Audio-Based Sequential Music Recommendation

Rodrigo Borges
Computer Science Department
University of São Paulo & Tampere University
São Paulo, Brazil
rcborges@ime.usp.br

Marcelo Queiroz
Computer Science Department
University of São Paulo
São Paulo, Brazil
mqz@ime.usp.br

***Abstract*—We propose an audio-based recommendation model designed to predict the upcoming track within a listening session, given the audio associated with the current track. Instead of relying on users' feedback, as most recommenders, the proposed model aims to learn intrinsic audio elements that can be leveraged in the context of sequential recommendation. The proposed model is evaluated using Mel-spectrogram and raw audio as input data and, in its best configuration, was able to predict almost 65% unseen transitions used in the evaluation phase, and 3.5% cold-start transitions, i.e. transitions from tracks that were never seen by the model.**

***Index Terms*—Audio-Based music recommendation, Gated Recurrent Unit, Audio Content**

I. Introduction

Sequential music recommendation methods are usually designed to suggest the track that will be listened to in the near future (next-track), given the information about the tracks that were listened to in the past [1], [2]. These methods are usually trained with datasets containing information about track/track transitions, and the main assumption is that future transitions can be predicted (suggested) based on historical data [3]–[6].

Music recommender methods may also use the audio signals associated with each track for calculating recommendations to users, in which case they are called audio-based methods (as opposed to rating-based methods). Suggesting tracks based on audio content is both technically and conceptually challenging, because it is based on different music listening hypotheses, i.e. that intrinsic elements, such as rhythm and timbre, are also important for music recommendation alongside extrinsic elements, such as listening habits and patterns.

Audio signals associated with tracks may be used by recommendation methods in two ways: audio may serve as the primary resource for providing recommendations, or it may be an auxiliary source of information within rating-based strategies. In the former case, tracks are usually suggested based on similarities defined within the audio domain, which potentially produces novel and fairly unbiased results, but imposes severe limitations on the quality of recommendations [7]. In the latter case, information extracted from the audio signal is incorporated in methods that were originally designed to operate with user/item interaction data, for alleviating situations of item cold-start, i.e. suggesting items that were never seen by the model [8], [9].

In this work, we propose *Audio-Based GRU4REC*, AGRU4REC for short, a method designed to suggest the next track within a listening session given an audio representation (also referred to as audio feature[1]) associated with the current track. The proposed method is inspired in GRU4REC [10], which was originally proposed for recommending the next item to a user, given the information of the previously consumed item. AGRU4REC has no access to any metadata that identifies the current track, and suggests tracks based exclusively on the audio contents.

II. Related Work

Music recommenders that rely solely on audio information have considered the recommendation task as a task of selecting tracks from an audio-based representation space according to a set of tracks that a user has listened to before or is currently listening to [11]. This representation space is built in such a way that similar tracks are supposed to be located close to each other, which makes similarity measuring strategies an essential choice in this context [12]–[15]. There is no consensus however on what it means for two audios to be similar to each other, a broad question which is beyond the scope of this work.

Audio files can also be exploited by rating-based methods for mitigating the cold-start limitation. In [16], users are clustered according to listening habits, and track audios are clustered into music genres. Preferences may be modelled for each user cluster and musical genre, thus mitigating the lack of interaction information for new tracks. The idea that the similarity between tracks can be defined through user access patterns, and that this similarity can be estimated from the audio domain, was explored in [17]. A similar idea, based on learning-to-rank, was proposed in [18]. When given a query track, the ranking system retrieves other tracks sorted by relevance according to user access patterns, and a corresponding ranking is simultaneously learned using the query audio as input. After training, the ranking system is supposed to retrieve relevant tracks when queried with the audio of a new track, i.e., as a *query-by-example* system.

