Content based User Preference Modeling in Music Generation

Xichu Ma
ma_xichu@u.nus.edu
National University of Singapore, Singapore

Yuchen Wang
yuchen_wang@u.nus.edu
National University of Singapore, Singapore

Ye Wang
wangye@comp.nus.edu.sg
National University of Singapore, Singapore

ABSTRACT

Automatic music generation (AMG) has been an emerging research topic in AI in recent years. However, generating user-preferred music remains an unsolved problem. To address this challenge, we propose a hierarchical convolutional recurrent neural network with self-attention (CRNN-SA) to extract user music preference (UMP) and map it into an embedding space where the common UMPs are in the center and uncommon UMPs are scattered towards the edge. We then propose an explainable music distance measure as a bridge between the UMP and AMG; this measure computes the distance between a seed song and the user's UMP. That distance is then employed to adjust the AMG's parameters which control the music generation process in an iterative manner, so that the generated song will be closer to the user's UMP in every iteration. Experiments demonstrate that the proposed UMP embedding model successfully captures individual UMPs and that our proposed system is capable of generating user-preferred songs.

CCS CONCEPTS

• **Applied computing** → **Sound and music computing**; • **Computing methodologies** → *Knowledge representation and reasoning*; • **Information systems** → **Personalization**.

KEYWORDS

User Modeling; Music Preference; Music Generation; Contrastive Learning

ACM Reference Format:
Xichu Ma, Yuchen Wang, and Ye Wang. 2022. Content based User Preference Modeling in Music Generation. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22), Oct. 10–14, 2022, Lisboa, Portugal.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3503161.3548169

1 INTRODUCTION

Automatic music generation (AMG) has been an emerging research topic in AI in recent years. However, current AMG models mainly cope with musicality. Generating music that matches user music preference (UMP) remains an unsolved problem due to the following challenges: 1) The underlying principles that form user music preference are still unclear [48]. 2) It is hard to find suitable features and representations for UMP modeling. 3) It is difficult for users to describe UMPs, especially for non-specialists without music knowledge. 4) Incorporating UMP as meaningful rules or conditions for AMG can be tricky.

To address these challenges, we make the first attempt to utilize and employ user modeling, the core technique that has been studied in many recommender systems. This technique allows us to extract UMP from user feedback and employ the extracted UMP to control AMG in an interpretable manner.

As shown in Figure 1, we first propose a hierarchical convolutional recurrent neural network with a self-attention (CRNN-SA) based UMP embedding model, which extracts multifaceted audio and musical features from a user's listening histories. Through subtly designed contrastive loss, the UMP embedding model manages to scatter UMPs in the embedding space where the following distribution pattern is observed: popular preferences anchor in the space's center like a planet and unique music tastes surround them like satellites. Next, we raise an explainable embedding distance model, which serves as the bridge linking UMP and AMG. The model compares an input song to a user's UMP and yields a distance value, which guides how to generate a song that approaches

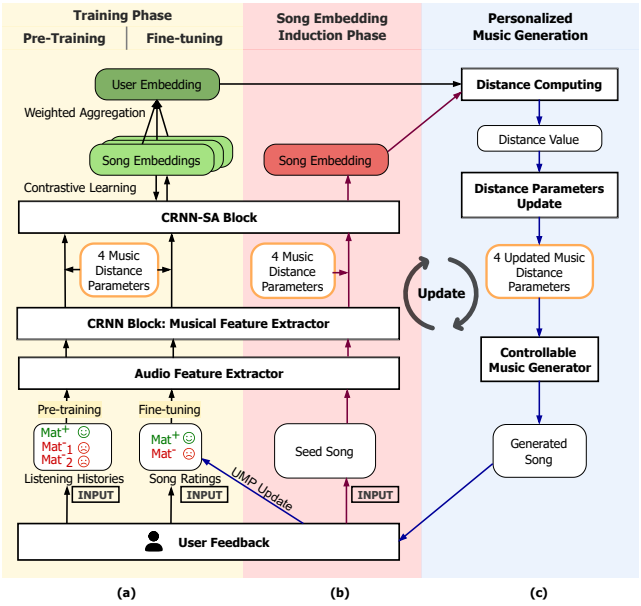

Figure 1: Workflow of the UMP Extraction and Personalized Music Generation System. (a) Training Phases of the UMP Embedding Model. (b) Process of Computing the Embedding of an Input Seed Song. (c) Process of Personalized Music Generation.

MM '22, October 10–14, 2022, Lisboa, Portugal

ACM ISBN 978-1-4503-9203-7/22/10.
https://doi.org/10.1145/3503161.3548169

the user's preference. Finally, with the help of a controllable music generator that takes in a seed song and four controlling parameters to generate a new song [8], we convert the distance value to the four parameters and generate a new song that's closer to the UMP than the seed song.

We conduct both objective and subjective experiments, and the results show that our proposed system successfully embedded diverse UMPs and effectively generated songs that users like better than the input songs. Therefore, our proposed UMP-aware AMG system is expected to improve the quality of music-related services, enhancing user experience and user stickiness.

The main contributions of this work are three-folded:

- To the best of our knowledge, we make the first attempt to capture UMPs with an innovative CRNN-SA model using contrastive learning to differentiate between common and unique music preferences.
- We raise an embedding distance model which computes the cross-modal distance between a user's UMP and a seed song based on four musical characters. The model successfully functions as the intermedium between UMP and personalized AMG.
- We incorporate the computed distance into a controllable AMG model with a style transfer technique to update the four music distance parameters, which in turn creates a new song closer to the UMP.

The remainder of the paper is organized as follows. We first review relevant work in Section 2. We then present the methodology of the work in Section 3, detailing the designs of the proposed UMP model, the contrastive loss function, the embedding distance model, and the personalized AMG model. Next, we introduce the objective and subjective experiments in Sections 4 and 5. Finally, we discuss and conclude the paper in Sections 6 and 7 respectively.

2 RELATED WORK

2.1 User Preference Modeling

Learning users' preferences and personalizing users' experiences are important factors in increasing user satisfaction [49]. User preference modeling has been studied in many fields such as music, fashion, and e-commerce for item recommendation [6, 7, 29, 40, 60]. People's preferences in these fields, however, change over time at different rates. Music preference is one of the most stable, formed as early as adolescence with little fluctuation afterwards [31]. Thus, related work in music preference modeling rarely emphasizes variations in the temporal dimension. In music recommender systems, studies attempt to learn users' preferences by jointly learning music features and users' personal information such as cultural background [60] or listening behaviors such as frequency or recency [29] and then predict the likability of an input music genre. Common techniques include collaborative filtering (CF) [24], content-based filtering (CBF) [41], and hybrid solutions, which integrate the former two by applying collaborative filtering on learned content features [44]. Various deep learning architectures, including CNNs [19, 44, 55], RNNs [22], Deep Belief Networks (DBNs) [58], and GANs [9, 15], have also been used to extract UMP from the user-item interaction matrix. The correlation of UMP with other characteristics, such as emotions and personalities, have also been studied [38]. UMPs are usually modeled as the aggregation of all their preferred artists and songs or as the major group of samples after clustering operations [13, 47].

