee and Gert RG Lanckriet. “The Natural Language of Playlists”. In: *Conference of the International Society for Music Information Retrieval*. Vol. 11. 2011, pp. 537–542 (cit. on pp. 20, 43–45).

[McF+12a] Brian McFee and Gert R. G. Lanckriet. “Hypergraph Models of Playlist Dialects”. In: *Proceedings of the 13th International Society for Music Information Retrieval Conference*. 2012, pp. 343–348 (cit. on p. 23).

[McF+12b] Brian McFee, Thierry Bertin-Mahieux, Daniel P.W. Ellis, and Gert R.G. Lanckriet. “The Million Song Dataset Challenge”. In: *Proceedings of the 21st International Conference on World Wide Web*. WWW ’12 Companion. 2012, pp. 909–916 (cit. on p. 16).

[Moe+10] Bart Moens, Leon van Noorden, and Marc Leman. “D-Jogger: Syncing Music with Walking”. eng. In: *Proceedings of Sound and Music Computing*. Vol. online. 2010, pp. 451–456 (cit. on p. 6).

[Mol+12] Omar Moling, Linas Baltrunas, and Francesco Ricci. “Optimal Radio Channel Recommendations with Explicit and Implicit Feedback”. In: *Proceedings of the Sixth ACM Conference on Recommender Systems*. RecSys ’12. 2012, pp. 75–82 (cit. on p. 5).

[Moo+12] Jashua L. Moore, Shuo Chen, Thorsten Joachims, and Douglas Turnbull. “Learning to Embed Songs and Tags for Playlist Prediction”. In: *Conference of the International Society for Music Information Retrieval*. Vol. 12. 2012, pp. 349–354 (cit. on pp. 8, 15, 45).

[Mül15] Meinard Müller. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer International Publishing, 2015 (cit. on pp. 5, 16).

[Oh+11] Jinoh Oh, Sun Park, Hwanjo Yu, Min Song, and Seung-Taek Park. “Novel Recommendation Based on Personal Popularity Tendency”. In: *Proceedings of the 2011 IEEE 11th International Conference on Data Mining*. ICDM ’11. 2011, pp. 507–516 (cit. on p. 28).

[Oor+13] Aäron van den Oord, Sander Dieleman, and Benjamin Schrauwen. “Deep Content-Based Music Recommendation”. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2013, pp. 2643–2651 (cit. on p. 16).

[Pac+01] François Pachet, Gert Westermann, and Damien Laigre. “Musical Data Mining for Electronic Music Distribution”. In: *Proceedings of the First International Conference on WEB Delivering of Music*. WEDELMUSIC ’01. 2001, p. 101 (cit. on p. 19).

[Pan+08] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, et al. “One-Class Collaborative Filtering”. In: *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*. ICDM ’08. 2008, pp. 502–511 (cit. on p. 18).

[Par+11] Sung Eun Park, Sangkeun Lee, and Sang-goo Lee. “Session-Based Collaborative Filtering for Predicting the Next Song”. In: *Proceedings of the 2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering*. CNSI ’11. 2011, pp. 353–358 (cit. on p. 20).

[Pic+15] M. Pichl, E. Zangerle, and G. Specht. “Towards a Context-Aware Music Recommendation Approach: What is Hidden in the Playlist Name?” In: *2015 IEEE International Conference on Data Mining Workshop*. 2015, pp. 1360–1365 (cit. on pp. 27, 44).

[Pla+01] John C. Platt, Christopher J. C. Burges, Steven Swenson, Christopher Weare, and Alice Zheng. “Learning a Gaussian Process Prior for Automatically Generating Music Playlists”. In: *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*. NIPS’01. 2001, pp. 1425–1432 (cit. on p. 43).

[Poh+05] Tim Pohle, Elias Pampalk, and Gerhard Widmer. “Generating Similarity-Based Playlists using Traveling Salesman Algorithms”. In: *Proceedings of the 8th International Conference on Digital Audio Effects*. 2005, pp. 220–225 (cit. on p. 15).

[Poh+07] Tim Pohle, Peter Knees, Markus Schedl, and Gerhard Widmer. “Building an Interactive Next-Generation Artist Recommender Based on Automatically Derived High-Level Concepts”. In: *International Workshop on Content-Based Multimedia Indexing*. 2007, pp. 336–343 (cit. on p. 16).

[Pu+11] Pearl Pu, Li Chen, and Rong Hu. “A User-centric Evaluation Framework for Recommender Systems”. In: *Proceedings of the Fifth ACM Conference on Recommender Systems*. RecSys ’11. 2011, pp. 157–164 (cit. on p. 47).

[Pál+14] Róbert Pálovics, András A. Benczúr, Levente Kocsis, Tamás Kiss, and Erzsébet Frigó. “Exploiting Temporal Influence in Online Recommendation”. In: *Proceedings of the 8th ACM Conference on Recommender Systems*. RecSys ’14. 2014, pp. 273–280 (cit. on p. 5).

[Rib+14] Marco Tulio Ribeiro, Nivio Ziviani, Edleno Silva De Moura, Itamar Hata, Anisio Lacerda, and Adriano Veloso. “Multiobjective Pareto-Efficient Approaches for Recommender Systems”. In: *ACM Transactions on Intelligent Systems Technology* 5.4 (Dec. 2014), 53:1–53:20 (cit. on p. 28).

[Sar+12] Andy M Sarroff, Andy M Casey MichaelSarroff, and Michael Casey. “Modeling and Predicting Song Adjacencies in Commercial Albums”. In: *Proceedings of Sound and Music Computing*. 2012 (cit. on p. 34).

[Sch+11] Jan Schlüter and Christian Osendorfer. “Music Similarity Estimation with the Mean-Covariance Restricted Boltzmann Machine”. In: *10th International Conference on Machine Learning and Applications and Workshops*. Vol. 2. 2011, pp. 118–123 (cit. on p. 16).

[Sch+17a] Thomas Schäfer and Claudia Mehlhorn. “Can personality traits predict musical style preferences? A meta-analysis”. In: *Personality and Individual Differences* 116.Supplement C (2017), pp. 265 –273 (cit. on p. 57).

[Sch+17b] Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, and Mehdi Elahi. “Current Challenges and Visions in Music Recommender Systems Research”. In: *CoRR* abs/1710.03208 (2017). arXiv: 1710.03208 (cit. on pp. 7, 44, 45, 57).

[Sch+17c] Markus Schedl, Peter Knees, and Fabien Gouyon. “New Paths in Music Recommender Systems Research”. In: *Proceedings of the Eleventh ACM Conference on Recommender Systems*. RecSys ’17. 2017, pp. 392–393 (cit. on pp. 3, 6, 7).

[Sch17] Markus Schedl. “Investigating Country-Specific Music Preferences and Music Recommendation Algorithms with the LFM-1b Dataset”. In: *International Journal of Multimedia Information Retrieval* 6.1 (2017), pp. 71–84 (cit. on p. 57).

[Sha+09] Yuval Shavitt and Udi Weinsberg. “Song Clustering Using Peer-to-Peer Co-occurrences”. In: *11th IEEE International Symposium on Multimedia*. 2009, pp. 471–476 (cit. on p. 18).

[Sha+95] Upendra Shardanand and Pattie Maes. “Social Information Filtering: Algorithms for Automating Word of Mouth”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’95. 1995, pp. 210–217 (cit. on p. 2).

[Shi+12] Yue Shi, Xiaoxue Zhao, Jun Wang, Martha Larson, and Alan Hanjalic. “Adaptive Diversification of Recommendation Results via Latent Factor Portfolio”. In: *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’12. 2012, pp. 175–184 (cit. on p. 28).

[Sla+06] Malcolm Slaney and William White. “Measuring Playlist Diversity for Recommendation Systems”. In: *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*. AMCMM ’06. 2006, pp. 77–82 (cit. on pp. 28, 34, 43).

[Sla11] Malcolm Slaney. “Web-Scale Multimedia Analysis: Does Content Matter?” In: *IEEE MultiMedia* 18.2 (2011), pp. 12–15 (cit. on p. 15).

[Stu+11] Simone Stumpf and Sam Muscroft. “When Users Generate Music Playlists: When Words Leave Off, Music Begins?” In: *2011 IEEE International Conference on Multimedia and Expo*. 2011, pp. 1–6 (cit. on p. 36).

[Swe+02] Kirsten Swearingen and Rashmi Sinha. “Interaction Design for Recommender Systems”. In: *Designing Interactive Systems* 6.12 (2002), pp. 312–334 (cit. on p. 36).

[Tan+16] Yong Kiam Tan, Xinxing Xu, and Yong Liu. “Improved Recurrent Neural Networks for Session-based Recommendations”. In: *CoRR* abs/1606.08117 (2016) (cit. on p. 20).

[Tin+17] Nava Tintarev, Christoph Lofi, and Cynthia C.S. Liem. “Sequences of Diverse Song Recommendations: An Exploratory Study in a Commercial System”. In: *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. UMAP ’17. 2017, pp. 391–392 (cit. on pp. 7, 36).

[Tur+15] Roberto Turrin, Massimo Quadrana, Andrea Condorelli, Roberto Pagano, and Paolo Cremonesi. “30Music Listening and Playlists Dataset”. In: *Poster Proceedings RecSys ’15*. 2015 (cit. on pp. 22, 44).

[Tza+02] George Tzanetakis and Perry Cook. “Musical Genre Classification of Audio Signals”. In: *IEEE Transactions on Speech and Audio Processing* 10.5 (2002), pp. 293–302 (cit. on p. 16).

[Tza02] George Tzanetakis. “Manipulation, Analysis and Retrieval Systems for Audio Signals”. PhD thesis. Princetion University, 2002 (cit. on p. 16).

[Vas+16] Flavian Vasile, Elena Smirnova, and Alexis Conneau. “Meta-Prod2Vec: Product Embeddings Using Side-Information for Recommendation”. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys ’16. 2016, pp. 225–232 (cit. on pp. 5, 8, 18).

[VG+05] Rob Van Gulik and Fabio Vignoli. “Visual Playlist Generation on the Artist Map.” In: *Conference of the International Society for Music Information Retrieval (ISMIR)*. Vol. 5. 2005, pp. 520–523 (cit. on p. 16).

[Vig+05] Fabio Vignoli and Steffen Pauws. “A Music Retrieval System Based on User Driven Similarity and Its Evaluation.” In: *Conference of the International Society for Music Information Retrieval*. 2005, pp. 272–279 (cit. on p. 15).

[Voz+03] Emmanouil Vozalis and Konstantinos G Margaritis. “Analysis of Recommender Systems Algorithms”. In: *The 6th Hellenic European Conference on Computer Mathematics & its Applications*. 2003, pp. 732–745 (cit. on p. 3).

[Wan+12] Xinxi Wang, David Rosenblum, and Ye Wang. “Context-aware Mobile Music Recommendation for Daily Activities”. In: *Proceedings of the 20th ACM International Conference on Multimedia*. MM ’12. 2012, pp. 99–108 (cit. on p. 57).

[Wan+14] Xinxi Wang and Ye Wang. “Improving Content-based and Hybrid Music Recommendation Using Deep Learning”. In: *Proceedings of the 22Nd ACM International Conference on Multimedia*. MM ’14. 2014, pp. 627–636 (cit. on p. 16).

[Wei+16] Claus Weihs, Dietmar Jannach, Igor Vatolkin, and Guenter Rudolph, eds. *Music Data Analysis: Foundations and Applications*. CRC Press, 2016 (cit. on p. 16).

[Whi+02] Brian Whitman and Steve Lawrence. “Inferring Descriptions and Similarity for Music from Community Metadata.” In: *Proceedings of the 2002 International Computer Music Conference*. 2002 (cit. on p. 17).

[Wu+13] Xiang Wu, Qi Liu, Enhong Chen, Liang He, Jingsong Lv, Can Cao, et al. “Personalized Next-song Recommendation in Online Karaokes”. In: *Proceedings of the 7th ACM Conference on Recommender Systems*. RecSys ’13. 2013, pp. 137–140 (cit. on pp. 8, 24).

[Wur90] Richard Saul Wurman. *Information Anxiety: What to Do when Information Doesn’t Tell You What You Need to Know*. New York, NY: Bantam, 1990 (cit. on p. 2).