A novel approach is proposed for a dynamic content-based music recommender in [19], [20]. Ratings given by users are modelled as a combination of two factors, an affinity for

[1] This method presupposes a choice of a specific audio feature representation for the tracks, which is simply referred to in the sequel as "audio feature"; in the experimental part of this paper, we use Mel spectrograms and raw audio waveforms.

the audio content, and a factor responsible for diversity. The affinity for audio features is modelled as an inner product of a user preference variable and the audio features of listened tracks. The diversity is implemented with an exponential curve that prevents the recommender to repeat a song that was recently suggested. The system, however, iterates through every track for maximizing the quantile value of the estimated distribution, inspired by Bayesian-UCB [21], and this can be time-consuming.

To the best of our knowledge, one single method was already proposed in the specific context of audio-based sequential music recommendation [22]. The method, named Adaptive Linear Mapping Model (ALMM), adapts the content-boost methodology [8] to the next-track recommendation task. ALMM decomposes a set of personalized transition matrices as a product of three latent matrices: *user embedding*, *previous-track embedding*, and *next-track embedding*, in a similar fashion to FPMC [23]. The two last matrices, the ones associated with the previous and next tracks, are also factorized as linear products of an audio feature matrix and auxiliary matrices that are learned during the optimization process. The final recommendation score for a specific track can be calculated directly from its audio features with the help of the auxiliary matrices.

III. Method

Audio-Based GRU4REC is an audio-based recommendation model composed of one Convolutional Neural Network (CNN), one Gated Recurrent Unit (GRU), and one Multi-Layer Perceptron (MLP). The model is trained to predict the next track within a listening session, given approximately 3 seconds of the audio associated with the current track. More details about the model are presented in the sequel.

A. Problem Definition

A listening session of size T is denoted as $\{s^{(1)}, s^{(2)}, \ldots, s^{(T)}\}$, where $s^{(t)} \in S$ is the track observed at instant t, with $0 < t \leq T$. Typically, a temporal dependency among consecutive tracks is assumed according to the conditional probabilities $p(s^{(t)}|s^{(t-1)}, \ldots,\ s^{(t-m)})$, taking the previous m tracks into consideration. Here, we assume a dependency between the current track and the previous audio features, expressed as $p(s^{(t)}|A^{(t-1)}, \ldots, A^{(t-m)})$, where $A^{(t)}$ is the audio feature (a Mel spectrogram or any other selected representation) associated with track $s^{(t)}$ observed at instant t.

Our aim is to train a model that is able to predict the upcoming track $s^{(t+1)}$ given the audio feature associated with the current track $A^{(t)}$. In other words, a model that estimates $p(s^{(t+1)}|A^{(t)})$.

B. Audio-Based GRU4REC

Audio-Based GRU4REC (AGRU4REC) was inspired in the method GRU4REC [10], originally proposed as a track/track transition model. AGRU4REC suggests the next track within a listening session given an audio feature.

The model consists of three stages described as follows. First, a function $f(\cdot)$ maps an audio feature $A^{(t)}$ to an audio embedding $D^{(t)}$, in such a way that $f(A^{(t)}) = D^{(t)}$. Second, another embedding is calculated by a function $g(\cdot)$ with memory, i.e. a function that is able to store its parameters so they can be used in the next round of recommendation. Let $g(\cdot)$ be the function that maps the audio embedding to the new embedding, named sequence-aware embedding $E^{(t)}$, and let $H^{(t)}$ be the current state of the function $g(\cdot)$. At instant t, a sequence-aware embedding is calculated considering the state stored at instant $t-1$, in such a way that $g(D^{(t)}, H^{(t-1)}) = E^{(t)}$. When a listening session ends, the state H is reset, assuming that listening sessions are independent of each other. Finally, a function $q(\cdot)$ maps the session-aware embedding to the scores corresponding to the next track in session $Y^{(t+1)}$, in such a way that $q(E^{(t)}) = Y^{(t+1)}$. The output $Y^{(t+1)}$ has size $|S|$, and contains the scores attributed to each track $s \in S$. The highest the score attributed to a track, the higher the probability that this track is the next one in a current listening session.