In contrast, user preferences in visual arts such as fashion experience periodic changes as influenced by social trends. There are studies applying the LSTM-encoder-decoder-framework-based Knowledge Enhanced Recurrent Network (KERN) to embed users' age, gender, and location information obtained from social media posts, then learn their sequential interaction with fashion elements and ultimately predict the coming fashion trend [33, 34].

User preferences in online shopping change more frequently. To adapt to this high intensity of preference change, existing work in e-commerce recommendations and advertisements employs reinforcement learning (RL) with deep Q-Network (DQN) and neural interactive collaborative filtering (NICF) to update users' shopping preferences in real time based on their current feedback for the recommended items. [61, 62].

Music preferences are less time dependent than preferences in other fields and are thus possible to obtain from listening histories [31]. Current studies on music preference are mostly for music recommender systems, which are difficult to generalize to AMG. Thus, we notice the significant research value of modeling and incorporating UMP in AMG and other music-related tasks.

2.2 Music Generation

Existing AMG approaches can be categorized into several main classes [5]: 1) *Statistics-based* methods like Markov chains, which generate notes, chords, and high-level structures such as bars and sections based on their previous frequency of appearance [3, 23, 37, 46, 50]. 2) *Rule-based* approaches, which adopt traditional music theories or create new song writing rules, encode chords/notes as grammar symbols and generate music based on rules and grammar like natural language processing (NLP) tasks [25, 57]. Nevertheless, 1) and 2) both suffer from repetitive content as they are limited to a certain corpus or rule set. 3) In *Evolutionary computation* methods, genetic algorithms start with an existing piece or a random solution and iteratively approach results that meet the defined requirements [2, 12, 32, 36, 45]. They are beneficial for improvisation and composition but may still lack variety in styles and structures. 4) In *Multi-agent systems*, multiple agents with perception and actions (each of which may represent a personality or an emotion) interact with each other to model musical behaviors [28]. An example is the Belief-Desire-Intention Architecture, where two agents, each targeting a different aspect of composition (i.e., theory and style), interactively create harmonic pieces [42]. 5) More recently, CNNs [53, 59], RNNs [14, 21, 35, 51], GANs [1, 11], and Transformers [10, 20] have been frequently used in *Deep learning-based* melody generations as they have commonly recognized advantages in extracting patterns, handling long-dependencies, and imitating realistic pieces, although their mechanisms and generated content are less interpretable and controllable.

Unfortunately, none of these approaches can generate customized music favored by a specific user because current models concentrate on basic requirements like musicality rather than UMP. Therefore, our work aims to fill this gap by applying a UMP embedding model to AMG.

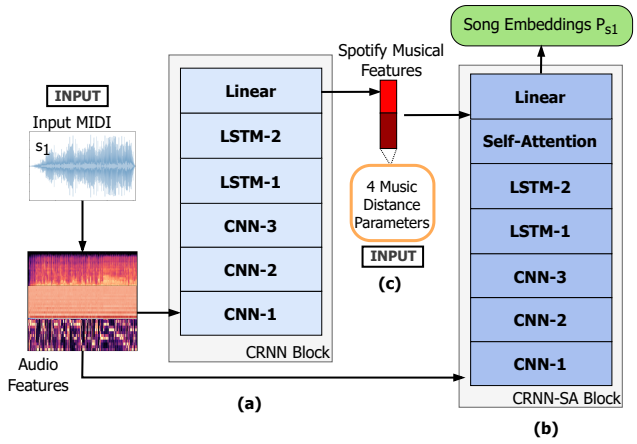

Figure 2: The Architecture of the Hierarchical CRNN-SA based UMP Embedding Model.

2.3 Limitations and Proposed Improvements

First, several works on music recommender systems randomly label songs that do not appear in users' listening histories as negative samples, making an unwarranted assumption on how the user may react to an unheard piece of music. The quantity of negative (i.e., unheard) samples surpassing positive samples can lead to some models for likability prediction "cheating" by simply increasing the probability of making "dislike" prediction. He et al. reported the bias brought by sampling non-label histories as negative, proposing popularity aware weighting as a potential strategy. As such, our work [17]. As such, our work counterposes users' listened songs with "popular but unheard" ones to avoid such risky assumptions, which also keeps the quantities of positive and negative samples at the same level in training. Second, the overwhelmingly larger amount of hit songs leads to neglect of the less popular ones [4, 30]. Our work takes care of both by modeling their relation through their locations in the song embedding space. Third, most applications that model user preference focus on specific tasks and are hard to generalize to others. Our work, in contrast, provides a general user preference embedding model which may apply to multiple downstream tasks. Finally, while most AMG tasks mainly focus on musicality, we seek to integrate UMP into the generation process as well.

3 METHODOLOGY

3.1 Problem Formulation

This paper aims to model UMP and apply it to personalized AMG. The problem can be formulated as follows. First, we seek to define UMP p_u: Given the listening records of the public, $\mathcal{H}(U)$, a specific user $u \in U$ and his or her ratings of a group of songs $\mathcal{H}(u) = \{s_1: r_1, s_2: r_2, ..., s_n: r_n\}$ where $s_i \in S$ and $r_i \in \mathbb{Z}^+$, our UMP embedding model $\mathcal{M}$ specified by its trainable parameters θ projects the songs to an embedding space based on the ratings. The UMP is a representative embedding vector $p_u \in P$ aggregated from the song embeddings.

$$p_u = \mathcal{M}_\theta\big(u, \mathcal{H}(u) \,\big|\, \mathcal{H}(U)\big) \tag{1}$$

Next, we seek to find a function $\mathcal{D}(s, p_u)$: Given any song s and a user's preference embedding p_u, $\mathcal{D}(s, p_u)$ computes and outputs their distance regarding four musical characters. Afterwards, our personalized AMG model $\mathcal{G}$ takes an old song s_{old} and the distance deduced by $\mathcal{D}$. Then the AMG model updates four distance parameters (which represent a variant of s_{old}) to control the generation of a new song s_{new} that is closer to the user's preference embedding vector in the embedding space, namely, $s_{\text{new}} = \mathcal{G}(s_{\text{old}}, \mathcal{D}(s_{\text{old}}, p_u))$. The user's rating (denoted as $\mathcal{R}$) for the new song should be higher than for the old one. Therefore, our objective in personalized AMG is formulated as follows.

$$\arg\max_\theta \mathcal{R}\Big(u, \mathcal{G}\Big(s, \mathcal{D}(s, p_u)\Big)\Big) \tag{2}$$

3.2 UMP Embedding Model

We design a hierarchical UMP embedding model with one convolutional recurrent neural network (CRNN) block at the bottom to learn representative embeddings of musical features and one CRNN-SA block piled on top to further condense the learned musical feature embeddings to song embeddings. The model is optimized with a contrastive learning loss which is subtly designed to differentiate users' distinct preferences. As illustrated in Figure 2, the model first extracts musical features from songs' elementary audio features with the first CRNN block and then extracts song embeddings from the combined features with the second CRNN-SA block. The song embeddings represent the song's UMP embedding vector. In the resultant song embedding space, the contrastive loss makes songs' embeddings closer to one another if they are liked by the same users and farther away otherwise. Then the UMP can be represented as the weighted aggregation of a user's preferred songs' embeddings.