[Zan+12] Eva Zangerle, Wolfgang Gassler, and Günther Specht. “Exploiting Twitter’s Collective Knowledge for Music Recommendations.” In: *Proceedings of the 21st International World Wide Web Conference: Making Sense of Microposts*. 2012, pp. 14–17 (cit. on pp. 18, 22).

[Zha+08] Mi Zhang and Neil Hurley. “Avoiding Monotony: Improving the Diversity of Recommendation Lists”. In: *Proceedings of the 2008 ACM Conference on Recommender Systems*. RecSys ’08. 2008, pp. 123–130 (cit. on p. 28).

[Zhe+10] Elena Zheleva, John Guiver, Eduarda Mendes Rodrigues, and Nataša Milić-Frayling. “Statistical Models of Music-Listening Sessions in Social Media”. In: *Proceedings of the 19th International Conference on World Wide Web*. 2010, pp. 1019–1028 (cit. on p. 17).

[Zie+05] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. “Improving Recommendation Lists Through Topic Diversification”. In: *Proceedings of the 14th International Conference on World Wide Web*. WWW ’05. 2005, pp. 22–32 (cit. on pp. 28, 44).

[Özg+14] Özlem Özgöbek, Jon Atle Gulla, and Riza Cenk Erdur. “A Survey on Challenges and Methods in News Recommendation”. In: *Proceedings of the 10th International Conference on Web Information Systems and Technologies*. 2014, pp. 278–285 (cit. on p. 21).

Web pages

[Ber14] Erik Bernhardsson. *Recurrent Neural Networks for Collaborative Filtering*. 2014. URL: https://erikbern.com/2014/06/28/recurrent-neural-networks-for-collaborative-filtering.html (cit. on p. 20).

[Fri16] Joshua P. Friedlander. *News and Notes on 2017 Mid-Year RIAA Revenue Statistics*. 2016. URL: https://www.riaa.com/wp-content/uploads/2017/09/RIAA-Mid-Year-2017-News-and-Notes2.pdf (cit. on p. 1).

[Goo13] Howard Goodall. *BBC Howard Goodall's Story of Music – Part1*. 2013. URL: https://www.youtube.com/watch?v=I0Y6NPahlDE (cit. on p. 1).

[Hog15] Marc Hogan. *Up Next: How Playlists Are Curating the Future of Music*. 2015. URL: https://pitchfork.com/features/article/9686-up-next-how-playlists-are-curating-the-future-of-music/ (cit. on p. 2).

[Joh+15] Chris Johnson and Edward Newett. *From Idea to Execution: Spotify's Discover Weekly*. 2015. URL: https://de.slideshare.net/MrChrisJohnson/from-idea-to-execution-spotifys-discover-weekly/ (cit. on pp. 5, 17, 18).

[Joh14] Chris Johnson. *Algorithmic Music Discovery at Spotify*. 2014. URL: https://de.slideshare.net/MrChrisJohnson/algorithmic-music-recommendations-at-spotify/ (cit. on pp. 17, 18).

[Pre17] Spotify Press. *About Spotify*. 2017. URL: https://press.spotify.com/us/about/ (cit. on p. 2).

List of Figures

Publications

In this thesis by publication the following six works of the author are included. These publications are related to next-track music recommendation. The full texts of these works can be found after this list.

- Dietmar Jannach, Iman Kamehkhosh, and Geoffray Bonnin. “Analyzing the Characteristics of Shared Playlists for Music Recommendation”. In: *Proceedings of the 6th Workshop on Recommender Systems and the Social Web at ACM RecSys*. 2014

- Dietmar Jannach, Lukas Lerche, and Iman Kamehkhosh. “Beyond "Hitting the Hits": Generating Coherent Music Playlist Continuations with the Right Tracks”. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. RecSys ’15. 2015, pp. 187–194

- Dietmar Jannach, Iman Kamehkhosh, and Geoffray Bonnin. “Biases in Automated Music Playlist Generation: A Comparison of Next-Track Recommending Techniques”. In: *Proceedings of the 24th Conference on User Modeling, Adaptation and Personalization*. UMAP ’16. 2016, pp. 281–285

- Dietmar Jannach, Iman Kamehkhosh, and Lukas Lerche. “Leveraging Multi-dimensional User Models for Personalized Next-track Music Recommendation”. In: *Proceedings of the 32nd ACM SIGAPP Symposium on Applied Computing*. SAC ’17. 2017, pp. 1635–1642

- Iman Kamehkhosh and Dietmar Jannach. “User Perception of Next-Track Music Recommendations”. In: *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. UMAP ’17. 2017, pp. 113–121

- Iman Kamehkhosh, Dietmar Jannach, and Malte Ludewig. “A Comparison of Frequent Pattern Techniques and a Deep Learning Method for Session-Based Recommendation”. In: *Proceedings of the Workshop on Temporal Reasoning in Recommender Systems at ACM RecSys*. 2017, pp. 50–56

In addition to these six main publications, the author of this thesis worked on the following other publications related to recommender systems that are not part of this thesis.

- Dietmar Jannach, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac. “What Recommenders Rrecommend: An Analysis of Recommendation Biases and Possible Countermeasures”. In: *User Modeling and User-Adapted Interaction* 25.5 (2015), pp. 427–491

- Iman Kamehkhosh, Dietmar Jannach, and Lukas Lerche. “Personalized Next-Track Music Recommendation with Multi-dimensional Long-Term Preference Signals”. In: *Proceedings of the Workshop on Multi-dimensional Information Fusion for User Modeling and Personalization at ACM UMAP*. 2016

Analyzing the Characteristics of Shared Playlists for Music Recommendation

Dietmar Jannach
TU Dortmund, Germany
dietmar.jannach@tu-dortmund.de

Iman Kamehkhosh
TU Dortmund, Germany
iman.kamehkhosh@tu-dortmund.de

Geoffray Bonnin
TU Dortmund, Germany
geoffray.bonnin@tu-dortmund.de

ABSTRACT

The automated generation of music playlists – as supported by modern music services like last.fm or Spotify – represents a special form of music recommendation. When designing a "playlisting" algorithm, the question arises which kind of quality criteria the generated playlists should fulfill and if there are certain characteristics like homogeneity, diversity or freshness that make the playlists generally more enjoyable for the listeners. In our work, we aim to obtain a better understanding of such desired playlist characteristics in order to be able to design better algorithms in the future. The research approach chosen in this work is to analyze several thousand playlists that were created and shared by users on music platforms based on musical and meta-data features.

Our first results for example reveal that factors like popularity, freshness and diversity play a certain role for users when they create playlists manually. Comparing such user-generated playlists with automatically created ones moreover shows that today's online playlisting services sometimes generate playlists which are quite different from user-created ones. Finally, we compare the user-created playlists with playlists generated with a nearest-neighbor technique from the research literature and observe even stronger differences. This last observation can be seen as another indication that the accuracy-based quality measures from the literature are probably not sufficient to assess the effectiveness of playlisting algorithms.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.5.5 [**Information Interfaces and Presentation**]: Sound and Music Computing

General Terms

Playlist generation, Music recommendation

Keywords

Music, playlist, analysis, algorithm, evaluation

Proceedings of the 6th Workshop on Recommender Systems and the Social Web (RSWeb 2014), collocated with ACM RecSys 2014, 10/06/2014, Foster City, CA, USA.

.

1. INTRODUCTION

The automated creation of playlists or personalized radio stations is a typical feature of today's online music platforms and music streaming services. In principle, standard recommendation algorithms based on collaborative filtering or content-based techniques can be applied to generate a ranked list of musical tracks given some user preferences or past listening history. For several reasons, the generation of playlists however represents a very specific music recommendation problem. Personal playlists are, for example, often created with a certain goal or usage context (e.g., sports, relaxation, driving) in mind. Furthermore, in contrast to relevance-ranked recommendation lists used in other domains, playlists typically obey some homogeneity and coherence criteria, i.e., there are quality characteristics that are related to the transitions between the tracks or to the playlist as a whole.

In the research literature, a number of approaches for the automation of the playlist generation process have been proposed, see, e.g., [2, 6, 8, 10, 11] or the recent survey in [3]. Some of them for example take a seed song or artist as an input and look for similar tracks; others try to find track co-occurrence patterns in existing playlists. In some approaches, playlist generation is considered as an optimization problem. Independent of the chosen technique, a common problem when designing new playlisting algorithms is to assess whether or not the generated playlists will be positively perceived by the listeners. User studies and online experiments are unfortunately particularly costly in the music domain. Researchers therefore often use offline experimental designs and for example use existing playlists shared by users on music platforms as a basis for their evaluations. The assumption is that these "hand-crafted" playlists are of good quality; typical measures used in the literature include the *Recall* [8] or the *Average Log-Likelihood (ALL)* [11]. Unfortunately, both measures have their limitations, see also [2]. The *Recall* measure for example tells us how good an algorithm is at predicting the tracks selected by the users, but does not explicitly capture specific aspects such as the homogeneity or the smoothness of track transitions.

To design better and more comprehensive quality measures, we however first have to answer the question of what users consider to be desirable characteristics of playlists or what the driving principles are when users create playlists. In the literature, a few works have studied this aspect using different approaches, e.g., user studies [1, 7] or analyzing forum posts [5]. The work presented in this paper continues these lines of research. Our research approach is however

different from previous works as we aim to identify patterns in a larger set of manually created playlists that were shared by users of three different online music platforms. To be able to take a variety of potential driving factors into account in our analysis, we have furthermore collected various types of meta-data and musical features of the playlist tracks from public music databases.

Overall, with our analyses we hope to obtain insights on the principles which an automated playlist generation system should observe to end up with better-received or more "natural" playlists. To test if current music services and a nearest-neighbor algorithm from the literature generate playlists that observe the identified patterns and make similar choices as real users, we conducted an experiment in which we analyzed commonalities and differences between automatically generated and user-provided playlists.

Before reporting the details of our first analyses, we will first discuss previous works in the next section.

2. PREVIOUS WORKS

In [14], Slaney and White addressed the question if users have a tendency to create very homogeneous or rather diverse playlists. As a basis for determining the diversity they relied on an objective measure based on genre information about the tracks. Each track was considered as a point in the genre space and the diversity was then determined by calculating the volume of an ellipsoid enclosing the tracks of the playlist. An analysis of 887 user-created playlists indicated that diversity can be considered to be a driving factor as users typically create playlists covering several genres.

Sarroff and Casey more recently [13] focused on track transitions in album playlists and made an analysis to determine if there are certain musical characteristics that are particularly important. One of the results of their investigation was that fade durations and the mean timbre of the beginnings and endings of consecutive tracks seem to have a strong influence on the ordering of the tracks.

Generally, our work is similar to [14] and [13] in that we rely on user-created ("hand-crafted") playlists and look at meta-data and musical features of the tracks to identify potentially important patterns. The aspects we cover in this paper were however not covered in their work and our analysis is based on larger datasets.

Cunningham et al., [5], in contrast, relied on another form of track-related information and looked at the user posts in the forum of the Art of the Mix web site. According to their analysis, the typical principles for setting up the playlists mentioned by the creators were related to the artist, genre, style, event or activity but also the intended purpose, context or mood. Some users also talked about the smoothness of track transitions and how many tracks of one single artist should be included in playlists. Placing the most "important" track at the end of a playlist was another strategy mentioned by some of the playlist creators.

A different form of identifying playlist creation principles is to conduct laboratory studies with users. The study reported in [7] for example involved 52 subjects and indicated that the first and the last tracks can play an important role for the quality of a playlist. In another study, Andric and Haus [1] concluded that the ordering of tracks is not important when the playlist mainly contains tracks which the users like in general.

Reynolds et al. [12] made an online survey that revealed that the context and environment like the location activity or the weather can have an influence both on the listeners' mood and on the track selection behavior of playlist creators. Finally, the study presented in [9] again confirmed the importance of artists, genres and mood in the playlist creation process.