Function $f(\cdot)$ is implemented with a CNN, function $g(\cdot)$ is implemented with a GRU network, and function $q(\cdot)$ is implemented with an MLP. The hidden state of the GRU network H is initialized containing zeros, and the training process is summarized in the sequel.

The audio embedding $D^{(t)}$ is first obtained from its corresponding audio feature $A^{(t)}$ (Figure 1, left) and it propagates to the GRU network. The *reset* ($R^{(t)}$) and *update* ($Z^{(t)}$) gates of the GRU network are the first parameters to be adjusted, respectively, with equations:

$$R^{(t)} = \sigma(\mathbf{W}^{rs}D^{(t)} + \mathbf{W}^{rh}H^{(t-1)} + B_r) \qquad (1)$$

$$Z^{(t)} = \sigma(\mathbf{W}^{zs}D^{(t)} + \mathbf{W}^{zh}H^{(t-1)} + B_z) \qquad (2)$$

where $\mathbf{W}^{xy}$ are weight matrices for mapping x to y, to be adjusted during training, and B_r and B_z are biases. Sigmoid is applied to transform the input values to the range (0,1). When presenting the audio embedding corresponding to the first track of each listening session, $H^{(t-1)}$ is set equal to zero for ensuring independency between sessions, and the second terms of both equations are not considered in the calculation of $R^{(t)}$ and $Z^{(t)}$.

A *candidate hidden state* $N^{(t)}$ is calculated, incorporating the reset gate:

$$N^{(t)} = tanh(\mathbf{W}^{ns}D^{(t)} + \mathbf{W}^{nh}(R^{(t)} \odot H^{(t-1)}) + B_n)) \quad (3)$$

where $\odot$ is the Hadamard (elementwise) product and $tanh$ is applied to ensure that the values remain in the interval (-1,1). For now, when entries in the reset gate are set to 1, then the candidate's new state reminds the hidden state calculated for standard RNN. When the reset gate is set equal to 0 the architecture resembles a standard MLP having $D^{(t)}$ in the input.

The final hidden state incorporates the update gate, and is calculated with:

$$H^{(t)} = (1 - Z^{(t)}) \odot N^{(t)} + Z^{(t)} \odot H^{(t-1)}, \qquad (4)$$

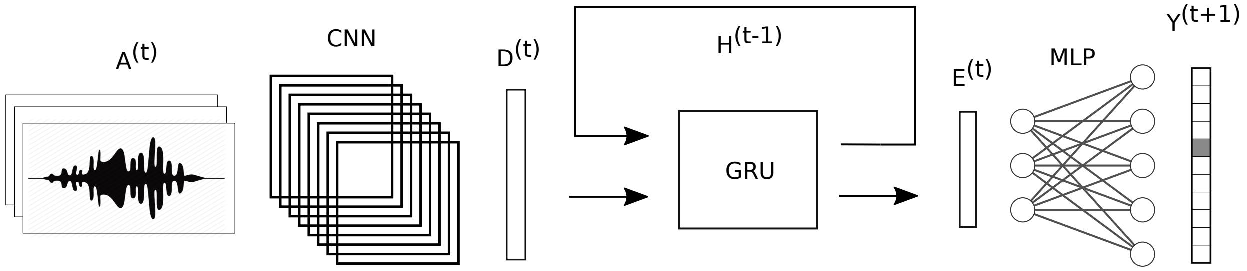

Fig. 1. *Audio-Based GRU4REC* (AGRU4REC), a spatio-temporal recommendation model inspired in GRU4REC [10].

where $H^{(t-1)}$ is the hidden state at time $t-1$. The update gate $Z^{(t)}$ determines to which extent the new hidden state $H^{(t)}$ is inherited from the previous hidden state $H^{(t-1)}$, and how much of the new candidate state is considered.

