ANALYSIS OF SONG/ARTIST LATENT FEATURES AND ITS APPLICATION FOR SONG SEARCH

Kosetsu Tsukuda **Masataka Goto**
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{k.tsukuda, m.goto}@aist.go.jp

ABSTRACT

For recommending songs to a user, one effective approach is to represent artists and songs with latent vectors and predict the user's preference toward the songs. Although the latent vectors represent the characteristics of artists and songs well, they have typically been used only for computing the preference score. In this paper, we discuss how we can leverage these vectors for realizing applications that enable users to search for songs from new perspectives. To this end, by embedding song/artist vectors into the same feature space, we first propose two concepts of artist-song relationships: overall similarity and prominent affinity. Overall similarity is the degree to which the characteristics of a song are similar overall to the characteristics of the artist; while prominent affinity is the degree to which a song prominently represents the characteristics of the artist. By using Last.fm play logs for two years, we analyze the characteristics of the concepts. Moreover, based on the analysis results, we propose three applications for song search. Through case studies, we demonstrate that our proposed applications are beneficial for searching for songs according to the users' various search intents.

1. INTRODUCTION

Embedding songs into a feature space is beneficial for realizing various applications for music information retrieval (MIR). For example, by using tags of each song, songs can be embedded into a tag-based feature space [1]; this enables a user to search for similar songs of her favorite song. In another example, by embedding songs into the Arousal-Valence space [2] according to their audio features [3], a user can search for the song with the highest Arousal value from her favorite artist's songs. The same can be said for artists; by embedding artists into a feature space based on the topics of lyrics, artists similar to a user's favorite artist in terms of topic similarity can be retrieved [4].

Embedding heterogeneous data into a feature space is also useful for MIR tasks: song recommendations for playlists by embedding tags and songs [5, 6], computing tag similarity by embedding artists and tags [7], etc.

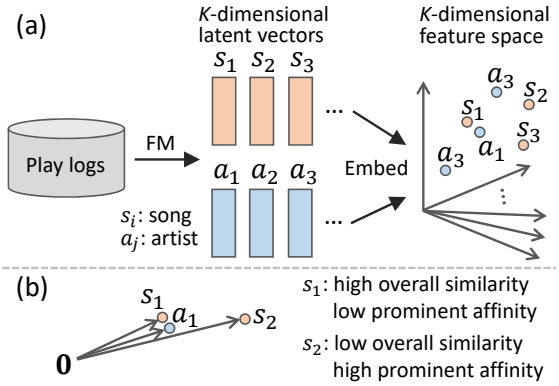

Figure 1. Overview of our proposed ideas: (a) embedding artists' latent vectors and songs' latent vectors into the same feature space and (b) concepts of overall similarity and prominent affinity.

In spite of the potential to embed heterogeneous data, there have been few studies that have embedded songs and artists [8]. If both songs and artists are embedded into the same feature space, greater variety of MIR applications can be realized. For example, given an artist, we can search for songs that have similar characteristics to those of the artist (i.e., songs whose feature vectors are close to the feature vector of the artist).

To realize this, we use Factorization Machines (FM) [9], which is an item recommendation technique. By using FM, each song, user, and artist can be represented by K-dimensional latent vectors. Although such vectors are usually used to compute a user's preference toward a song, we leverage the latent vectors of songs and artists to embed songs and artists into the same feature space (Fig. 1 (a)).

In the embedded feature space, we propose two concepts of artist-song relationships: overall similarity and prominent affinity. Suppose the vector's direction of artist a_1 in Fig. 1 (b) represents the sadness. The length of the vector indicates the degree of sadness. In this case, song s_1 has almost the same degree of sadness and s_1 is similar overall to a_1. On the other hand, song s_2 has a higher degree of sadness. This means that s_2 prominently represents a_1's characteristics and has a higher prominent affinity with a_1 in terms of sadness. We can recommend s_1 and s_2 to a user as a_1's characteristic songs because of the high overall similarity and the high prominent affinity, respectively. Such a recommendation would be useful when the user listens to one of a_1's songs for the first time because she can decide whether to listen to a's other songs after listening to a characteristic song of a_1. By using these concepts, we can realize not only such song recommendations

but also various applications for MIR, as in Section 5.

Our main contributions in this paper are as follows.

- We propose the concepts of overall similarity and prominent affinity between an artist and a song by leveraging their latent vectors learned through FM (Section 3).
- By using Last.fm play logs for two years, we show various characteristics of overall similarity and prominent affinity. For example, we reveal that an artist's popular songs tend to have high prominent affinity with the artist (Section 4).
- We demonstrate that the concepts of overall similarity and prominent affinity can be used to realize various applications for MIR. Specifically, we show three applications: familiarity-oriented search, typicality-oriented search, and analogy search (Section 5).