In this discussion, we have focused on previous attempts to understand how users create playlists and what their characteristics are. Playlist generation algorithms however do not necessarily have to rely on such knowledge. Instead, one can follow a statistical approach and only look at co-occurrences and transitions of tracks in existing playlists and use these patterns when creating new playlists, see e.g., [2] or [4]. This way, the quality factors respected by human playlist creators are implicitly taken into account. Such approaches, however, cannot be directly applied for many types of playlist generation settings, e.g., for creating "thematic" playlists (e.g., Christmas Songs) or for creating playlists that only contain tracks that have certain musical features. Pure statistical methods are not aware of these characteristics and the danger exists that tracks are included that do not match the purpose of the list and thus lead to a limited overall quality.

3. CHARACTERISTICS OF PLAYLISTS

The ultimate goal of our research is to analyze the structure and characteristics of playlists in order to better understand the principles used by the users to create them. This section is a first step toward this goal.

3.1 Data sources

As a basis for the first analyses that we report in this paper, we used two types of playlist data.

3.1.1 Hand-crafted playlists

We used samples of hand-crafted playlists from three different sources. One set of playlists was retrieved via the public API of last.fm[1], one was taken from the Art of the Mix (AotM) website[2], and a third one was provided to us by 8tracks[3]. To enhance the data quality, we corrected artist misspellings using the API of last.fm.

Overall, we analyzed over 10,000 playlists containing about 108,000 different tracks of about 40,000 different artists. As a first attempt toward our goal, we retrieved the features listed in Table 1 using the public API of last.fm and The Echo Nest (tEN), and the MusicBrainz database.

Some dataset characteristics are shown in Table 2. The "usage count" statistics express how often tracks and artists appeared overall in the playlists. When selecting the playlists, we made sure that they do not simply contain album listings. The datasets are partially quite different, e.g., with respect to the average playlist lengths. The 8tracks dataset furthermore has the particularity that users are not allowed to include more than two tracks of one artist, in case they want to share their playlist with others.

Figure 1 shows the distributions of playlist lengths. As can be seen, the distributions are quite different across the datasets. On 8tracks, a playlist generally has to comprise

[1] `http://www.last.fm`
[2] `http://www.artofthemix.org`
[3] `http://8tracks.com`

Source	Information	Description
last.fm	Tags	Top tags assigned by users to the track.
last.fm	Playcounts	Total number of times the users played the track.
tEN	Genres	Genres of the artist of the track. Multiple genres can be assigned to a single artist.
tEN	Danceability	Suitability of the track for dancing, based on various information including the beat strength and the stability of the tempo.
tEN	Energy	Intensity released throughout the track, based on various information including the loudness and segment durations.
tEN	Loudness	Overall loudness of the track in decibels (dB).
tEN	Tempo	Speed of the track estimated in beats per minute (BPM).
tEN	Hotttnesss	Current reputation of the track based on its activity on some web sites crawled by the developers.
MB	Release year	Year of release of the corresponding album.

Table 1: Additional retrieved information.

	lastfm	AotM	8tracks
Playlists	1,172	5,043	3,130
Tracks	24,754	61,935	29,732
Artists	9,925	23,029	13,379
Avg. tracks/playlist	26.0	19.7	12.5
Avg. artists/playlist	16.8	17.8	11.5
Avg. genres/playlist	2.7	3.5	3.4
Avg. tags/playlist	473.4	418.7	297.4
Avg. track usage count	1.2	1.6	1.3
Avg. artist usage count	3.0	4.3	2.9

Table 2: Some basic statistics of the datasets.

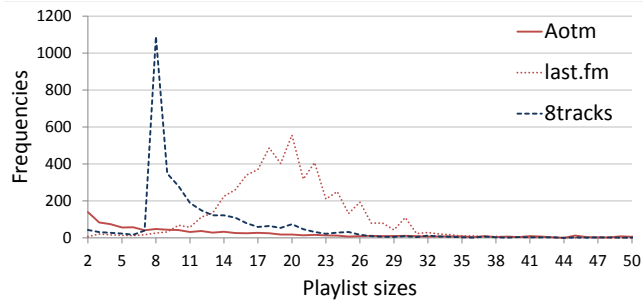


Figure 1: Distribution of playlists sizes.

at least 8 tracks. The lengths of the last.fm playlists seem to follow a normal distribution with a maximum frequency value at around 20 tracks. Finally, the sizes of the AotM playlists are much more equally distributed.

3.1.2 *Generated playlists*

To assess if the playlists generated by today's online services are similar to those created by users, we used the public API of The Echo Nest. We chose this service because it uses a very large database and allows the generation of playlists from several seed tracks, as opposed to, for instance, iTunes Genius or last.fm radios. We split the existing hand-crafted playlists in half, provided the first half of the list as seed tracks to the music service and then analyzed the characteristics of the playlist returned by The Echo Nest and compared them to the patterns that we found in hand-crafted playlists. Instead of observing whether a playlister generates playlists that are generally similar to playlists created by hand, our goal here is to break down their different characteristics and observe on what specific dimensions they differ. Notice that using the second half as seed would not be appropriate as the order of the tracks may be important.

We also draw our attention to the ability of the algorithms of the literature to reproduce the characteristics of hand-crafted playlists. According to some recent research, one of the most competitive approaches in terms of recall is the simple k-nearest-neighbors (kNN) method [2, 8]. More precisely, given some seed tracks, the algorithm extracts the k most similar playlists based on the number of shared items and recommends the tracks of these playlists. This algorithm does not require a training step and scans the entire set of available playlists for each recommendation.

3.2 Detailed observations

In the following sections, we will look at general distributions of different track characteristics.

3.2.1 *Popularity of tracks*

The goal of the first analysis here is to determine if users tend to position tracks in playlists depending on their popularity. In our analysis, we measure the popularity in terms of play counts. Play counts were taken from last.fm, because this is one of the most popular services and the corresponding values can be considered indicative for a larger user group.

For the measurement, we split the playlists into two parts of equal size and then determined the average play counts on last.fm for the tracks for each half. To measure to which extent the user community favors certain tracks in the playlists, we calculated the Gini index, a standard measure of inequality[4]. Table 3 shows the results. In the last column, we report the statistics for the tracks returned by The Echo Nest (tEN) and kNN playlisters[5]. We provided the first half of the hand-crafted playlists as seed tracks and the playlisters had to select the same number of tracks as the number of remaining tracks.

The results show that users actually tend to place more popular items in the first part of the list in all datasets, when play counts are considered. The Echo Nest playlister does not seem to take that form of popularity into account

[4]We organized the average play counts in 100 bins.
[5]We determined 10 as the best neighborhood size for our data sets based on the recall value, see Section 4.

Play counts	1st half	2nd half	tEN
last.fm	1,007k	893k	629k
AotM	671k	638k	606k
8tracks	953k	897k	659k

Gini index	1st half	2nd half	tEN
last.fm	0.06	0.04	0.04
AotM	0.20	0.18	0.22
8tracks	0.09	0.09	0.08

Play counts	1st half	2nd half	kNN
last.fm	1,110k	943k	1,499k
AotM	645k	617k	867k
8tracks	1,008k	984k	1,140k

Gini index	1st half	2nd half	kNN
last.fm	0.12	0.09	0.33
AotM	0.26	0.23	0.43
8tracks	0.15	0.12	0.28

Table 3: Popularity of tracks in playlists (last.fm play counts) and concentration bias (Gini coefficient).

and recommends on average less popular tracks. These differences are statistically significant according to a Student's t-test ($p < 10^{-5}$ for The Echo Nest playlister and $p < 10^{-7}$ for the kNN playlister). This behavior indicates also that The Echo Nest is successfully replicating the fact that the second halves of playlists are supposed to be less popular than the first half.

The Gini index reveals that there is a slightly stronger concentration on some tracks in the first half for two of three datasets and the diversity slightly increases in the second part. The absolute numbers cannot be directly compared across datasets, but for the AotM dataset the concentration is generally much higher, which is also indicated by the higher "track reuse" in Table 2. Interestingly, The Echo Nest playlister quite nicely reproduces the behavior of real users with respect to the diversity of popularity.

In the lower part of Table 3, we show the results for the kNN method. Note that these statistics are based on a different sample of the playlists than the previous measurement. The reason is that both The Echo Nest and the kNN playlisters cannot produce playlists for all of the first halves provided as seed tracks. We therefore considered only playlists, for which the corresponding algorithm could produce a playlist.

Unlike the playlister of The Echo Nest, the kNN method has a strong trend to recommend mostly very popular items. This can be caused by the fact that the kNN method by design recommends tracks that are often found in similar playlists. Moreover, based on the lower half of Table 3, the popularity correlates strongly with the seed track popularity. As a result, the kNN shows a potentially undesirable trend to reinforce already popular items to everyone. At the same time, it concentrates the track selection on a comparable small number of tracks as indicated by the very high value for the Gini coefficient.

3.2.2 *The role of freshness*

Next, we analyzed if there is a tendency of users to create playlists that mainly contain recently released tracks. As a measure, we compared the creation year of each playlist with the average release year of its tracks. We limit our analysis to the last.fm and 8tracks datasets because we only could acquire creation dates for these two.

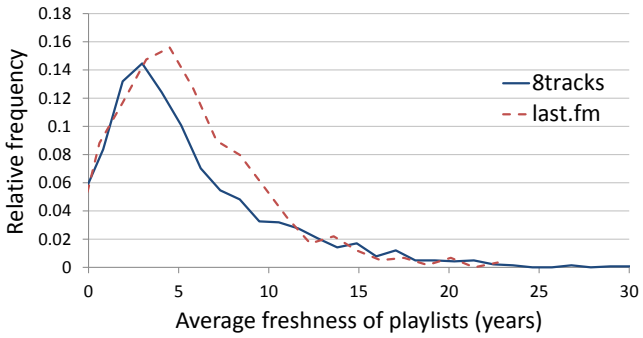


Figure 2: Distribution of average freshness of playlists (comparing playlist creation date and track release date).

Figure 2 shows the statistics for both datasets. We organized the data points in bins (x-axis), where each bin represents an average-freshness level, and then counted how many playlists fall into these levels. The relative frequencies are shown on the y-axis. The result are very similar for both datasets, with a slight tendency to include older tracks for last.fm. On both datasets, more than half of the playlists contain tracks that were released on average in the last 5 years, the most frequent average age being between 4 and 5 years for last.fm and between 3 and 4 years for 8tracks. Similarly, on both datasets, more than 75% of the playlists contain tracks that were released on average in the last 8 years.

We also analyzed the standard deviation of the resulting freshness values and observed that more than half of the playlists have a standard deviation of less than 4 (years), while more than 75% have a standard deviation of less than 7 (years) on both datasets. Overall, this suggests that playlists made by users are often homogeneous with regard to the release date.

Computing the freshness for the generated playlists would require to configure the playlisters in such a way that they select only tracks that were not released after the playlists' creation years. Unfortunately, The Echo Nest does not allow such a configuration. Moreover, for the kNN approach, the playlists that are more recent would have to be ignored, which would lead to a too small sample size and not very reliable results anymore.

3.2.3 *Homogeneity and diversity*

Homogeneity and diversity can be determined in a variety of ways. In the following, we will use simple measures based on artist and genre counts. The genres correspond to the genres of the artists of the tracks retrieved from The Echo Nest. Basic figures for artist and genre diversity are already given in Table 2. On AotM, for example, having several tracks of an artist in a playlist is not very common[6]. On last.fm, we in contrast very often see two or more tracks of

[6]On 8tracks, artist repetitions are limited due to license constraints

one artist in a playlist. A similar, very rough estimate can be made for the genre diversity. If we ordered the tracks of a playlist by genre, we would encounter a different genre on last.fm only after having listened to about 10 tracks. On AotM and 8tracks, in contrast, playlists on average cover more genres.

Table 4 shows the diversities of the first and second halves of the hand-crafted playlists, and for the automatic selections using the first halves as seeds. As a measure of diversity, we simply counted the number of artists and genres and divided by the corresponding number of tracks. The values in Table 4 correspond the averages of these diversity measures.