In this model, the session-aware embedding $E^{(t)}$ is a copy of $H^{(t)}$, and $Y^{(t+1)}$ is obtained from $E^{(t)}$, considering that $Y^{(t+1)} = tanh(q(E^{(t)}))$ (Figure 1, right).

The model is trained in mini-batches, and the goal is to minimize the TOP1 loss function, calculated as [10]:

$$Loss = \frac{1}{|S|}\sum_{j=1}^{|S|} \sigma(\hat{y}_j - \hat{y}_i) + \sigma(\hat{y}_j^2), \tag{5}$$

where $\hat{y}_i$ is the score given to the right track $s^{(t+1)}$, and $\hat{y}_j$ is the score given to any other track observed within a mini-batch (negative samples). An extra regularization term forces negative samples to have scores close to zero.

IV. Experiments

A. Dataset

LFM-1b is among the biggest datasets publicly available containing music consumption information [24]. It contains data extracted from the LastFM[2] streaming platform from 2005 to 2014 in the format (user, artist, album, track, timestamp), where each row is associated with a listening event. We separated the user-track interactions from the year 2013, taking into account that this was the most recent year available with a relevant number of interactions.

In order to separate dataset entries in listening sessions, we separated the tracks listened by the same user, ordered these events by timestamp, and sessions are assumed as non-interrupted sequences of listening events. More specifically, a session is assumed as starting with the first track of the list, and whenever an interval between adjacent tracks is longer than 30 minutes, the current session is finished and the following track is assumed to belong to a new session.

Audio files corresponding to tracks that were listened to by at least 10 users were downloaded from the Spotify website with the help of their API[3] and of the Spotipy[4] Python library. The URL of a 30s mp3 preview for each song is included and was used to download the corresponding files. In total, mp3 previews for 237,705 tracks were downloaded. All the information downloaded from the Spotify website was exclusively applied for research purposes.

B. Data Partition and Feature Extraction

Around 19,000,000 non-interrupted listening sessions were derived from user-track interaction data, considering intervals shorter than 30 minutes between listening events as a criterion for including tracks in the same session. Among these sessions, 889,968 included only tracks associated with downloaded audio previews and were considered in the experiments. Listening sessions containing less than 5 and more than 100 events were removed, as well as sessions with less than 2 unique tracks (i.e. sessions containing a single song multiple times). Finally, we split the whole set into training/validation/test subsets according to proportions of 80/10/10% and ordered by timestamp (training on the oldest 80% sessions, validating on the next 10% and testing on the newest 10%). The idea is to simulate a situation when the system is exposed to interactions that happened in the past and evaluate its performance with listening sessions that happen in the future.

The raw audios were extracted from the mp3 previews with the Librosa Python library[5], with a sampling rate of 22,050 Hz. Mel-spectrograms were also computed from these excerpts with the same library, using 128 Mel filters and FFT window and hop sizes of 2048 and 512 samples, respectively, and the Hann window function, resulting in Mel-spectrograms of dimension 128 × 1292. The magnitudes of the Mel-spectrograms were compressed by a nonlinear curve $\log(1 + C|A|)$ where $|A|$ is the magnitude and C is set to 10, as suggested in [25].

C. CNN Architectures and Training

In order to compare the performances of AGRU4REC using raw waveforms and Mel-spectrograms as input, as suggested in [26], we implemented the method using one-dimensional (1D) and two-dimensional (2D) CNNs. The 1D CNN was inherited from [25], and the 2D CNN was implemented in such a way that its architecture (number of layers, activation functions, dropout rate, normalization layers) was kept as similar as possible to the 1D architecture.

The models were trained with audio features corresponding to approximately 3 seconds of the audio previews: 59,049 samples in the case of 1D CNN, and 115 FFT frames in the

[2] https://www.last.fm/

[3] https://developer.spotify.com/documentation/web-api/

[4] https://github.com/plamere/spotipy

[5] https://librosa.org

TABLE I

Results measured for the next-track prediction task. Results are reported separately for all transitions in the test subset (overall), and for transitions from tracks that appear in the test slice for the first time (cold-start). Whenever the results measured for one method are better than the others, the values are highlighted.