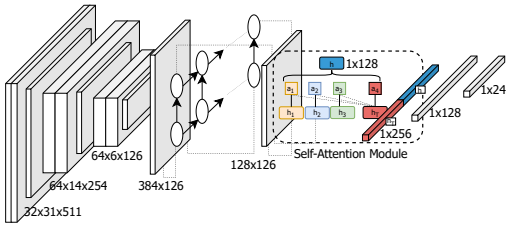

Figure 3: The Network Structure of the CRNN(-SA) Block.

As shown by Figure 3, we employ the same CRNN structure for both parts of the hierarchical model – the musical feature extractor and the UMP extractor while the latter has an extra Self-Attention (-SA) module (Figure 2-b). The LSTM and CNN have 2 and 3 layers respectively. The number of CNN kernels are 32, 64, and 64, with the size of 3×3, stride 1×1, and a 2×2 sized AvgPool kernel. The activation function is the *tanh*, and the dropout rate is 0.1. The hidden dimension in the LSTM is 128. The input is the stacked preliminary audio features $f_a \in \mathbb{R}^{L \times T}$ of size (1024×64), the musical feature extractor outputs the musical features $f_m \in \mathbb{R}^K$ of a specified dimension 12, and the UMP output $p_u / p_s \in \mathbb{R}^K$ is the embedding vector of a specified dimension 24. We train the model with a train/valid/test ratio of 8:1:1, Adam [27] as the optimizer, a batch size of 32/192, a learning rate of 1e-4/2e-4 for the musical feature extractor and UMP extractor respectively.

3.2.1 Audio and Musical Features Exaction. To concisely represent songs and allow the UMP embedding model to learn UMP effectively, we take the songs and extract audio features known to greatly impact auditory perception. We utilize audio features rather than symbolic representations because the number of available songs in MIDI format annotated with listening histories is insufficient to train an effective UMP model.

We retrieve basic audio features, including the Mel-Spectrogram, Mel-frequency cepstral coefficients (MFCCs), and Chromatograms (Chroma) from song clips of 30-60 seconds, with a window size of 2048, hop length of 1024, and sample rate of 22050Hz.

In addition, we also employ 12 high-level musical features provided by Spotify's Web API [1], which are found to be representative and effective for describing music preference in previous works [18, 39, 43]. We divide the 12 musical features into the following four sub-groups and train in four independent trials. The first group, { ***danceability, energy, liveness, valence, loudness*** }, are subjective descriptions relevant to characteristics such as rhythm, beat, speed, regularity, and stability. The second group, { ***speechness, acousticness, instrumentalness*** }, are all related to human voice detection. The last two groups, { ***tempo, time signature*** } and { ***key, mode*** }, are objective and measurable music features.

For songs whose musical features are not directly available from the Spotify Web API, we predict the first two groups of features from the basic audio features with the musical feature extraction model (Figure 2-a). The final L1 losses are 0.086 and 0.070 respectively. And we use Librosa's API and Krumhansl-Schmuckler key-finding algorithm [54] to retrieve the other two groups.

After computing the musical features of a song, we normalize the feature values V to the scale of [0, 1] by min-max scaling to avoid imbalances among scales of their values.

3.2.2 Hierarchical CRNN-SA Based UMP Embedding Model. After training the musical feature extraction model (Figure 2-a), we create a CRNN-SA block to further extract the song embedding (Figure 2-b). The CRNN-SA block takes both the audio features and the extracted musical features to avoid possible information loss in deep models, as inspired by residual networks [52]. The CRNN and CRNN-SA blocks together form a hierarchical CRNN-SA UMP embedding model to project songs to corresponding points in the UMP embedding space.

Specifically, the CRNN-SA block is a self-attention-enhanced version of CRNN structure [56], which seeks to highlight parts of the output sequence of the LSTM that are more important. The attention weights (denoted as a_t) indicate the significance of the output sequence items (denoted as h_t) at each timestamp t, proportional to its similarity (denoted as e_t) to the last output in the sequence (denoted as h_T). The attention-weighted summation of all h_t is concatenated with h_T and fed to linear layers to get the final output embedding p_s.

$$a_t = \frac{exp(h_t \cdot h_T)}{\sum_{i=1}^{T} exp(h_i \cdot h_T)}; \quad p_s = \tanh(W(\sum_{t=1}^{T} a_t h_t + h_T) + b) \quad (3)$$

Because users have different focusing points when listening to songs, their music preferences are likely to be strongly related to specific parts of a piece. Therefore, with the self-attention layer, the model can concentrate on these special parts and give more accurate preference embeddings.

[1] https://developer.spotify.com/console/get-audio-features-track/

3.3 Training Procedures

To capture both public and individual music preferences, we optimize the hierarchical UMP embedding model with contrastive learning in two phases, as shown in Figure 1-a. First, we pre-train it with listening histories that record the number of times users had heard each song. This allows us to project the public trends and the other songs according to their interaction forces defined under a contrastive loss. Second, we fine-tune an exclusive copy of the model for each new user based on the user's ratings of a group of songs. As shown by Figure 1-b & c, if we apply the user's unique UMP embedding model to personalized AMG, the generated songs can be rated again and added to the user's training data, thus iteratively improving the performance of the UMP model with a cycle of "rate"-"refine"-"generate". The two phases of training are further detailed in section 4 and 5 correspondingly.

3.4 Contrastive Loss Design

3.4.1 General Loss Terms of Preference. Aiming to optimize the UMP embedding model so that users' unique music tastes can be captured and differentiated from each other and also the public trends, we define three UMP-relevant attributes reflecting the relationship between users and songs. They then fuse into the contrastive loss function, which defines the interaction forces among songs' and users' embeddings in the UMP hyperspace.

Popularity. Popularity decides the probability that a song s lies in the public trends. It can be expressed as the summation of all users' attention degrees to the song, as shown by Formula (4) where the attention is the number of times that user u has listened to song s, denoted as $\mathcal{T}(u, s)$.

$$\mathrm{Pop}(s) = \sum_{u \in U} \mathcal{T}(u, s) \quad (4)$$

We set a threshold th to separate out a group of most popular songs that have been played for more than $th_{\mathrm{ListeningTimes}}$ times to represent the public trends.

$$\mathrm{IsPop}(s) = \mathrm{Pop}(s) > th_{\mathrm{ListeningTimes}} \quad (5)$$

Likability. Likability is a binary variable indicating whether a user likes a song. It will be used in the contrastive loss as the label of a sample song, determining whether the song belongs to the positive or negative sample groups. When pre-training the UMP model, the value is true if the song has been listened to by the user, and false if the song belongs to the most popular songs (top 1024) but has not been listened to by the user. During the fine-tuning phase, the value is true if the user rates the song above his or her average rating (denoted as $\overline{\mathcal{R}_u}$) where $\mathcal{R}$ denotes the rating.