		1st half	2nd half	tEN
last.fm	artists	0.74	0.76	0.93
	genres	2.26	2.30	2.12
AotM	artists	0.93	0.93	0.94
	genres	3.26	3.22	2.41
8tracks	artists	0.97	0.98	0.99
	genres	3.74	3.85	2.89

		1st half	2nd half	kNN
last.fm	artists	0.74	0.76	0.87
	genres	2.32	2.26	3.11
AotM	artists	0.94	0.94	0.91
	genres	3.27	3.21	3.70
8tracks	artists	0.97	0.98	0.93
	genres	3.94	3.92	4.06

Table 4: Diversity of playlists (Number of artists and genres divided by the corresponding number of tracks).

Regarding the diversity of the hand-crafted playlists, the tables show that users tend to keep a same level of artist and genre diversity throughout the playlists. We can also notice that the playlists of last.fm are much more homogeneous. The diversity values of the automatic selections reveal several things. First, The Echo Nest playlister tends to always maximize the artist diversity independently of the diversity of the seeds; on the contrary, the kNN playlister lowered the initial artist diversities, except on the last.fm dataset, where it increased them, though less than The Echo Nest playlister. Regarding the genre diversity, we can observe an opposite tendency for both playlisters: The Echo Nest playlister tends to reduce the genre diversity while the kNN playlister tends to increase it. Again, these difference are statistically significant ($p < 0.03$ for The Echo Nest playlister and $p < 0.006$ for the kNN playlister). Overall, the resulting diversities of the both approaches tend to be rather dissimilar to those of the hand-crafted playlists.

3.2.4 *Musical features (The Echo Nest)*

Figure 3 shows the overall relative frequency distribution of the numerical features from The Echo Nest listed in Table 1 for the set of tracks appearing in our playlists on a normalized scale. For the loudness feature, for example, we see that most tracks have values between 40 and 50 on the normalized scale. This would translate into an actual loudness value of -20 to 0 returned by The Echo Nest, given that the range is -100 to 100.

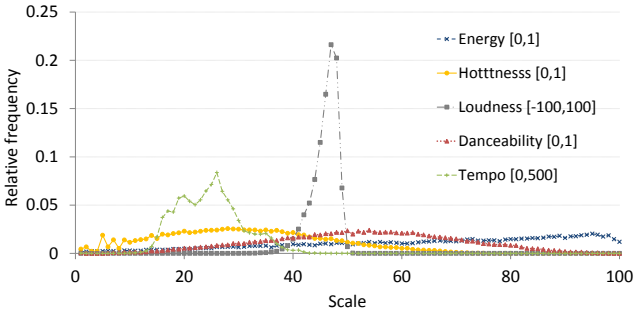


Figure 3: Distribution of The Echo Nest track musical features independently of playlists.

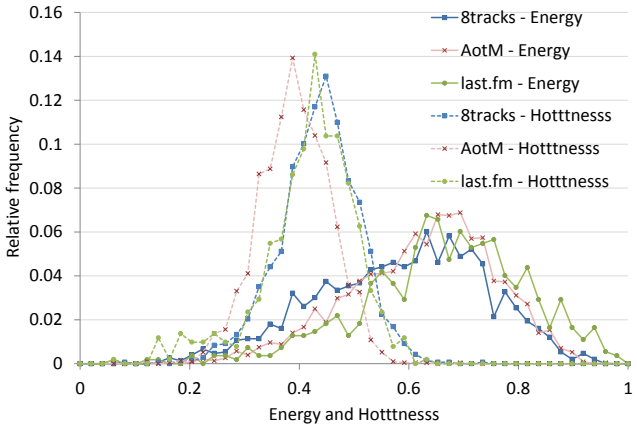


Figure 4: Distribution of mean energy and "hotttnesss" levels in playlists.

To understand if people tend to place tracks with specific feature values into their playlists, we then computed the distribution of the average feature values of each playlist. Figure 4 shows the results of this measurement for the energy and "hotttnesss" features. For all the other features (danceability, loudness and tempo), the distributions were similar to those of Figure 3, which could mean that they are generally not particularly important for the users.

When looking at the energy feature, we see that users tend to include tracks from a comparably narrow energy spectrum with a low average energy level, even though there exist more high-energy tracks in general as shown in Figure 3. A similar phenomenon of concentration on a certain range of values can be observed for the "hotttnesss" feature. As a side aspect, we can observe that the tracks shared on AotM are on average slightly less "hottt" than those of both other platforms[7].

We finally draw our attention to the feature distributions of the generated playlists. Figure 5 as an example shows the distributions of the energy and "hotttnesss" factors for

[7]The results for the "hotttnesss" we report here correspond to the values at the time when we retrieved the data using the API of The Echo Nest, and not to those at the time when the playlists were created. This is not important as we do not look at the distributions independently, but compare them to the distributions in Figure 3.

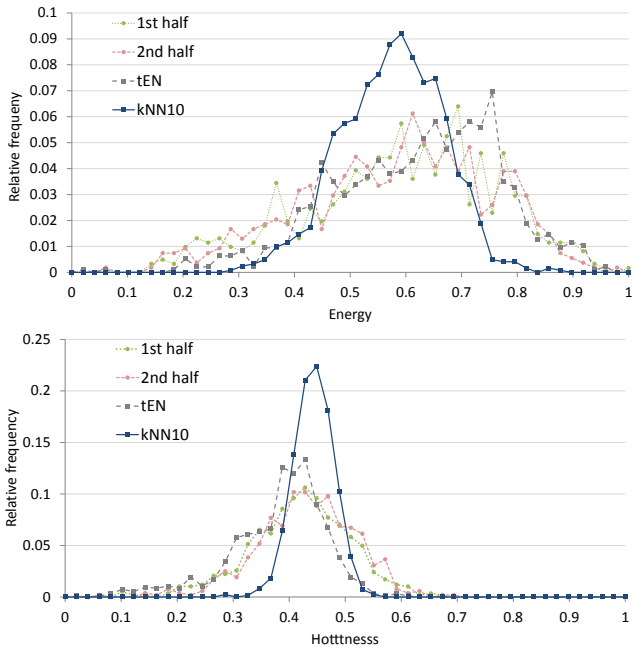


Figure 5: Comparison of the distribution of energy and "hotttnesss" levels for hand-crafted and generated playlists.

the first halves and second halves of the playlists of all three datasets, together with the distributions of the tracks selected by The Echo Nest and kNN playlisters.

The figure shows that The Echo Nest playlister tends to produce a distribution that is quite similar to the distribution of the seed tracks. The kNN playlister, in contrast, tends to concentrate the distributions toward the maximum values of the distributions of the seeds. We could observe this phenomenon of concentration for all the features on all three datasets, except for the danceability on the AotM dataset.

3.2.5 *Transitions and Coherence*

We now focus on the importance of transitions between the tracks, and define the *coherence* of a playlist as the average similarity between its consecutive tracks. Such similarities can be computed according to various criteria. We used the binary cosine similarity of the genres and artists[8], and the Euclidean linear similarity for the numerical track features of The Echo Nest. Table 5 shows the corresponding results for the first and second halves of the hand-crafted playlists, and for the automatic selections using the first halves as seeds.

We can first see that for all datasets and for all criteria, the second halves of the playlists have a lower coherence than the first halves. If we assume that the coherence is representative of the effort of the users to create good playlists, then the tracks of the second halves seem to be slightly less carefully selected than those of the first halves.

[8]In the case of artists, this means that the similarity equals 1 if both tracks have the same artist, and 0 else. The metric thus measures the proportion of cases when the users consecutively selected tracks from the same artist.

		1st half	2nd half	tEN
last.fm	artists	0.19	0.18	0
	genres	0.43	0.40	0.56
	energy	0.76	0.71	0.77
	hotttnesss	0.81	0.76	0.83
AotM	artists	0.05	0.05	0
	genres	0.24	0.22	0.50
	energy	0.75	0.74	0.75
	hotttnesss	0.83	0.82	0.85
8tracks	artists	0.02	0.01	0
	genres	0.22	0.22	0.52
	energy	0.73	0.71	0.76
	hotttnesss	0.81	0.79	0.85

		1st half	2nd half	kNN
last.fm	artists	0.22	0.21	0.02
	genres	0.44	0.42	0.14
	energy	0.76	0.76	0.75
	hotttnesss	0.83	0.82	0.83
AotM	artists	0.05	0.05	0.03
	genres	0.22	0.21	0.13
	energy	0.75	0.74	0.73
	hotttnesss	0.83	0.82	0.84
8tracks	artists	0.02	0.01	0.03
	genres	0.22	0.22	0.17
	energy	0.74	0.73	0.74
	hotttnesss	0.82	0.80	0.84

Table 5: Coherence of first, second and generated halves.

Another interesting phenomenon is the high artist coherence values on the last.fm dataset. These values indicate that last.fm users have a surprisingly strong tendency to group tracks from the same artist together, which was not successfully reproduced by the two playlisters. Both playlisters actually seem to have a tendency to produce always the same coherence values, independently of the coherence values of the seed. A last interesting result is the high coherence of artist genres on the AotM and 8tracks datasets – the high genre coherence values on last.fm can be explained by the high artist coherence values.

4. STANDARD ACCURACY METRICS

Our analysis so far has revealed some particular characteristics of user-created playlists. Furthermore, we observed that the nearest-neighbor playlisting scheme can produce playlists that are quite different to those generated by the commercial Echo Nest service, e.g., in terms of average track popularity (Table 3).

In the research literature, "hit rates" (recall) and the average log-likelihood ($\mathcal{ALL}$) are often used to compare the quality of playlists generated by different algorithms [2, 8, 11]. The goal of our next experiment was to find out how The Echo Nest playlister performs on these measures. As it is not possible to acquire probability values for the tracks selected by The Echo Nest playlister, the $\mathcal{ALL}$ cannot be

used[9]. In the following we thus only focus on the precision and recall.

The upper part of Figure 6 shows the recall values at list length 100 for the different datasets[10]. Again, we split the playlists and used the first half as seed tracks. Recall was then computed by comparing the computed playlists with the "hidden" tracks of the original playlist. We measured recall for tracks, artists, genres and tags. The results show that the kNN method quite clearly outperforms the playlister of The Echo Nest on the recall measures across all datasets except for the artist recall for the last.fm dataset. The differences are statistically significant for all the experiments except for the track and artists recall on last.fm ($p < 10^{-6}$) according to a Student's t-test. As expected, the kNN method leads to higher absolute values for larger datasets as more neighbors can be found.

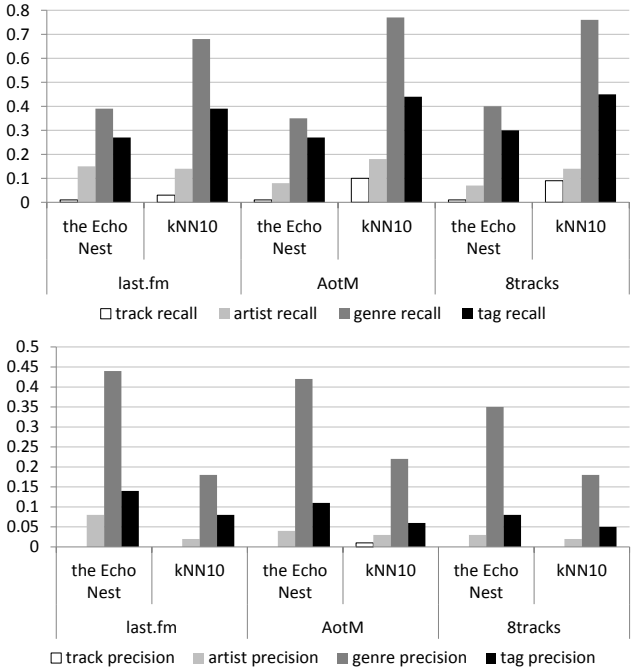


Figure 6: Recall and Precision for the covered cases.

The lower part of Figure 6 presents the precision results. The precision values for tracks are as expected very low and close to zero which is caused by the huge set of possible tracks and the list length of 100. We can however observe a higher precision for the kNN method on the AotM dataset ($p < 10^{-11}$), which is the largest dataset. Regarding artist, genre and tag prediction, The Echo Nest playlister lead to a higher precision ($p < 10^{-3}$) than the kNN playlister on all datasets.