		REC@1	**REC@20**	**REC@100**	**MRR@1**	**MRR@20**	**MRR@100**
Overall	ALMM	0.018	0.222	0.421	0.018	0.053	0.058
	AGRU4REC (MEL)	0.222	0.497	0.640	0.222	0.292	0.296
	AGRU4REC (RAW)	**0.245**	**0.512**	**0.651**	**0.245**	**0.314**	**0.317**
Warm-Start	ALMM	0.019	0.234	0.445	0.019	0.056	0.061
	AGRU4REC (MEL)	0.237	0.529	0.681	0.237	0.311	0.315
	AGRU4REC (RAW)	**0.262**	**0.547**	**0.694**	**0.262**	**0.335**	**0.339**
Cold-Start	ALMM	0.001	0.005	0.021	0.001	0.002	0.002
	AGRU4REC (MEL)	0.001	0.008	0.027	0.001	0.002	0.002
	AGRU4REC (RAW)	**0.002**	**0.013**	**0.036**	**0.002**	**0.003**	**0.004**

case of 2D CNN. The architecture selected for mapping Mel-spectrograms to audio embeddings has 5 convolutional layers, followed by a linear layer, applied for reducing the embedding size. The architecture selected for mapping raw waveforms to audio embeddings has 11 convolutional layers, followed by a linear layer, also applied for reducing the embedding size. The source code for reproducing the experiments is publicly available[6]. In order to improve the model's generalization ability, 30-second audio features were separated into 10 equally-sized slices, and at each training round a random slice is chosen to train the model.

D. Previous Approaches

The original ALMM method suggests the use of personalized transition matrices, but preliminary results showed that using a single transition matrix produced better results, and so results reported here use the latter strategy. The ALMM method was trained and evaluated on the LFM-1b dataset using the same audio codeword histograms defined in [9], in order to preserve the structure of the original method.

E. Evaluation Metrics

The AGRU4REC and ALMM methods were trained for 35 epochs, and were evaluated according to their performances in the test subset. The Recall (REC@K) was used for measuring the recommendation accuracy, and Mean Reciprocal Rank (MRR@K) was used for measuring the quality of the ranking in the results. Both metrics were implemented according to [27].

In the case of AGRU4REC, the input audio is sliced in 10 equally-sized slices, as mentioned before, and K tracks are recommended for each slice. The results are measured considering the recommendations calculated for all slices.

V. Results

All methods were evaluated in transitions between tracks that both appeared in the training and test set, a context referred to as warm-start (this is the most common recommendation scenario). In this context, which may be considered the easiest scenario, AGRU4REC produced better results than ALMM and the best results for each considered metric. In Table I it can be seen that improvements obtained by AGRU4REC relative to ALMM range from factors of 1.6× (REC@100) up to 11× (REC@1 and MRR@1), where best improvements do occur for the more demanding metrics which only consider the first position of the corresponding ranked lists.

The training/validation/test splits refer to sessions (and not tracks), but it is important to differentiate between tracks that did appear in training sessions and those that did not. Among all 237,705 tracks used in the experiments, 5,459 were observed in the test subset for the first time. Transitions from these tracks to tracks appearing in the training subset were considered cold-start transitions and were evaluated separately (Table I). When assessing these transitions, AGRU4REC presented better results compared to ALMM, both in terms of Recall and MRR, for all values of K, with improvements ranging from 50% (REC@100) to 160% (REC@20). The low values are indicative of the difficulty of sequential cold-start prediction, but nevertheless, allow a comparison between these methods.