$$\mathrm{Like}^{\mathrm{pt}}(s, u) = \begin{cases} 1, & \text{if } \mathcal{T}(u, s) > 0 \\ 0, & \text{if } \mathcal{T}(u, s) = 0 \wedge \mathrm{IsPop}(s) \\ \varnothing, & \text{otherwise} \end{cases} \quad (6)$$

$$\text{Like}^{\text{ft}}(s,u) = \begin{cases} 1 & \text{if } \mathcal{R}(u,s) > \overline{\mathcal{R}_u} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Significance. $\text{Sig}(u,s)$ describes to what degree the user favors the song. It equals $\mathcal{T}$ during pre-training and $\mathcal{R}$ during fine-tuning.

$$\text{Sig}^{\text{pt}}(u,s) = \mathcal{T}(u,s); \qquad \text{Sig}^{\text{ft}}(u,s) = \mathcal{R}(u,s) \tag{8}$$

The individual UMP embedding can now be defined as the average of the song embeddings weighted by their significance to the user, where p_u and p_s denote the UMP and song embedding respectively.

$$p_u = \sum_{s \in \mathcal{H}(u)} \text{Sig}(u,s) p_s \tag{9}$$

Moreover, we note the generalizability of the presented functions, and their applicability to other domains with slight modifications. For example, in online shopping advertisements, the advertisers may choose to link popularity to monthly sales, number of clicks, or other factors, while likability refers to whether the item's tags fall within the buyer's interests.

3.4.2 Contrastive Loss for UMP Embedding Model. Based on the loss terms defined above, we design a contrastive loss function to allow songs that the same user likes to attract each other in the embedding space while pushing other songs away. As shown in Formula (10), we use the cosine similarity of two songs' embeddings a and b to indicate their distance.

$$\text{Sim}(a,b) = \cos(a,b) = \frac{a \cdot b}{\|a\|\|b\|} = \frac{\sum_{i=1}^{n} a_i \times b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \times \sqrt{\sum_{i=1}^{n} b_i^2}} \tag{10}$$

For computational convenience, we first apply Euclidean normalization to scale the vectors to the unit length so that the similarity can be computed directly as the vectors' dot product.

During pre-training, samples of each batch are selected according to the listening history of a specific user, with an equal number of the following three types of songs: 1) n non-most-popular songs liked by the current user, 2) n most popular songs that the user does not like, and 3) n non-most-popular songs not liked by the current user but liked by another user. We stack the UMP embeddings of these three groups of songs into three matrices, denoted as Mat^+, Mat_1^-, and Mat_2^-, where the first group serves as positive samples and the rest serve as negative samples (Figure 1-a). The inclusion criteria of these three types of samples can be formulated as the condition expression (11). We expect large distances between Mat^+ and the rest, but small distances within Mat^+ and Mat^- themselves. Therefore, the loss function can be defined as Formula (12). Computationally, the three matrices are normalized and multiplied pair-wise following Formula (10). The contrastive loss is defined as the weighted summation of their similarities.

$$\begin{aligned} & i \neq j \wedge \neg\text{IsPop}(\text{Mat}^+) \wedge \text{Like}^{\text{pt}}(\text{Mat}^+, u_i) \\ & \wedge \text{IsPop}(\text{Mat}_1^-) \wedge \neg\text{Like}^{\text{pt}}(\text{Mat}_1^-, u_i) \\ & \wedge \neg\text{IsPop}(\text{Mat}_2^-) \wedge \neg\text{Like}^{\text{pt}}(\text{Mat}_2^-, u_i) \wedge \text{Like}^{\text{pt}}(\text{Mat}_2^-, u_j) \end{aligned} \tag{11}$$

$$\begin{aligned} \mathcal{L}^{pt} = & \text{Sim}(\text{Mat}^+, \text{Mat}_1^-) + \text{Sim}(\text{Mat}^+, \text{Mat}_2^-) - \text{Sim}(\text{Mat}^+, \text{Mat}^+) \\ & - \text{Sim}(\text{Mat}_1^-, \text{Mat}_1^-) + \textit{margin} \end{aligned} \tag{12}$$

where *margin* is a large number to ensure that the loss value is positive at all times during optimization.

During fine-tuning, we sample from a user' rated songs and use the user's average rating to split them into positive and negative samples. The loss function can thus be defined as Formula (13).

$$\begin{aligned} \mathcal{L}^{ft} = & \text{Sim}(\text{Mat}^+, \text{Mat}^+) + \text{Sim}(\text{Mat}^-, \text{Mat}^-) - \text{Sim}(\text{Mat}^+, \text{Mat}^-) \\ & + \textit{margin} \end{aligned} \tag{13}$$

3.5 Embedding Distance Model

We propose an embedding distance model to link up the UMP embedding model and personalized AMG by computing the cross-modal distance between a user's UMP embedding and the input seed song's embedding. First, we raise the concept of music distance hyperspace where any two songs can be regarded as a variant of each other specified by four music distance parameters. Second, the four distance parameters are appended to a song's audio and musical features in the UMP embedding model (Figure 2-c) so that the output song embedding is influenced by the four parameters. Hence, the computed distance between a user's UMP embedding and a seed song's embedding can instruct the music generator and adapt it to the UMP.

Music Distance Hyperspace. Following a controllable AMG framework [8], we treat each existing song s_i as a seed song. For any seed song, we create a corresponding distance hyperspace that originates from the song, with four music distance parameters as the four axes. Each parameter describes the distance between the seed song and one of its variants regarding one of four musical characters including α (chord progression distance), β (melody's statistical distance), γ (melodic rhythmic distance), and λ (melody's contour distance). Then we can consider any other song as a variant of the seed song which corresponds to a specific point in that seed song's hyperspace. Namely, any song can be denoted by a 5D vector {seed song, α, β, γ, λ}.

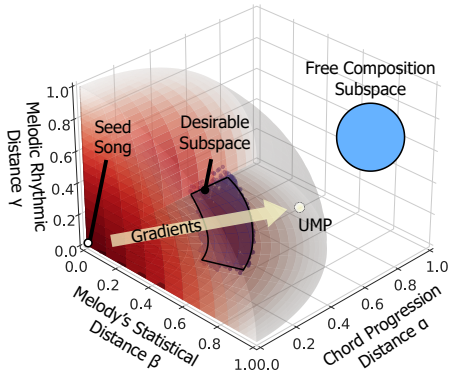

Figure 4: The Music Distance Hyperspace. The origin point represents the seed song. Each axis describes its distance to a variant song regarding one musical character. Only 3 out of 4 axes are present for visualization purposes.

As illustrated in Figure 4, these four distance parameters are all in the range of [0, 1]. The origin point represents the seed song itself (i.e., a variant that is identical to the seed song), while points

near the location [1, 1, 1, 1] represents songs (variants) with weak or no relation to the seed song (i.e., free composition subspace).

Theoretically, a UMP embedding corresponds to a point in a seed song's music distance hyperspace because a UMP embedding is essentially the aggregation of several song embeddings. For example, if a user likes song s_i, his or her UMP is likely to correspond to a certain point that is close to the origin in the hyperspace of s_i.