[9]Another possible measure is the Mean Reciprocal Rank (MRR). Applied to playlist generation, one limitation of this metric is that it corresponds to the assumption that the rank of the test track or artist to predict should be as high as possible in the recommendation list, although many other tracks or artist may be more relevant and should be ranked before.

[10]We could not measure longer list lengths as 100 is the maximum playlist length returned by The Echo Nest.

With respect to the evaluation protocol, note that we only measured precision and recall when the playlister was able to return a playlist continuation given the seed tracks. This was however not always the case for both techniques. In Table 6, we therefore report the detailed coverage figures, which show that the kNN method was more often able to produce a playlist. If recall is measured for all seed playlists, the differences between the algorithms are even larger. When measuring precision for all playlists, the differences between the playlisters become very small.

Dataset	tEN	kNN
last.fm	28.33	66.89
AotM	42.75	86.52
8tracks	35.3	43.8

Table 6: Coverage of the playlisters.

Overall, measuring precision and recall when comparing generated playlists with those provided by users in our view represents only one particular form of assessing the quality of a playlist generator and should be complemented with additional measures. Precision and recall as measured in our experiments for example do not consider track transitions. There is also no "punishment" if a generated playlist contains individual non-fitting tracks that would hurt the listener's overall enjoyment.

5. PUBLIC AND PRIVATE PLAYLISTS

Some music platforms and in particular 8tracks let their users create "private" playlists which are not visible to others and public ones that for example are shared and used for social interaction like parties, motivation for team sport or romantic evening. The question arises if public playlists have different characteristics than those that were created for personal use only, e.g., because sharing playlists to some extent can also serve the purpose of creating a public image of oneself.

We made an initial analysis on the 8tracks dataset. Table 7 shows the average popularity of the tracks in the 8tracks playlists depending on whether they were in "public" or "private" playlists (the first category contains 2679 playlists and the second 451). As can be seen, the tracks of the private playlists are much more popular on average than the tracks in the public playlists. Moreover, as indicated by the corresponding Gini coefficients, the popular tracks are almost equally distributed across the playlists. Furthermore, Figure 7 shows the corresponding freshness values. We can see that the private playlists generally contained more recent tracks than public playlists.

	Play counts	Gini index
Public playlists	870k	0.20
Private playlists	935k	0.06

Table 7: Popularity of tracks in 8tracks public and private playlists and Gini index.

These results can be interpreted at least in two different ways. First, users might create some playlists for their personal use to be able to repeatedly listen to the latest popular tracks. They probably do not share these playlists because

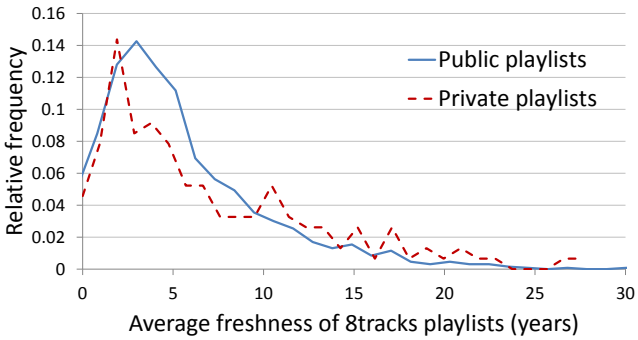


Figure 7: Distribution of average freshness of 8tracks public and private playlists.

sharing a list of current top hits might be of limited value for other platform members who might be generally more interested in *discovering* not so popular artists and tracks. Second, users might deliberately share playlists with less popular or known artists and tracks to create a social image on the platform.

Given these first observations, we believe that our approach has some potential to help us better understand some elements of user behavior on social platforms in general, i.e., that people might not necessarily only share tracks that match their actual taste.

6. SUMMARY AND OUTLOOK

The goal of our work is to gain a better understanding of how users create playlists in order to be able to design future playlisting algorithms that take these "natural" characteristics into account. The first results reported in this paper indicate, for example, that features like track freshness, popularity aspects, or homogeneity of the tracks are relevant for users, but not yet fully taken into account by current algorithms that are considered to create high-quality playlists in the literature. Overall, the observations also indicate that additional metrics might be required to assess the quality of computer-generated playlists in experimental settings that are based on historical data such as existing playlists or listening logs.

Given the richness of the available data, many more analyses are possible. Currently, we are exploring "semantic" characteristics to automatically identify the underlying theme or topic of the playlists. Another aspect not considered so far in our research is the popularity of the playlists. For some music platforms, listening counts and "like" statements for playlists are available. This additional information can be used to further differentiate between "good" and "bad" playlists and help us obtain more fine-granular differences with respect to the corresponding playlist characteristics. Last, we plan to extend our experiments and analysis by considering other music services, in particular last.fm radios, and other playlisting algorithms, in particular algorithms that exploit content information.

7. REFERENCES

[1] A. Andric and G. Haus. Estimating Quality of Playlists by Sight. In *Proc. AXMEDIS*, pages 68–74, 2005.

[2] G. Bonnin and D. Jannach. Evaluating the Quality of Playlists Based on Hand-Crafted Samples. In *Proc. ISMIR*, pages 263–268, 2013.

[3] G. Bonnin and D. Jannach. Automated generation of music playlists: Survey and experiments. *ACM Computing Surveys*, 47(2), 2014.

[4] S. Chen, J. L. Moore, D. Turnbull, and T. Joachims. Playlist Prediction via Metric Embedding. In *Proc. KDD*, pages 714–722, 2012.

[5] S. Cunningham, D. Bainbridge, and A. Falconer. 'More of an Art than a Science': Supporting the Creation of Playlists and Mixes. In *Proc. ISMIR*, pages 240–245, 2006.

[6] A. Flexer, D. Schnitzer, M. Gasser, and G. Widmer. Playlist Generation Using Start and End Songs. In *Proc. ISMIR*, pages 173–178, 2008.

[7] D. L. Hansen and J. Golbeck. Mixing It Up: Recommending Collections of Items. In *Proc. CHI*, pages 1217–1226, 2009.

[8] N. Hariri, B. Mobasher, and R. Burke. Context-Aware Music Recommendation Based on Latent Topic Sequential Patterns. In *Proc. RecSys*, pages 131–138, 2012.

[9] M. Kamalzadeh, D. Baur, and T. Möller. A Survey on Music Listening and Management Behaviours. In *Proc. ISMIR*, pages 373–378, 2012.

[10] A. Lehtiniemi and J. Seppänen. Evaluation of Automatic Mobile Playlist Generator. In *Proc. MC*, pages 452–459, 2007.

[11] B. McFee and G. R. Lanckriet. The Natural Language of Playlists. In *Proc. ISMIR*, pages 537–542, 2011.

[12] G. Reynolds, D. Barry, T. Burke, and E. Coyle. Interacting With Large Music Collections: Towards the Use of Environmental Metadata. In *Proc. ICME*, pages 989–992, 2008.

[13] A. M. Sarroff and M. Casey. Modeling and Predicting Song Adjacencies In Commercial Albums. In *Proc. SMC*, 2012.

[14] M. Slaney and W. White. Measuring Playlist Diversity for Recommendation Systems. In *Proc. AMCMM*, pages 77–82, 2006.

Beyond "Hitting the Hits" – Generating Coherent Music Playlist Continuations with the Right Tracks

[Placeholder]

Dietmar Jannach
TU Dortmund, Germany
dietmar.jannach@tu-dortmund.de

Lukas Lerche
TU Dortmund, Germany
lukas.lerche@tu-dortmund.de

Iman Kamehkhosh
TU Dortmund, Germany
iman.kamehkhosh@tu-dortmund.de

RecSys '15, September 16–20, 2015, Vienna, Austria
DOI: http://dx.doi.org/10.1145/2792838.2800182

Biases in Automated Music Playlist Generation: A Comparison of Next-Track Recommending Techniques

[Placeholder]

Dietmar Jannach
TU Dortmund, Germany
dietmar.jannach@
tu-dortmund.de

Iman Kamehkhosh
TU Dortmund, Germany
iman.kamehkhosh@
tu-dortmund.de

Geoffray Bonnin
LORIA, Nancy, France
geoffray.bonnin@
loria.fr

UMAP '16, July 13-17, 2016, Halifax, NS, Canada
DOI: http://dx.doi.org/10.1145/2930238.2930283

Leveraging Multi-Dimensional User Models for Personalized Next-Track Music Recommendation

[Placeholder]

Dietmar Jannach
TU Dortmund, Germany
dietmar.jannach@
tu-dortmund.de

Iman Kamehkhosh
TU Dortmund, Germany
iman.kamehkhosh@
tu-dortmund.de

Lukas Lerche
TU Dortmund, Germany
lukas.lerche@
tu-dortmund.de

SAC '17, April 03 - 07, 2017, Marrakech, Morocco
DOI: http://dx.doi.org/10.1145/3019612.3019756

User Perception of Next-Track Music Recommendations

[Placeholder]

Iman Kamehkhosh
TU Dortmund, Germany
iman.kamehkhosh@
tu-dortmund.de

Dietmar Jannach
TU Dortmund, Germany
dietmar.jannach@
tu-dortmund.de

UMAP'17 July 9–12, 2017, Bratislava, Slovakia
DOI: http://dx.doi.org/10.1145/3079628.3079668

A Comparison of Frequent Pattern Techniques and a Deep Learning Method for Session-Based Recommendation

Iman Kamehkhosh
TU Dortmund
iman.kamehkhosh@tu-dortmund.de

Dietmar Jannach
TU Dortmund
dietmar.jannach@tu-dortmund.de

Malte Ludewig
TU Dortmund
malte.ludewig@tu-dortmund.de

ABSTRACT

Making session-based recommendations, i.e., recommending items solely based on the users' last interactions without having access to their long-term preference profiles, is a challenging problem in various application fields of recommender systems. Using a coarse classification scheme, the proposed algorithmic approaches to this problem in the research literature can be categorized into *frequent pattern mining* algorithms and approaches that are based on *sequence modeling*. In the context of methods of the latter class, recent works suggest the application of recurrent neural networks (RNN) for the problem. However, the lack of established algorithmic baselines for session-based recommendation problems makes the assessment of such novel approaches difficult.

In this work, we therefore compare a state-of-the-art RNN-based approach with a number of (heuristics-based) frequent pattern mining methods both with respect to the accuracy of their recommendations and with respect to their computational complexity. The results obtained for a variety of different datasets show that in every single case a comparably simple frequent pattern method can be found that outperforms the recent RNN-based method. At the same time, the proposed much more simple methods are also computationally less expensive and can be applied within the narrow time constraints of online recommendation.

CCS CONCEPTS

•General and reference →Evaluation; •Information systems →Recommender systems; •Computing methodologies →*Neural networks; Rule learning;*

KEYWORDS

Session-Based Recommendations; Deep Learning; Frequent Pattern Mining; Benchmarking

1 INTRODUCTION

Making recommendations solely based on a user's current session and most recent interactions is a nontrivial problem for recommender systems. On an e-commerce website, for instance, when a visitor is new (or not logged in), there are no long-term user models that can be applied to determine suitable recommendations for this user. Furthermore, recent work shows that considering the user's short-term intent has often more effect on the accuracy of the recommendations than the choice of the method used to build the long-term user profiles [20]. In general, such types of problems are common on e-commerce sites, e.g., when returning users do not log in every time they use the site. The same challenges can, however, be observed also for other application domains, in particular for news and media (music and video) recommendation [21, 33].

The problem of predicting the next actions of users based solely on their sequence of actions in the current session is referred to in the literature as *session-based recommendation*. A number of algorithmic approaches have been proposed over the years to deal with the problem. Early academic approaches, for example, rely on the detection of sequential patterns in the session data of a larger user community. In principle, even simpler methods can be applied. Amazon's "Customers who bought … also bought" feature represents an example that relies on simple co-occurrence patterns to generate recommendations, in that case in the context of the very last user interaction (an item view event). A number of later works then explored the use of Markov models [30, 35, 39], and most recently, researchers explored the use of recurrent neural networks (RNN) for the session-based next-item recommendation problem [16, 17, 38, 42].