The 1D CNN trained with raw waveforms turned out to be more versatile than the 2D CNN trained with Mel-spectrograms, achieving better results than the latter in every metric and every scenario considered. This difference might be attributed both to the audio feature used, respectively raw waveforms or Mel-spectrogram, and to the CNN architecture, which is 1D or 2D, respectively.

One example of an audio-based recommendation instance is presented in Table II. AGRU4REC was able to recommended non-obvious track/track transitions, from different artists, and with consistent results.

VI. Conclusions

The proposed audio-based recommendation models achieved satisfactory accuracy and ranking quality. These models can be also used as auxiliary recommendation models, to be consulted whenever the current track that a user might be listening to is not known by a feedback-based recommendation model. According to the results presented here, AGRU4REC can improve the accuracy of any current-track cold-start recommendation model up to 3.6%.

[6] https://www.github.com/rcaborges/AGRU4REC

TABLE II
One example of audio-based recommendation. The table shows one track/track transition from the test subset, and the tracks recommended by AGRU4REC sorted by relevance.

	Track	Artist
Previous Track	Blunderbuss	Jack White
Next Track	Speak to Me/Breathe	Pink Floyd
Rec. Tracks	Atom Heart Mother	Pink Floyd
	Cirrus Minor	Pink Floyd
	Astronomy Domine	Pink Floyd
	Comfortably Numb	Pink Floyd
	If	Pink Floyd
	Speak to Me/Breathe	Pink Floyd
	Shine on You Crazy Diamond	Pink Floyd
	One of These Days	Pink Floyd
	Let There Be More Light	Pink Floyd
	Wish You Were Here	Pink Floyd

One potential application for audio-based models is the generation of playlists given a local collection of tracks stored on a user's device. In this specific case, AGRU4REC could be applied for generating recommendations based on the stored audio files, even without having access to metadata associated with these tracks. This can be particularly interesting in a situation where users are trying to expand their music collection with tracks that are related to the ones they already have.

VII. Acknowledgements

During the development of this project, the first author received financial support from CAPES Grant 88881.189985/2018-01 and the second author received financial support from CNPq Grant 310141/2022-2.

References

[1] Brian McFee and Gert R. G. Lanckriet, "The natural language of playlists," in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, 2011, pp. 537–542.

[2] Bruno L. Pereira, Alberto Ueda, Gustavo Penha, Rodrygo L. T. Santos, and Nivio Ziviani, "Online learning to rank for sequential music recommendation," in *Proceedings of the 13th ACM Conference on Recommender Systems*, 2019, RecSys '19, p. 237–245.

[3] Mehdi Hosseinzadeh Aghdam, Negar Hariri, Bamshad Mobasher, and Robin Burke, "Adapting recommendations to contextual changes using hierarchical hidden markov models," in *Proceedings of the 9th ACM Conference on Recommender Systems*, 2015, RecSys '15, p. 241–244.

[4] Malte Ludewig and Dietmar Jannach, "Evaluation of session-based recommendation algorithms," *User Modeling and User-Adapted Interaction*, vol. 28, no. 4-5, pp. 331–390, 2018.

[5] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach, "Sequence-aware recommender systems," *ACM Comput. Surv.*, vol. 51, no. 4, 2018.

[6] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Jiajie Xu, Victor S.Sheng S.Sheng, Zhiming Cui, Xiaofang Zhou, and Hui Xiong, "Recurrent convolutional neural network for sequential recommendation," in *The World Wide Web Conference*, 2019, WWW '19, p. 3398–3404.

[7] Arthur Flexer, Martin Gasser, and Dominik Schnitzer, "Limitations of interactive music recommendation based on audio content," in *AM '10, The 5th Audio Mostly Conference, Piteå, Sweden, September 15-17, 2010*. 2010, p. 13, ACM.

[8] Peter Forbes and Mu Zhu, "Content-boosted matrix factorization for recommender systems: Experiments with recipe recommendation," in *Proceedings of the Fifth ACM Conference on Recommender Systems*, 2011, RecSys '11, p. 261–264.