Embedding Distance Computing. Although there is not a formulable mapping to convert a UMP to exact coordinates in the hyperspace, we propose to quantitatively compare the difference between a user's UMP embedding with an input seed song's embedding and then output a distance value, which indicates how close the seed song is to the UMP with respect to their cosine similarity.

To connect the UMP with the music distance hyperspace, we append the four distance parameters $\{\alpha, \beta, \gamma, \lambda\}$ of an input song to its concatenated audio/musical features in our proposed hierarchical CRNN-SA model (Figure 2-c). Thus, the impacts of the four musical characters on the UMP can be jointly learned through both original songs (which have the parameters [0, 0, 0, 0]) and song variants (which have the corresponding four distance parameters they are generated from).

3.6 UMP-Aware Music Generation

As shown in Figure 4, we assume that any seed song has a "desirable subspace" in its hyperspace, which falls between the origin and the corresponding point of the UMP (as the imaginary UMP point in the figure). The points in the desirable subspace represent certain special variants that 1) accord well with a user's UMP, 2) originate from the seed song, and 3) are sufficiently different from the seed song to be claimed as a new piece. UMP-aware AMG is to search for such a desirable subspace in a seed song's hyperspace according to the gradients of the four parameters deduced from the distance value (as shown by the yellow arrow in Figure 4). Therefore, personalized AMG is converted to sampling a point in the desirable subspace corresponding to a variant that approaches the UMP. We finally propose the strategy of how to deduce the updated four distance parameters according to the distance value $\mathcal{D}(s, p_u)$ computed above and use the updated parameters to generate a new song $\mathcal{G}\Big(s, \mathcal{D}\big(s, p_u\big)\Big)$ that is closer to the given UMP.

$$\mathcal{M}\Big(u, \mathcal{G}\Big(s, \mathcal{D}\big(s, p_u\big)\Big)\Big|\mathcal{H}(U)\Big) \rightarrow \mathcal{M}\big(u, \mathcal{H}(u)\,\big|\,\mathcal{H}(U)\big) \tag{14}$$

As illustrated in Figure 1-c, we apply style transfer techniques in our strategy: for an input seed song s, the UMP embedding model computes its embedding $p_s = \mathcal{M}\big(u, s\,\big|\,\mathcal{H}(U)\big)$. Because the seed song is not likely to perfectly align with the UMP, the distance value between these two vectors is computed as:

$$\mathcal{L}^{\text{bp}} = -S_{p_u,p_s} = -\cos(p_u, p_s) \tag{15}$$

Instead of backpropagating the distance value (loss) to update the network parameters, we freeze the parameters of the UMP networks and backpropagate the loss to the four distance parameters of the input seed song. The four parameters are then updated according to the gradients, formulated as follows,

$$\arg\max_{\alpha,\beta,\gamma,\lambda} \mathcal{L}^{\text{bp}} \Leftrightarrow \{\alpha, \beta, \gamma, \lambda\} \mathrel{-}= gd * sl \tag{16}$$

where gd denotes gradients and sl denotes a hyperparameter step length, deciding to what extent the user expects the generated song to approach the UMP.

The seed song with the updated four distance parameters is the input of the controllable music generator to generate a new song. This music generator takes in a seed song in MIDI format and four controllable distance parameters as inputs. Then it jointly transforms hard-coded pop music composition rules, the seed song, and the four parameters into a predicting probability of next notes. Finally, it selectively picks notes and concatenates them into a piece of song. The details of the controllable music generation process are presented in [8]. To our knowledge, this system is the first of its kind that achieves the goal of personalized AMG.

4 OBJECTIVE EVALUATION

We conduct objective experiments to test our UMP embedding model. The results show that it can capture UMP and differentiate users' unique tastes from public trends.

4.1 Datasets

Our primary dataset is the Echo Nest Taste Profile Subset [2] (here denoted "the Echo dataset"), which contains 48,373,586 listening events formatted in triplets of { user, song, times }, representing the number of times that a user has listened to a song. There are 1,019,318 unique users and 384,546 unique songs in total. The majority of users (> 70%) have listened to at least 40 songs, and the majority of songs (> 70%) have been listened to at least 10 times. We further align 303,487 of the songs with their MP3 clips of 30-60 seconds in [58], originally collected from 7digital[3]. The audio files are padded to the same length before retrieving the preliminary audio features. Among songs with available MP3 files, 251,859 songs have available musical features from Spotify Web API, which we take as the training set for our musical feature extractor, as mentioned in section 3.2.1. After training, it then predicts the musical features of the rest songs.

4.2 Compared Methods

Since there are few previous studies on UMP modeling, we compare several different architectures as ablated models.

- **UMP-MLP**: a basic MLP is used instead of the proposed upper CRNN-SA block.
- **UMP-L'**: a degraded loss function is used which does not have the inter-user term $\text{Sim}(\text{Mat}^{+}, \text{Mat}_2^{-})$.
- **UMP-audio**: only preliminary audio features are employed.
- **UMP-musical**: only the extracted musical features are used.
- **UMP-w/oSA**: the self-attention layer is removed.

4.3 Objective Measures

To evaluate the performance of the UMP models, we compare their validation losses $\mathcal{L}^{pt}$, which serve as measures to evaluate the models' ability to embed songs. We also propose the following three objective measures to test how well a UMP model can project

[2] http://millionsongdataset.com/tasteprofile/
[3] https://sg.7digital.com/

similar UMPs to a cluster while keeping the clusters distinct from the public trends as well as other UMP clusters.

1) *Positioning* refers to the average dot product similarity between the public music trend and two typical groups of users' music preferences: the most representative U_c and the most unique U_u users. We rank the users by the proportion of the most popular 1024 songs in their listening histories and select the top and bottom 0.05% of U as U_c and U_u respectively; we then compute the public trends *pop* as the average of the most popular songs' embeddings. An effective UMP model should output sharply contrasting positioning values for the two user groups. 2) *Centrality* and 3) *Dispersity* refer to the average dot product similarity among the UMP embeddings of the most alike users U_{like} and the most unlike users U_{unlike}, respectively. We rank the permutations of user pairs by the overlap in their listening histories, then select the two groups from the top and bottom 0.05% user pairs. An effective UMP model should yield a large value for Centrality and a small value for Dispersity. These three objective measures are computed by Formula (17). We deliberately define the two objective measures at the user level rather than song level so that they are distinct from the loss terms in training, to avoid the fallacy of circular justification.

$$\begin{aligned} \text{Positioning} &= \{\frac{1}{|U_c|}(U_c \cdot pop),\ \frac{1}{|U_u|}(U_u \cdot pop)\} \\ \text{Centrality} &= \frac{1}{|U_{\text{like}}|^2}\sum u_i \cdot u_j,\ u_i, u_j \in U_{\text{like}} \\ \text{Dispersity} &= \frac{1}{|U_{\text{unlike}}|^2}\sum u_i \cdot u_j,\ u_i, u_j \in U_{\text{unlike}} \end{aligned} \tag{17}$$

4.4 Results and Visualization

The results of the defined objective measures, along with the validation loss, are listed in Table 1. Our proposed UMP embedding model $\mathcal{M}$ outperforms the other baselines in all three objective measures, indicating the advantages of 1) CRNN-SA over basic MLP in extracting complex musical features and preferences, 2) the proposed contrastive loss function in differentiating different users, and 3) the usage of fused audio and musical features over individual ones. (Note that UMP-L' uses a different loss function, so its $\mathcal{L}^{pt}$ value is not applicable for comparison.)