Today, RNNs can be considered one of the state-of-the-art methods for sequence learning tasks. They have been successfully explored for various sequence-based prediction problems in the past [5, 9, 11, 18] and in a recent work, Hidasi et al. [16] investigated an RNN variant based on gated recurrent units (GRU) for the session-based recommendations problem. In their work, they benchmarked their RNN-based method GRU4REC with different baseline methods on two datasets. Their results showed that GRU4REC is able to outperform the baseline approaches in terms of accuracy for top-20 recommendation lists.

While these results indicate that RNNs can be successfully applied for the given recommendation task, we argue that the experimental evaluation in [16] does not fully inform us about different aspects of the effectiveness and the practicability of the proposed method. First, regarding the effectiveness it is unclear if the methods to which GRU4REC was compared are competitive. Second, as the evaluation was based on one single training-test split and only using accuracy measures, further investigations are necessary to assess, for example, if some algorithms exhibit certain biases, e.g., to recommend mostly popular items. Third, even if the RNN method is effective, questions regarding the scalability of the method should be discussed, in particular as hyper-parameter optimization for the complex networks can become very challenging in practice.

The goal of this work is to shed light on these questions and in the remainder of this paper we will report the detailed results of comparing a state-of-the-art RNN-based method with a number of computationally more efficient pattern mining approaches in different dimensions.

Workshop on Temporal Reasoning in Recommender Systems, collocated with ACM RecSys'17, Como, Italy.

2 PREVIOUS WORKS

In session-based recommendation problems, we are given a sequence of the most recent actions of a user and the goal is to find items that are relevant in the context of the user's specific short-term intent. One traditional way to determine recommendations given a set of recent items of interest is to apply frequent pattern mining techniques, e.g., based on association rules (AR) [1]. AR are often applied for market basket analysis with the goal to find sets of items that are bought together with some probability [14]. The order of the items or actions in a session is irrelevant for AR-based approaches. Sequential patterns mining (SP) [2] techniques, in contrast, consider the order of the elements in sessions when identifying frequent patterns. In one of the earlier works, Mobasher et al. [32] used frequent pattern mining methods to predict a user's next navigation action. In another work, Yap et al. [47] propose a sequential pattern-mining-based next-item recommendation framework, which weights the patterns according to their estimated relevance for the individual user. In the domain of music recommendation, Hariri et al. more recently [15] propose to mine sequential patterns of latent topics based on the tags attached to the tracks to predict the context of the next song.

A different way of finding item-to-item correlations is to look for sessions that are similar to the current one (*neighbors*), and to determine frequent item co-occurrence patterns that can be used in the prediction phase. Such neighborhood-based approaches were for example applied in the domains of e-commerce and music in [4] or [26]. In some cases and application domains, simple co-occurrence patterns are despite their simplicity quite effective, see, e.g., [20, 40] or [44].

Differently from such pattern- and co-occurrence-based techniques, a number of recent approaches are based on *sequence modeling* using, e.g., Markov models. The main assumption of Markov-model-based approaches in the context of session-based recommendation is that the selection of the next item in a session is dependent on a limited number of previous actions. Shani et al. [35] were among the first who applied first-order Markov chains (MC) for session-based recommendation and showed the superiority of sequential models over non-sequential ones. In the music domain, McFee and Lanckriet [30] proposed a music playlist generation algorithm based on MCs that – given a seed song – selects the next track from uniform and weighted distributions as well as from k-nearest neighbor graphs. Generally, a main issue when applying Markov chains in session-based recommendation is that the state space quickly becomes unmanageable when all possible sequences of user selections should be considered [12, 16].

More recent approaches to sequence modeling for session-based recommendation utilize recurrent neural networks (RNN). RNNs process sequential data one element at a time and are able to selectively pass information across sequence steps [28]. Zhang et al. [49], for example, successfully applied RNNs to predict advertisement clicks based on the users' browsing behavior in a sponsored search scenario. For session-based recommendations, Hidasi et al. [16] investigated a customized RNN variant based on gated recurrent units (GRU) [5] to model the users' transactions within sessions. They also tested several ranking loss functions in their solutions. Later on, in [17] and [42] RNN-based approaches were proposed which leverage additional item features to achieve higher accuracy. For the problem of news recommendation, Song et al. [36] proposed a temporal deep semantic structured model for the combination of long-term static and short-term temporal user preferences. They considered different levels of granularity in their model to process both fast and slow temporal changes in the users' preferences. In general, neural networks have been used for a number of recommendation-related tasks in recent years. Often, such networks are used to learn embeddings of content features in compact fixed-size latent vectors, e.g., for music, for images, for video data, for documents, or to represent the user [3, 6–8, 13, 25, 29, 46]. These representations are then integrated, e.g., in content-based approaches, in variations of latent factor models, or are part of new methods for computing recommendations [7, 8, 13, 27, 37, 43, 45].

In the work presented in this paper, we will compare different existing and novel pattern-mining-based approaches with a state-of-the-art RNN-based algorithm.

3 EXPERIMENT CONFIGURATIONS

3.1 Algorithms

3.1.1 RNN Baseline. GRU4REC is an RNN-based algorithm that uses Gated Recurrent Units to deal with the vanishing or exploding gradient problem.[16]. In our experiments, we used the Python implementation that is shared by the authors online.[1]

3.1.2 Session-based kNN – KNN. The KNN method searches the k most similar past sessions ("neighbors") in the training data based on the set of items in the current session. Since the process of determining the neighbor sessions becomes very time-consuming as the number of sessions increases, we use an special in-memory index data structure (cache) in our implementation. Technically, in the training phase, we create a data structure that maps the training sessions to their set of items and one structure that maps the items to the sessions in which they appear. To make recommendations for the current session s, we first create a union of the sessions in which the items of s appear. This union will be the set of *possible* neighbors of the current session. This is a fast operation as it only involves a cache lookup and set operations. To further reduce the computational complexity of the prediction process, we select a subsample of these possible neighbors using a heuristic. In this work, we took the m most *recent* sessions as focusing on recent trends has shown to be effective for recommendations in e-commerce [23]. We then compute the similarity of these m most recent possible neighbors and the current session and select the k most similar sessions as the neighbor sessions of the current session. Again through lookup and set union operations, we create the set of *recommendable* items R that contains items that appear in one of the k sessions. For each recommendable item i in R, we then compute the KNN score as the sum of the similarity values of s and its neighbor sessions $n \in N_s$ which contains i (Equation 1). The indicator function $1_n(i)$ returns 1 if n contains i and 0 otherwise, see also [4].

$$score_{\text{KNN}}(i, s) = \Sigma_{n \in N_s} sim(s, n) \times 1_n(i) \quad (1)$$

In our experiments, we tested different distance measures to determine the similarity of sessions. The best results were achieved when the sessions were encoded as binary vectors of the item space

[1] https://github.com/hidasib/GRU4Rec

and when using cosine similarity. In our implementation, the set operations, similarity computations, and the final predictions can be done very efficiently as will be discussed later in Section 4.2.2. Our algorithm has only two parameters, the number of neighbors k and the number of sampled sessions m. For the large e-commerce dataset used in [16], the best parameters were, for example, achieved with $k = 500$ and $m = 1000$. Note that the kNN method used in [16] is based on item-to-item similarities while our kNN methods aims to identify similar sessions.

3.1.3 kNN Temporal Extension – TKNN. The KNN method, when using cosine similarity as a distance measure, does not consider the temporal sequence of the events in a session. To be able to leverage the temporal information within the KNN technique, we designed an additional temporal-filtering heuristic for it. The proposed TKNN method uses the same scoring scheme as the KNN method (Equation 1). The only difference is that, given the current session s, we consider item i as being recommendable only if it appears in the neighbor session n directly after a certain item. In our implementation, that certain item is the last item of the current session s. Technically, we therefore use a slightly different implementation of the indicator function of Equation 1: $1_n(i) = 1$ if neighbor session n contains i and (j, i) is a subsequence of n, where j is the last item of the current session and thus the basis to predict the next item.

3.1.4 Simple Association Rules – AR. To assess the strength of simple two-element co-occurrence patterns, we included a method named AR which can be considered as an association rule technique with a maximum rule size of two. Technically, we create a rule $r_{p,q}$ for every two items p and q that appear together in the training sessions. We determine the *weight*, $w_{p,q}$, of each rule simply as the number of times p and q appear together in past sessions. Given the current session s, the AR score of a target item i will be then computed as

$$score_{\text{AR}}(i, s) = w_{i,j} \times 1_{AR}(r_{i,j}) \tag{2}$$

where j is the last item of the current session s for which we want to predict the successor and AR is the set of rules and their weights as determined based on the training data. The indicator function $1_{AR}(r_{i,j}) = 1$ when AR contains $r_{i,j}$ and 0 otherwise.

3.1.5 Simple Sequential Rules – SR. The SR method is a variant of AR, which aims to take the order of the events into account. Similar to the AR method, we create a sequential rule for the co-occurrence of every two items p and q as $r_{p,q}$ in the training data. This time, however, we consider the distance between p and q in the session when computing the weight of the rules. In our implementation, we use the multiplicative inverse as a weight function and set $w_{p,q} = 1/x$, where x is the number of items that appear between p and q in a session. Other heuristics such as a linear or a logarithmic function can also be used. In case that those two items appear together in another session in the training data, the weight of the rule in that session will be added to the current weight. We finally normalize the weight and divide it by the total number of sessions that contributed to the weight. Given the current session s, the SR score of a target item i is then computed as

$$score_{\text{SR}}(i, s) = w_{j,i} \times 1_{SR}(r_{j,i}) \tag{3}$$

Table 1: Dataset characteristics.

	RSC	TMall	#nowplaying	30Music	AotM	8tracks
Sessions	8M	4.6M	95K	170K	83K	500K
Events	32M	46M	1M	2.9M	1.2M	5.8M
Items	38K	620K	115K	450K	140K	600K
Avg. E/S	3.97	9.77	10.37	17.03	14.12	11.40
Avg. I/S	3.17	6.92	9.62	14.20	14.11	11.38

where j is the last item of session s and SR is the set of sequential rules. The indicator function $1_{SR}(r_{j,i}) = 1$ when SR contains $r_{j,i}$ and 0 otherwise.

3.1.6 Hybrid Approaches. We made additional experiments with several hybrids that combine different algorithms. At the end, a weighted combination of the two normalized prediction scores of the algorithms led to the best results in our experiments.

3.2 Datasets and Evaluation Protocol

We performed experiments using datasets from two different domains in which session-based recommendation is relevant, namely e-commerce and next-track music recommendation. The source code and the public datasets can be found online.[2]

3.2.1 E-commerce Datasets. For the e-commerce domain, we chose the *ACM RecSys 2015 Challenge* dataset (RSC) as used in [16]. The RSC dataset is a collection of sequences of click events in shopping sessions. The second e-commerce dataset is a public dataset published for the *TMall* competition. This dataset contains shopping logs of the users on the Tmall.com website.

3.2.2 Music Datasets. We used (a) two datasets that contain *listening logs* of several thousand users and (b) two datasets that comprise thousands of manually created *playlists*.

Listening logs: These used datasets are (almost one-year-long) sub-samples of two public datasets. First, we created a subset of the *#nowplaying* dataset [48], which contains music-related tweets on Twitter. Second, we used the recent *30Music* dataset [41], which contains listening sessions retrieved from Internet radio stations through the Last.fm API.

Playlists: Generally, music playlists are different in nature from listening logs and e-commerce user logs in various ways. Nonetheless, they are designed to be consumed in a listening session and the tracks are often arranged in a specific sequence. The used playlist datasets come from two different music platforms. The *Art-of-the-Mix* dataset (AotM) was published by [31] and contains playlists by music enthusiasts. The *8tracks* dataset was shared with us by the 8tracks platform. A particularity of the 8tracks dataset is that each public playlist can only contain two tracks per artist.

The dataset statistics are shown in Table 1. The total number of sessions is larger for the e-commerce datasets. However, the number of unique items in the music datasets, which corresponds to the number of tracks included in the playlists or the number of played tracks in the listening sessions, is higher than the number of items in e-commerce datasets.