[9] Aäron van den Oord, Sander Dieleman, and Benjamin Schrauwen, "Deep content-based music recommendation," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, 2013, NIPS'13, p. 2643–2651.

[10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk, "Session-based recommendations with recurrent neural networks," in *4th International Conference on Learning Representations, ICLR*, 2016.

[11] Pedro Cano, Markus Koppenberger, and Nicolas Wack, "Content-based music audio recommendation," in *Proceedings of the 13th Annual ACM International Conference on Multimedia*, 2005, MULTIMEDIA '05, p. 211–212.

[12] Malcolm Slaney, Kilian Q. Weinberger, and William White, "Learning a metric for music similarity," in *ISMIR 2008, 9th International Conference on Music Information Retrieval, Drexel University, Philadelphia, PA, USA, September 14-18, 2008*, 2008, pp. 313–318.

[13] B. Logan and A. Salomon, "A music similarity function based on signal analysis," in *IEEE International Conference on Multimedia and Expo, 2001. ICME 2001.*, 2001, pp. 745–748.

[14] Dmitry Bogdanov, Joan Serrà, Nicolas Wack, Perfecto Herrera, and Xavier Serra, "Unifying low-level and high-level music similarity measures," *IEEE Transactions on Multimedia*, vol. 13, no. 4, pp. 687–701, 2011.

[15] Matthew D. Hoffman, David M. Blei, and Perry R. Cook, "Content-based musical similarity computation using the hierarchical dirichlet process," in *ISMIR 2008, 9th International Conference on Music Information Retrieval, Drexel University, Philadelphia, PA, USA, September 14-18, 2008*, 2008, pp. 349–354.

[16] Qing Li, Byeong Man Kim, Dong Hai Guan, and Duk whan Oh, "A music recommender based on audio features," in *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2004, SIGIR '04, p. 532–533.

[17] B. Shao, D. Wang, T. Li, and M. Ogihara, "Music recommendation based on acoustic features and user access patterns," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 8, pp. 1602–1611, 2009.

[18] B. McFee, L. Barrington, and G. Lanckriet, "Learning content similarity for music recommendation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 8, pp. 2207–2218, 2012.

[19] Zhe Xing, Xinxi Wang, and Ye Wang, "Enhancing collaborative filtering music recommendation by balancing exploration and exploitation," in *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, 2014, pp. 445–450.

[20] Xinxi Wang, Yi Wang, David Hsu, and Ye Wang, "Exploration in interactive personalized music recommendation: A reinforcement learning approach," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 11, no. 1, 2014.

[21] Emilie Kaufmann, Olivier Cappe, and Aurelien Garivier, "On bayesian upper confidence bounds for bandit problems," in *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, 2012, vol. 22 of *Proceedings of Machine Learning Research*, pp. 592–600.

[22] Szu-Yu Chou, Yi-Hsuan Yang, Jyh-Shing Roger Jang, and Yu-Ching Lin, "Addressing cold start for next-song recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*, 2016, RecSys '16, p. 115–118.

[23] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *Proceedings of the 19th International Conference on World Wide Web*, 2010, WWW '10, pp. 811–820.

[24] Markus Schedl, "The lfm-1b dataset for music retrieval and recommendation," in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, 2016, ICMR '16, pp. 103–110.

[25] Jongpil Lee, Jiyoung Park, Keunhyoung Luke Kim, and Juhan Nam, "Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms," *CoRR*, vol. abs/1703.01789, 2017.

[26] Sander Dieleman and Benjamin Schrauwen, "End-to-end learning for music audio," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 6964–6968.

[27] Arthur Tofani, Rodrigo Borges, and Marcelo Queiroz, "Dynamic session-based music recommendation using information retrieval techniques," *User Model. User Adapt. Interact.*, vol. 32, no. 4, pp. 575–609, 2022.