Table 1: Results of three objective measures and validation loss. (P: Positioning; C: Centrality; D: Dispersity)

Model	$P(U_c)$	$P(U_u)$	C	D	$\mathcal{L}^{pt}$	*Params*
UMP-MLP	20.30	-2.41	17.34	0.55	0.7720	67.4M
UMP-L'	8.47	-13.97	5.40	5.12	-	0.5M
UMP-audio	16.63	-9.21	13.41	-0.23	0.6093	0.5M
UMP-musical	26.38	-0.30	22.63	2.43	0.8054	0.01M
UMP-w/oSA	23.46	-31.62	24.41	4.75	0.6029	0.5M
$\mathcal{M}$	**27.73**	**-14.82**	**27.02**	**-0.43**	**0.5870**	**0.5M**

We additionally visualize the distributions of song and user embeddings. We first project a group of songs' embeddings computed by our proposed model to a 2D surface by applying the T-distributed Stochastic Neighbor Embedding (t-SNE) [26]. Figure 5-a shows the dimension-reduced embeddings of the popular songs and 50 users' liked songs (64 songs per user). The popular songs are gathered

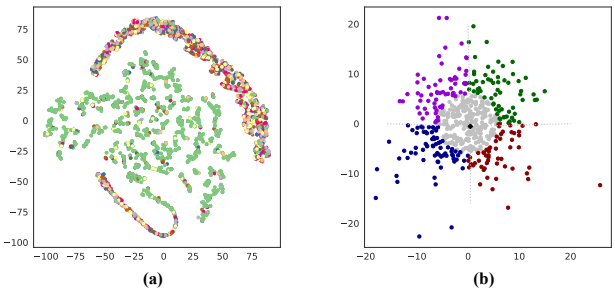

Figure 5: t-SNE Visualization of (a) the Embedding Results of Songs, (b) the Embedding Results of Users. The number of users is used as a parameter to specify the number of clusters, which serves to verify whether the preferences of different users can be distinguished.

in the center of the space (in green), and the rest of the songs liked by individual users are mapped to surrounding areas (in other colors). This "planet-satellite" shaped distribution aligns with our expectation of an effective UMP model that separates unique music preferences from the public trends.

And in Figure 5-b we visualize 500 UMP embeddings computed by our proposed UMP model in the 2D space where each sample point refers to the aggregation of a user's listening histories, as defined in Formula (9). In the plot, the majority of users are distinguishable and form a cluster in the center that represents the public trends, with sporadic distributions around the periphery representing other users' unique music tastes. The result also illustrates the potential of our UMP embedding model to categorize users into clusters based on the zones and directions where they fall in. Users in the same cluster may have similar music tastes, which could be applicable to downstream tasks like music recommendation, user classification, and so on.

5 SUBJECTIVE EVALUATION

We conduct subjective evaluations with 46 participants (university students) to verify the effectiveness of our system in UMP-aware AMG. Results show that our system is capable of generating songs that users prefer and continuously improving the lower-bound performance of music-related services.

5.1 Experiment Procedure

The experiment is an iteration of *listening - rating - model refinement - music generation*, repeated by 6 sessions (S_1-S_6) that span 16 days. We selected the 30 seed songs based on an initial user survey to cover the diverse music preferences of different people. In S_1, participants listen to and rate the 30 seed songs on a ten-point scale, and also browse a list of the most popular songs (as previously defined in the term Likability) and mark the songs they like. Each participant's choices update the general UMP model from pre-training, creating their own preliminary UMP models $\mathcal{M}_1$. Then the participants give ratings (r_1^A) for the 3 new songs, generated based on $\mathcal{M}_1$ using their 3 lowest-rated (r_1^{lowest}) songs as seed

songs. The following sessions are 3 days after the previous one. In S_i, $i \in [2, 6]$, the participants first listen again to the 3 songs that were generated in S_{i-1} and give them "refreshed" ratings r_i^B that update their UMP models immediately to M_i. They then rate 3 new songs that are generated based on their updated UMP models using the 3 current lowest-rated (r_i^{lowest}) songs as seed songs. The ratings of the 3 new songs, r_i^A, update their UMP models again. Finally, in S_6, besides the usual 6 songs (3 repeated and 3 new), the participants again listen to the 3 songs generated in S_1 for the third time and give ratings r_6^C.

5.2 Subjective Measures

To quantify the system's performance, we define the following three subjective measurements. 1) *m1: UMP Stability* tests the consistency of UMP over short (3 days) and relatively long (16 days) periods, computed as $\frac{1}{3}(r_i^B - r_{(i-1)}^A)$ and $\frac{1}{3}(r_6^C - r_1^A)$ respectively. With the assumption of stable UMPs, AMG can be iteratively improved by fine-tuning the generation model with new data instead of re-training new models from scratch. 2) *m2: Instant Improvement* measures the immediate increment of the users' rating of a generated song compared to the input seed song, computed as $r_i^A - r_i^{lowest}$. 3) *m3: Continuous Improvement* evaluates improvements in the system's low-bound performance. For each seed song, we consider the highest rating among all its generated variants to be its "representative performance" and the lowest values among all songs' representative performance to be the system's lower-bound performance. A continuous improvement of this measure shows the system's ability to upgrade service quality iteratively along with users' usage.

5.3 Results

The results show that our system achieved satisfactory instant and continuous performance improvements, as presented in Figure 6. The values are averaged over all 46 valid participants after filtering out careless entries through randomly inserted fake samples. Firstly, the blue line with x markers shows the average change per song in users' ratings of the same 3 songs after 3 days. After 16 days, the average change of users' ratings of the 3 generated songs of S_1 is 0.73 (with a standard deviation of 1.15). The small values of the UMP Stability verify the assumption that UMPs stay fixed over

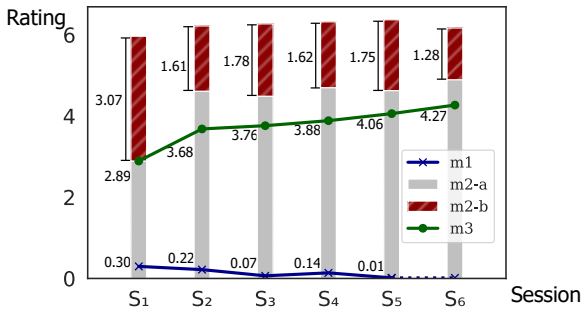

Figure 6: Results of Measures in Subjective Evaluation (m1: UMP Stability; m2-a: Input Songs' Ratings; m2-b: Generated Songs' Ratings; m3: Continuous Improvement)

short and relatively long periods and thus indicates the potential of the proposed system to iteratively improve the performance without starting over. Secondly, the stacked bars present the *m2* at each session, where the average rating increase is in red. Thirdly, the green line with circle markers traces *m3* of the system's lower-bound performance. The positive values of *m2* and an increasing trend in *m3* substantiate the system's capability to capture UMPs, push variants of seed songs towards UMPs instantly, and keep improving the guaranteed basic service quality iteratively. A series of demos[4] can illustrate how the four parameters control music generation and how the generated pieces fit user preferences.