[2] http://ls13-www.cs.tu-dortmund.de/homepage/rectemp2017

The music sessions are on average longer than the e-commerce sessions.[3] The last row of Table 1 shows the average number of unique items in each session ("Avg. I/S"). Comparing this value with the average session length ("Avg. E/S") indicates what we call the item *repetition rate* in each dataset. Including the same track more than once in a playlist is comparably uncommon. Listening to a track more than once during a listening session is, however, common. The difference between the average session length and the average number of items in each session for the e-commerce dataset indicates that re-occurring of the same item in a session is common in the e-commerce domain.

3.2.3 Evaluation Protocol. The general task of the session-based recommendation techniques in our experiment is to predict the next-item view event in a shopping session or to predict the next track that is played in a listening session or is included in a playlist. To evaluate the session-based algorithms, we use the same evaluation scheme as in [16]. We incrementally add events to the sessions in the test set and report the average hit rate (HR), which corresponds to recall in this evaluation setting, and the mean reciprocal rank (MRR), which takes the position of the hit into account. We tested list lengths of 1, 2, 3, 5, 7, 10, 15, and 20. While the experiments in [16] are done without cross-validation, we additionally apply a fivefold sliding-window validation protocol as in [24] to minimize the risk that the obtained results are specific to the single train-test split. We, therefore, created five train-test splits for each dataset. For the listening logs, we used 3 months of training data and the next 5 days as the test data and randomized splits for the playlists as they have no timestamps assigned.

4 RESULTS

4.1 Accuracy Results

Our first experiment used the exact same setup as described in [16], i.e., we use only one training-test split when comparing GRU4REC with our methods. As done in [16], we trained the algorithms using 6 months of data containing 7,966,257 sessions of 31,637,239 clicks on 37,483 items and tested them on the sessions of the next day.

In the subsequent sections, we then report the results of our comparison using the sliding-window validation scheme described above with recommendation list lengths varying from 1 to 20. In all experiments, we tuned the parameters for the different algorithms using grid search and optimized for HR@20 on validation sets (subsets of the training sets). GRU4REC was only optimized with 100 layers as done in [16] due to the computational complexity of the method. To test for statistical significance, we use Wilcoxon signed-rank test with $\alpha = 0.05$.

4.1.1 Results using the original evaluation setup. Table 2 shows the results ordered by the hit rate (HR@20) when using the original setup. We could reproduce the hit rate and MRR results from [16] (using their optimal parameters) for GRU4REC(1000,BPR) and GRU4REC(1000,TOP1), which use 1000 hidden units and the TOP1 and BPR's pairwise ranking loss functions, respectively. In Table 2, we additionally report the results for recommendation list length ten, which might be more important for different application domains.

[3]Note that each session in the TMall dataset is defined as the sequence of actions of a user during one day which results in relatively larger average session length.

Table 2: Results when using the evaluation scheme of [16].

Method	HR@10	MRR@10	HR@20	MRR@20
SR	**0.568**	**0.290**	**0.672**	**0.297**
TKNN	0.545	0.251	0.670	0.260
AR	0.543	0.273	0.655	0.280
KNN	0.521	0.242	0.641	0.250
GRU4REC(1000,BPR)	0.517	0.235	0.636	0.243
GRU4REC(1000,TOP1)	0.517	0.261	0.623	0.268

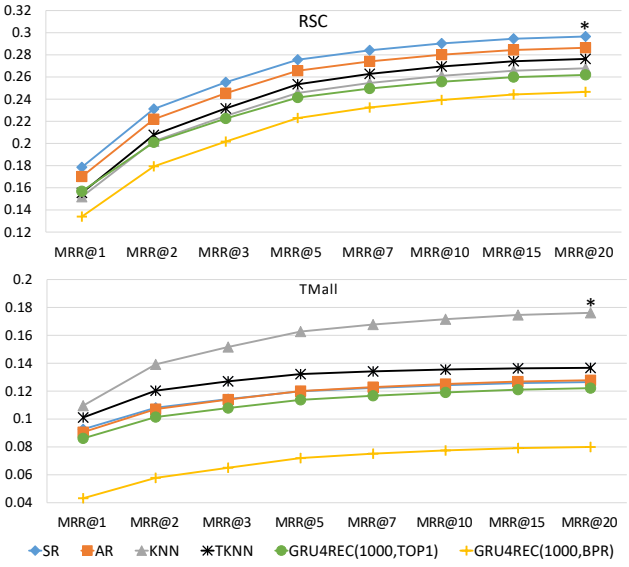


Figure 1: MRR results for the e-commerce datasets (* indicates statistical significance).

The best accuracy results were achieved by the SR method both for the hit rate and MRR and for both list lengths. In terms of the hit rate, every single frequent pattern method used in the experiment was better than the GRU4REC methods. A similar observation can be made also for the MRR, with the exception that the KNN-based methods consistently performed worse than the GRU4REC(1000,TOP1) method on this measure.

4.1.2 E-commerce Datasets. Figure 1 shows the MRR results for the algorithms on the two e-commerce datasets, RSC and TMall. For both datasets, we can observe that most of the frequent pattern methods lead to higher or at least similar MRR values as GRU4REC. There is, however, no clear "winner" across both datasets. The SR method works best for the RSC dataset. On the TMALL dataset, the KNN method outperforms the others, an effect which might be caused by the longer list session lengths for this dataset.[4] In both cases, however, the difference between the winning method and the best-performing GRU4REC configuration is statistically significant. This is indicated by a star symbol in Figure 1.

4.1.3 Listening Logs Datasets. Figure 2 shows the accuracy performance of the algorithms on two selected listening logs datasets.

[4]Remember that the sessions of the TMALL dataset cover the events of one day, as the time stamps in this dataset are given only in the granularity of days.

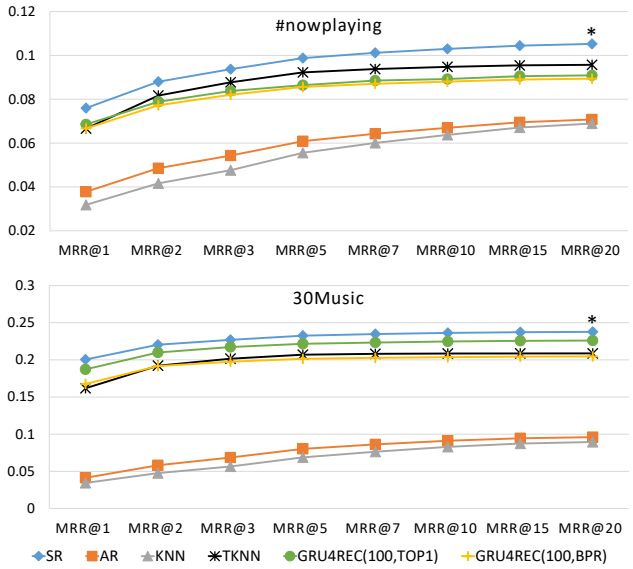


Figure 2: MRR results for the listening log datasets.

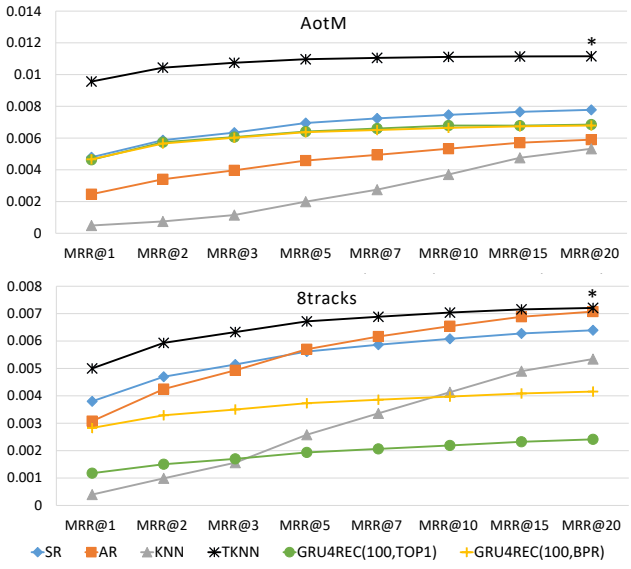


Figure 3: MRR results for the playlist datasets.

Similar to the e-commerce datasets, in all measurements, a frequent pattern approach, namely the SR method, outperforms GRU4REC. Here again, for MRR@20, the recommendations of SR are significantly more accurate than the recommendations of GRU4REC. Note that on the music datasets, we apply GRU4REC(100,TOP1) and GRU4REC(100,BPR), which use 100 hidden units and the TOP1 and BPR's pairwise ranking loss function, respectively.[5]

The TKNN method – the time-aware extension of KNN– works always significantly better than the KNN method on the listening logs datasets. TKNN also outperforms both GRU4REC configurations on the #nowplaying dataset for list lengths larger than 1.

Another observation on the listening logs datasets is that the sequence-based approaches (SR, TKNN and GRU4REC) work significantly better than methods that do not consider the temporal information in data (KNN and AR).

4.1.4 Playlists Datasets. Figure 3 shows the MRR results of the algorithms on the playlists datasets. On both datasets, the temporal extension of kNN, TKNN, leads to the best results across all recommendation list sizes and significantly outperforms both variants of GRU4REC. The performance of all other methods, however, seems to depend on the specifics of the datset. The SR method works well on both datasets. The relative performance of the AR method, however, depends on the dataset and the list length at which the measurement is made.

One interesting observation that we made for the music datasets is that the relative performance of KNN strongly improves in terms of the hit rate[6] when the recommendation list length is increased. This can, for example, be seen in Figure 4, which shows the hit rate results for the #nowplaying dataset. The hit rate of KNN on the #nowplaying dataset that is about 3% for list length one increases

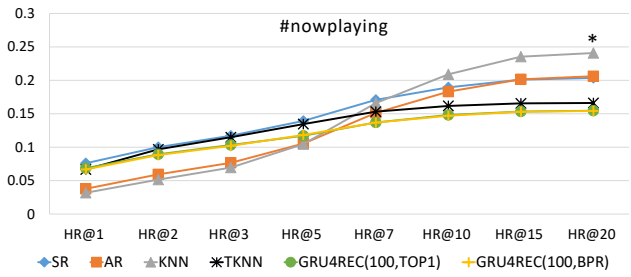


Figure 4: HR results for the #nowplaying dataset.

to 24% for list length 20. At the same time, the hit rate of some of the other methods only slightly increases, e.g., from 6% to 15%. As a result, across all four investigated music datasets, KNN outperforms all other algorithms in terms of HR@20. A similar trend can also be seen for AR, the other non-sequential approach.

4.1.5 Aggregated Ranking of Algorithms. To determine the ranking of different algorithms based on their accuracy results (MRR@20) across all six datasets, we applied the Borda Count (BC) rank aggregation strategy [10]. The results show that SR and TKNN are both ranked first (points = 30), followed by AR as the second best algorithm (20). The GRU4REC method with TOP1 ranking loss is ranked third (18). Finally, KNN and GRU4REC with BPR ranking loss are ranked fourth (15) and fifth (13), respectively.

4.1.6 Hybrid Approaches. We conducted a variety of additional experiments with different hybridization methods as described in Section 3.1.6 to analyze the effect of combining the algorithms. In general, a weighted combination of the two normalized prediction scores of a neighborhood-based and a sequence-based method led to the best results in our experiments. For instance, the combination of KNN and SR with a weight ratio of 3 to 7, WH(KNN,SR:0.3,0.7), outperformed all other individual algorithms on the 30Music dataset.

[5]Repeating the experiments with 1000 hidden layers for the GRU4REC methods did not lead to any better results on the music datasets.
[6]Generally, the hit rate results for the experiments, which we do not include here for space reasons, are similar to the MRR results.

Table 3: Results of the hybrid methods for 30Music.

Method	HR@5	MRR@5	HR@20	MRR@20
SR	0.285	0.233	0.332	0.238
KNN	0.142	0.069	0.342	0.089
GRU	0.275	0.222	0.315	0.226
WH(KNN,SR:0.3,0.7)	**0.298**	**0.243**	0.386	**0.252**
WH(KNN,GRU:0.6,0.4)	0.261	0144	**0.396**	0.159

Another example is combining the normalized score of KNN and GRU4REC(100,TOP1), which can outperform other algorithms in terms of HR@20. The differences between the winning hybrid approaches (printed in bold face in Table 3) and the best performing individual methods in each measurement were statistically significant. Similar results were also achieved for the other datasets, which we do not include here for space reasons.