6 DISCUSSION AND FUTURE WORK

Though our proposed UMP model can be simply connected to a deep learning-based music generator, we chose a rule- and statistics-based model as the music generator, which is highly interpretable and controllable with meaningful parameters. This allowed us to thoroughly study how UMP influences AMG. After profoundly understanding the UMP's impacts on AMG, we will be more confident to further explore the integration of UMPs in deep learning based AMG in the future.

By attaching the proposed UMP embedding model to music-related applications, we can compute non-specialists' UMPs via their interactions with our system without the requirements of musical knowledge, which is important for our intended clinical application. We seek to develop an AMG system which could generate patients' preferred music to motivate them to adhere to Rhythmic Auditory Stimulation [16], an evidence-based neurologic music therapy which is clinically proven to improve the gait performance, mobility and quality of life of Parkinson's patients. We will conduct a clinical experiment with Parkinson's patients to evaluate the usability and efficacy of our proposed technology upon the ethics approval.

Our UMP model can be easily generalized to other user preference related tasks by re-defining the proposed preference-relevant functions (i.e., popularity and likability threshold, negative sampling strategy). More functions can be added to enhance the model's power. For example, user profiling can also be incorporated if users' metadata is available.

7 CONCLUSION

User preference modeling can facilitate music-driven services, allowing accurate music classification, recommendation, and personalized AMG. This work effectively models UMP from users' listening histories and ratings, and then applies it to UMP-aware AMG. In both objective and subjective experiments, our proposed hierarchical CRNN-SA model with contrastive learning shows satisfactory performance in UMP modeling and personalized AMG, as well as the potential to improve music-related services' qualities continuously.

ACKNOWLEDGMENTS

This project is funded in part by the NUS Cross-Faculty Research Grant R-252-000-B21-133.

[4]https://soundcloud.com/acmmm2022-submission-demo/sets/demos-of-acmmm22-paper-content

REFERENCES

[1] Mohammad Akbari and Jie Liang. 2018. Semi-recurrent CNN-based VAE-GAN for sequential data generation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2321–2325.

[2] Manuel Alfonseca, Manuel Cebrián Ramos, and Alfonso Ortega. 2006. A fitness function for computer-generated music using genetic algorithms. *WSEAS Transactions on Information Science and Applications* (2006).

[3] Christopher Anderson, Arne Eigenfeldt, and Philippe Pasquier. 2013. The generative electronic dance music algorithmic system (GEDMAS). In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, Vol. 9. 5–8.

[4] Christine Bauer and Markus Schedl. 2019. Global and country-specific mainstreaminess measures: Definitions, analysis, and usage for improving personalized music recommendation systems. *PloS one* 14, 6 (2019), e0217389.

[5] Filippo Carnovalini and Antonio Rodà. 2020. Computational creativity and music generation systems: An introduction to the state of the art. *Frontiers in Artificial Intelligence* 3 (2020), 14.

[6] Chuanming Chen, Shuanggui Zhang, Qingying Yu, Zitong Ye, Zhen Ye, and Fan Hu. 2021. Personalized travel route recommendation algorithm based on improved genetic algorithm. *Journal of Intelligent & Fuzzy Systems* 40, 3 (2021), 4407–4423.

[7] Yen-Liang Chen, Yi-Hsin Yeh, and Man-Rong Ma. 2021. A movie recommendation method based on users' positive and negative profiles. *Information Processing & Management* 58, 3 (2021), 102531.

[8] Shuqi Dai, Xichu Ma, Ye Wang, and Roger B Dannenberg. 2021. Personalized popular music generation using imitation and structure. *arXiv preprint arXiv:2105.04709* (2021).

[9] Ervin Dervishaj and Paolo Cremonesi. 2022. GAN-based matrix factorization for recommender systems. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*. 1373–1381.

[10] Shangzhe Di, Zeren Jiang, Si Liu, Zhaokai Wang, Leyan Zhu, Zexin He, Hongming Liu, and Shuicheng Yan. 2021. Video Background Music Generation with Controllable Music Transformer. In *Proceedings of the 29th ACM International Conference on Multimedia*. 2037–2045.

[11] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. 2018. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[12] Martin Dostál. 2005. Genetic Algorithms As a Model of Musical Creativity–on Generating of a Human-Like Rhythmic Accompaniment. *Computing and Informatics* 24, 3 (2005), 321–340.

[13] Tomislav Duricic, Dominik Kowald, Markus Schedl, and Elisabeth Lex. 2021. My friends also prefer diverse music: homophily and link prediction with user preferences for mainstream, novelty, and diversity in music. In *Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. 447–454.

[14] Douglas Eck and Juergen Schmidhuber. 2002. A first look at music composition using lstm recurrent neural networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale* 103 (2002), 48.

[15] Min Gao, Junwei Zhang, Junliang Yu, Jundong Li, Junhao Wen, and Qingyu Xiong. 2021. Recommender systems based on generative adversarial networks: A problem-driven perspective. *Information Sciences* 546 (2021), 1166–1185.

[16] Jeffrey M Hausdorff, Justine Lowenthal, Talia Herman, Leor Gruendlinger, Chava Peretz, and Nir Giladi. 2007. Rhythmic auditory stimulation modulates gait variability in Parkinson's disease. *European Journal of Neuroscience* 26, 8 (2007), 2369–2375.

[17] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 549–558.

[18] Claire Howlin and Brendan Rooney. 2021. Patients choose music with high energy, danceability, and lyrics in analgesic music listening interventions. *Psychology of Music* 49, 4 (2021), 931–944.

[19] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE international conference on data mining*. Ieee, 263–272.

[20] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, and Douglas Eck. 2018. An improved relative self-attention mechanism for transformer with application to music generation. (2018).

[21] Yongjie Huang, Xiaofeng Huang, and Qiakai Cai. 2018. Music Generation Based on Convolution-LSTM. *Comput. Inf. Sci.* 11, 3 (2018), 50–56.

[22] Miao Jiang, Ziyi Yang, and Chen Zhao. 2017. What to play next? A RNN-based music recommendation system. In *2017 51st Asilomar Conference on Signals, Systems, and Computers*. IEEE, 356–358.

[23] Maximos Kaliakatsos-Papakostas and Emilios Cambouropoulos. 2014. Probabilistic harmonization with fixed intermediate chord constraints.. In *ICMC*.

[24] Vibhor Kant and Kamal K Bharadwaj. 2012. Enhancing recommendation quality of content-based filtering through collaborative predictions and fuzzy similarity measures. *Procedia engineering* 38 (2012), 939–944.

[25] Robert M Keller and David R Morrison. 2007. A grammatical approach to automatic improvisation. In *Proceedings of the Sound and Music Computing Conference*. Citeseer, 330–337.