4.2 Additional Analyses

Since prediction accuracy might not be the only possible relevant quality criterion in a domain [19], we made a number of additional analyses as shown in Figure 5.

4.2.1 Popularity Bias and Catalog Coverage. As in [22], we first measured the average popularity of the top-20 recommendations of the algorithms to assess possible recommendation biases. The popularity of an item is computed based on its number of occurrences in the training dataset. Overall, the recommendations of non-sequential approaches (KNN and AR) shows the highest bias towards popular items. The sequence-based approaches (SR and GRU4REC), in contrast, recommend comparably less popular items.

Additionally, we analyzed the catalog coverage of each algorithm by counting the number of different items that appear in the top-20 recommendation lists of all sessions in the test set. Overall, the recommendation lists of GRU4REC and SR include more different items than the other algorithms. The recommendations of neighborhood methods, KNN and TKNN, on the other hand, focus on smaller sets of items and show a higher concentration bias. This can be explained by considering the sampling strategy of KNN which focuses on a smaller subset of the sessions, e.g., those of the last few days.

4.2.2 Computational Complexity and Memory Usage. We measured the training time as well as the needed memory and time to generate recommendations for each algorithm. On a desktop computer with an Intel i7-4790k processor, training GRU4REC on one split of the RSC dataset with almost 4 million sessions and in its best configuration takes more than 12 hours. This time can be reduced to 4 hours when calculations are performed by the GPU (Nvidia GeForce GTX 960).[7] The KNN method needs about 27 seconds to build the needed in-memory maps, see Section 3.1.2. The well-performing SR method needs about 48 seconds to determine the rule weights. A specific advantage of the latter two methods is that they support incremental updates, i.e., new events can be immediately incorporated into the algorithms. Creating one recommendation list with GRU4REC needed, on average, about 12 ms. KNN needs about 26 ms for this task and SR only 3 ms.

[7] Training the model for 6 month of data using the GPU lasts about 8 hours.

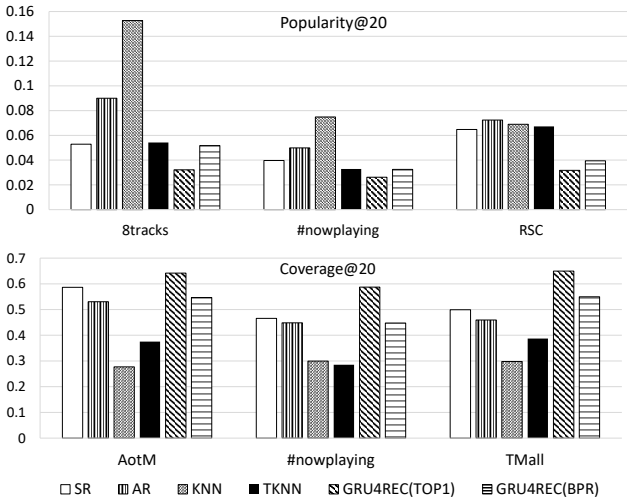


Figure 5: Popularity biases and catalog coverages of the algorithms on three sample datasets.

The raw data used for training the algorithms in this specific experiment (one split of the RSC dataset) occupies about 540 MB of main memory. The data structures used for training SR and KNN occupy about 50 MB and 3.2 GB, respectively. The model created by GRU4REC needs about 510 MB. Note that memory demand of GRU4REC depends on the algorithm parameters and significantly increases with the number of items. For the music and Tmall datasets, the memory demand of GRU4REC exceeded the capacity of our graphics card. Running GRU4REC using the CPU is multiple times slower than when a graphics card is used.

5 CONCLUSION AND FUTURE WORKS

Our work indicates that comparably simple frequent-pattern-based approaches can represent a comparably strong baseline when evaluating session-based recommendation problems. At the end, we could find at least one pattern-based approach that was significantly better than a recent RNN-based method. In particular the SR method was surprisingly effective, despite the fact that both learning and applying the rules is very fast.

Our results also indicates that the "winning" strategy seems to strongly depend on the characteristics of the data sets like average session lengths or repetition rates. Further research is still required to understand this relationship. In our future work, we will investigate the performance of additional session-based algorithms. These algorithms include both ones that are based on Markov models, e.g., Rendle et al.'s factorized Markov chains [34], as well as recently proposed improvements to GRU4REC, e.g., by Tan et al. [38]. We expect that continuously improved RNN-based methods will be able to outperform the frequent pattern based baselines used in the evaluation reported in this paper. These methods can, however, be computationally quite expensive. From a practical perspective, one has therefore to assess depending on the application domain if the obtained gains in accuracy justify the usage of these complex models, which cannot be easily updated online and whose predictions can be difficult to explain.

REFERENCES

[1] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. 1993. Mining Association Rules between Sets of Items in Large Databases. In *SIGMOD '93*. 207–216.

[2] Rakesh Agrawal and Ramakrishnan Srikant. 1995. Mining Sequential Patterns. In *ICDE '95*. 3–14.

[3] Trapit Bansal, David Belanger, and Andrew McCallum. 2016. Ask the GRU: Multi-task Learning for Deep Text Recommendations. In *RecSys '16*. 107–114.

[4] Geoffray Bonnin and Dietmar Jannach. 2014. Automated Generation of Music Playlists: Survey and Experiments. *ACM Computing Surveys* 47, 2 (2014), 26:1–26:35.

[5] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR* abs/1412.3555 (2014).

[6] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *RecSys '16*. 191–198.

[7] Sander Dieleman. 2016. Deep Learning for Audio-Based Music Recommendation. In *DLRS '16 Workshop*. 1–1.

[8] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems. In *WWW '15*. 278–288.

[9] Jeffrey L. Elman. 1990. Finding Structure in Time. *Cognitive Science* 14, 2 (1990), 179 – 211.

[10] Peter Emerson. 2013. The Original Borda Count and Partial Voting. *Social Choice and Welfare* 40, 2 (2013), 353–358.

[11] Alex Graves. 2013. Generating Sequences With Recurrent Neural Networks. *CoRR* abs/1308.0850 (2013). http://arxiv.org/abs/1308.0850

[12] Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural Turing Machines. *CoRR* abs/1410.5401 (2014).

[13] Yupeng Gu, Bo Zhao, David Hardtke, and Yizhou Sun. 2016. Learning Global Term Weights for Content-based Recommender Systems. In *WWW '16*. 391–400.

[14] Jiawei Han and Micheline Kamber. 2006. *Data Mining: Concepts and Techniques (Second Edition)*. Morgan Kaufmann.

[15] Negar Hariri, Bamshad Mobasher, and Robin Burke. 2012. Context-Aware Music Recommendation Based on Latent Topic Sequential Patterns. In *RecSys '12*. 131–138.

[16] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based Recommendations with Recurrent Neural Networks. *CoRR* abs/1511.06939 (2015).

[17] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. In *RecSys '16*. 241–248.

[18] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (Nov. 1997), 1735–1780.

[19] Dietmar Jannach and Gedas Adomavicius. 2016. Recommendations with a Purpose. In *RecSys '16*. 7–10.

[20] Dietmar Jannach, Lukas Lerche, and Michael Jugovac. 2015. Adaptation and Evaluation of Recommendations for Short-term Shopping Goals. In *RecSys '15*. 211–218.

[21] Dietmar Jannach, Lukas Lerche, and Iman Kamehkhosh. 2015. Beyond "Hitting the Hits": Generating Coherent Music Playlist Continuations with the Right Tracks. In *RecSys '15*. 187–194.

[22] Dietmar Jannach, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac. 2015. What recommenders recommend: an analysis of recommendation biases and possible countermeasures. *User Modeling and User-Adapted Interaction* (2015), 1–65.

[23] Dietmar Jannach and Malte Ludewig. 2017. Determining Characteristics of Successful Recommendations from Log Data – A Case Study. In *SAC '17*.

[24] Dietmar Jannach and Malte Ludewig. 2017. When Recurrent Neural Networks meet the Neighborhood for Session-Based Recommendation. In *RecSys 2017*. (forthcoming).

[25] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional Matrix Factorization for Document Context-Aware Recommendation. In *RecSys '16*. 233–240.

[26] Lukas Lerche, Dietmar Jannach, and Malte Ludewig. 2016. On the Value of Reminders within E-Commerce Recommendations. In *UMAP '16*. 27–25.

[27] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep Collaborative Filtering via Marginalized Denoising Auto-encoder. In *CIKM '15*. 811–820.

[28] Zachary Chase Lipton. 2015. A Critical Review of Recurrent Neural Networks for Sequence Learning. *CoRR* abs/1506.00019 (2015).

[29] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *SIGIR '15*. 43–52.

[30] Brian McFee and Gert R. G. Lanckriet. 2011. The Natural Language of Playlists. In *ISMIR '11*. 537–542.

[31] Brian McFee and Gert R. G. Lanckriet. 2012. Hypergraph Models of Playlist Dialects. In *ISMIR '12*. 343–348.

[32] Bamshad Mobasher, Honghua Dai, Tao Luo, and Miki Nakagawa. 2002. Using Sequential and Non-Sequential Patterns in Predictive Web Usage Mining Tasks. In *ICDM '02*. 669–672.

[33] Ozlem Ozgobek, Jon A. Gulla, and Riza C. Erdur. 2014. A Survey on Challenges and Methods in News Recommendation. In *WEBIST '14*. 278–285.

[34] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing Personalized Markov Chains for Next-basket Recommendation. In *WWW '10*. 811–820.

[35] Guy Shani, David Heckerman, and Ronen I. Brafman. 2005. An MDP-Based Recommender System. *The Journal of Machine Learning Research* 6 (Dec. 2005), 1265–1295.

[36] Yang Song, Ali Mamdouh Elkahky, and Xiaodong He. 2016. Multi-Rate Deep Learning for Temporal Recommendation. In *SIGIR '16*. 909–912.

[37] Florian Strub, Romaric Gaudel, and Jérémie Mary. 2016. Hybrid Recommender System based on Autoencoders. In *DLRS '16 Workshop*. 11–16.

[38] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved Recurrent Neural Networks for Session-based Recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems (DLRS '16)*. ACM, 17–22.

[39] Maryam Tavakol and Ulf Brefeld. 2014. Factored MDPs for Detecting Topics of User Sessions. In *RecSys '14*. 33–40.

[40] Roberto Turrin, Andrea Condorelli, Paolo Cremonesi, Roberto Pagano, and Massimo Quadrana. 2015. Large Scale Music Recommendation. In *LSRS 2015 Workshop at ACM RecSys*.

[41] Roberto Turrin, Massimo Quadrana, Andrea Condorelli, Roberto Pagano, and Paolo Cremonesi. 2015. 30Music Listening and Playlists Dataset. In *Poster Proceedings RecSys '15*.

[42] Bart lomiej Twardowski. 2016. Modelling Contextual Information in Session-Aware Recommender Systems with Neural Networks. In *RecSys '16*. 273–276.

[43] Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-Prod2Vec: Product Embeddings Using Side-Information for Recommendation. In *RecSys '16*. 225–232.

[44] Koen Verstrepen and Bart Goethals. 2014. Unifying Nearest Neighbors Collaborative Filtering. In *RecSys '14*. 177–184.

[45] Jeroen B. P. Vuurens, Martha Larson, and Arjen P. de Vries. 2016. Exploring Deep Space: Learning Personalized Ranking in a Semantic Space. In *DLRS '16 Workshop*. 23–28.

[46] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In *KDD '15*. 1235–1244.

[47] Ghim-Eng Yap, Xiao-Li Li, and Philip S. Yu. 2012. Effective Next-items Recommendation via Personalized Sequential Pattern Mining. In *DASFAA'12*. Berlin, Heidelberg, 48–64.

[48] Eva Zangerle, Martin Pichl, Wolfgang Gassler, and Günther Specht. 2014. #Nowplaying Music Dataset: Extracting Listening Behavior from Twitter. In *WISMM '14 Workshop at MM '14*. 21–26.

[49] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014. Sequential Click Prediction for Sponsored Search with Recurrent Neural Networks. In *AAAI '14*. 1369–1375.