[26] Tejas Khot. 2016. Visualizing High-Dimensional Data. *XRDS* 23, 2 (dec 2016), 66–67. https://doi.org/10.1145/3021604

[27] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[28] Alexis Kirke and Eduardo Reck Miranda. 2009. A survey of computer systems for expressive music performance. *ACM Computing Surveys (CSUR)* 42, 1 (2009), 1–41.

[29] Dominik Kowald, Elisabeth Lex, and Markus Schedl. 2020. Utilizing human memory processes to model genre preferences for personalized music recommendations. *arXiv preprint arXiv:2003.10699* (2020).

[30] Dominik Kowald, Markus Schedl, and Elisabeth Lex. 2020. The unfairness of popularity bias in music recommendation: A reproducibility study. In *European conference on information retrieval*. Springer, 35–42.

[31] Daniel J Levitin. 2006. *This is your brain on music: The science of a human obsession*. Penguin.

[32] Chien-Hung Liu and Chuan-Kang Ting. 2012. Polyphonic accompaniment using genetic algorithm with music theory. In *2012 IEEE Congress on Evolutionary Computation*. IEEE, 1–7.

[33] Yunshan Ma, Yujuan Ding, Xun Yang, Lizi Liao, Wai Keung Wong, and Tat-Seng Chua. 2020. Knowledge enhanced neural fashion trend forecasting. In *Proceedings of the 2020 International Conference on Multimedia Retrieval*. 82–90.

[34] Yunshan Ma, Lizi Liao, and Tat-Seng Chua. 2019. Automatic fashion knowledge extraction from social media. In *Proceedings of the 27th ACM International Conference on Multimedia*. 2223–2224.

[35] Dimos Makris, Maximos Kaliakatsos-Papakostas, Ioannis Karydis, and Katia Lida Kermanidis. 2017. Combining LSTM and feed forward neural networks for conditional rhythm composition. In *International conference on engineering applications of neural networks*. Springer, 570–582.

[36] Manuel Marques, V Oliveira, S Vieira, and AC Rosa. 2000. Music composition using genetic evolutionary algorithms. In *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512)*, Vol. 1. IEEE, 714–719.

[37] Matt McVicar, Satoru Fukayama, and Masataka Goto. 2014. AutoLeadGuitar: Automatic generation of guitar solo phrases in the tablature space. In *2014 12th International Conference on Signal Processing (ICSP)*. IEEE, 599–604.

[38] Alessandro B Melchiorre and Markus Schedl. 2020. Personality correlates of music audio preferences for modelling music listeners. In *Proceedings of the 28th ACM conference on user modeling, adaptation and personalization*. 313–317.

[39] Martijn Millecamp, Nyi Nyi Htun, Yucheng Jin, and Katrien Verbert. 2018. Controlling spotify recommendations: effects of personal characteristics on music recommender user interfaces. In *Proceedings of the 26th Conference on user modeling, adaptation and personalization*. 101–109.

[40] Achmad Arif Munaji and Andi Wahju Rahardjo Emanuel. 2021. Restaurant Recommendation System Based on User Ratings with Collaborative Filtering. In *IOP Conference Series: Materials Science and Engineering*, Vol. 1077. IOP Publishing, 012026.

[41] Senthilselvan Natarajan, Subramaniyaswamy Vairavasundaram, Sivaramakrishnan Natarajan, and Amir H Gandomi. 2020. Resolving data sparsity and cold start problem in collaborative filtering recommender system using linked open data. *Expert Systems with Applications* 149 (2020), 113248.

[42] Maria Navarro, Juan Manuel Corchado, and Yves Demazeau. 2016. MUSIC-MAS: Modeling a harmonic composition system with virtual organizations to assist novice composers. *Expert Systems with Applications* 57 (2016), 345–355.

[43] Rutger Nijkamp. 2018. *Prediction of product success: explaining song popularity by audio features from Spotify data*. B.S. thesis. University of Twente.

[44] Sergio Oramas, Oriol Nieto, Mohamed Sordo, and Xavier Serra. 2017. A deep multimodal approach for cold-start music recommendation. In *Proceedings of the 2nd workshop on deep learning for recommender systems*. 32–37.

[45] Ender Özcan and Türker Erçal. 2007. A genetic algorithm for generating improvised music. In *International Conference on Artificial Evolution (Evolution Artificielle)*. Springer, 266–277.

[46] François Pachet and Pierre Roy. 2011. Markov constraints: steerable generation of Markov sequences. *Constraints* 16, 2 (2011), 148–172.

[47] Martin Pichl, Eva Zangerle, and Günther Specht. 2016. Understanding playlist creation on music streaming platforms. In *2016 IEEE International Symposium on Multimedia (ISM)*. IEEE, 475–480.

[48] Peter J Rentfrow, Lewis R Goldberg, and Daniel J Levitin. 2011. The structure of musical preferences: a five-factor model. *Journal of personality and social psychology* 100, 6 (2011), 1139.

[49] Francesco Sanna Passino, Lucas Maystre, Dmitrii Moor, Ashton Anderson, and Mounia Lalmas. 2021. Where To Next? A Dynamic Model of User Preferences. In *Proceedings of the Web Conference 2021*. 3210–3220.

[50] Ian Simon, Dan Morris, and Sumit Basu. 2008. MySong: automatic accompaniment generation for vocal melodies. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 725–734.

[51] Bob L Sturm, Joao Felipe Santos, Oded Ben-Tal, and Iryna Korshunova. 2016. Music transcription modelling and composition using deep learning. *arXiv preprint arXiv:1604.08723* (2016).

[52] Ying Tai, Jian Yang, and Xiaoming Liu. 2017. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3147–3155.

[53] Senem Tanberk and Dilek Bilgin Tükel. 2021. Style-Specific Turkish Pop Music Composition with CNN and LSTM Network. In *2021 IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 000181–000185.

[54] David Temperley. 1999. What's key for key? The Krumhansl-Schmuckler key-finding algorithm reconsidered. *Music Perception* 17, 1 (1999), 65–100.

[55] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. *Advances in neural information processing systems* 26 (2013).

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[57] Isaac Wallis, Todd Ingalls, Ellen Campana, and Janel Goodman. 2011. A rule-based generative music system controlled by desired valence and arousal. In *Proceedings of 8th international sound and music computing conference (SMC)*. 156–157.

[58] Xinxi Wang and Ye Wang. 2014. Improving content-based and hybrid music recommendation using deep learning. In *Proceedings of the 22nd ACM international conference on Multimedia*. 627–636.

[59] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. 2017. MidiNet: A convolutional generative adversarial network for symbolic-domain music generation. *arXiv preprint arXiv:1703.10847* (2017).

[60] Eva Zangerle, Martin Pichl, and Markus Schedl. 2020. User models for culture-aware music recommendation: fusing acoustic and cultural cues. *Transactions of the International Society for Music Information Retrieval* 3, 1 (2020).

[61] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendations with negative feedback via pairwise deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1040–1048.

[62] Lixin Zou, Long Xia, Yulong Gu, Xiangyu Zhao, Weidong Liu, Jimmy Xiangji Huang, and Dawei Yin. 2020. Neural interactive collaborative filtering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 749–758.