2. RELATED WORK

2.1 Matrix Factorization for Song Recommendations

Matrix Factorization (MF) [10] has been widely used for item recommendations. In the context of song recommendations, too, the effectiveness of MF has been reported [11–14]. One of the characteristics of MF for song recommendations is that each user and song are represented by K-dimensional latent vectors. This enables the model to learn a user's latent preference toward songs. Although MF typically considers interactions between users and songs, Factorization Machines (FM) [9] can include side information in addition to information about users and songs. Side information can be an artist, category of a song, and even the weather when a user listens to a song. Because of such flexibility, FM has also been used for song recommendations [15–18]. The idea of using artist information in FM for song recommendations has already been proposed [15, 16]; in this case, each user, song, and artist are represented by K-dimensional latent vectors.

Although such latent vectors in FM represent the characteristics of users, songs, and artists well, they have typically been used for computing a user's preference score toward a song and generating a personalized ranked list of songs for each user. Different from existing studies, we analyze latent vectors of songs and artists based on new relationships between artists and songs (overall similarity and prominent affinity) and show their potential to implement MIR applications.

2.2 Heterogeneous Embedding for MIR

The usefulness of embedding heterogeneous data into a feature space has been reported in various MIR tasks, where data has been embedded by a Markov-model-based method [5, 19], co-occurrence-based method [20], etc. For example, by embedding tags and songs, song recommendations for playlists have been realized [5, 6]. Other examples include computing tag similarity by embedding artists and tags [7], retrieving songs by words by embedding songs and words in playlist titles [20], and visualizing the time-dependent listening preferences of a population by embedding users and songs [19]. Our study is different from theirs in that we embed artists and songs into a feature space.

The study closest to ours is that by Weston *et al.* [8]. They proposed a method for embedding artists and songs into a feature space and solved tasks such as predicting songs for a given artist and retrieving similar songs for a given song. To embed songs, their method requires audio data for all songs. In contrast, we use users' play logs. Although comparing the embedding accuracy of these two approaches is beyond the scope of this paper, our approach using play logs has an advantage over their approach in terms of the applicability of insights into the MIR community because various kinds of large play logs are easily accessible [21–27] compared to the accessibility of large audio data.

2.3 MIR Applications for Song Search

In the MIR community, various kinds of applications for song search have been proposed. These applications have enabled users to more easily find their desired songs and search for songs from a new perspective. For example, query by humming [28–34] and query by singing [35–38] enable users to search for songs even when they do not know the song title. Similarity-based song search is also beneficial for searching for new songs that are similar to a user's favorite song, where similarity between songs is measured by low-level acoustic features [39, 40], voice timbres of vocals [41], tags [1], and a combination of these characteristics [42]. When a user has a specific search intent, searching for songs using metadata [43,44] and words in lyrics [45] is also helpful to find her desired songs.

In this paper, we propose two concepts of artist-song relationships. These concepts can be used to search for an artist's characteristic songs, as we will show in Section 4.3. In addition, in Section 5, we demonstrate application examples that can be realized by leveraging these concepts and latent vectors of artists and songs. Our proposed applications are also helpful for searching for users' desired songs and new songs. We believe our study is beneficial for other researchers to implement MIR applications based on our proposed concepts.

3. OVERALL SIMILARITY AND PROMINENT AFFINITY

In this section, we first describe how to generate latent vectors of artists and songs through FM. We then propose the concepts of overall similarity and prominent affinity.

3.1 Notation

Let $\mathcal{U}$, $\mathcal{I}$, and $\mathcal{A}$ denote the sets of users, songs, and artists, respectively. $\mathcal{I}_u^+$ represents the set of songs preferred by user $u \in \mathcal{U}$. Following Lim *et al.* [46], we define the songs played $\geq \mu$ times by u as the preferred songs of u. By using the data, we first aim to accurately generate a per-

sonalized ranked list of songs for each user u from $\mathcal{I} \setminus \mathcal{I}_u^+$, which is a set of songs not included in u's preferred songs.

3.2 FM for Song Recommendation

FM [9] is a method for predicting a user's preference toward an item based on MF [10]. A typical MF deals with only interactions between users and items; while in FM, side information (e.g., the artist or category of a song) can also be included in the model. By considering artist information, we can extract new relationships between artists and songs, as we will describe in Section 3.3. Below, we describe FM considering artist information.

In FM, user u, song s, and artist a have K-dimensional latent vectors $\boldsymbol{\nu_u}$, $\boldsymbol{\nu_s}$, and $\boldsymbol{\nu_a}$, respectively. The preference score of user u toward song s based on the second-order estimator of FM is computed with the following model:

$$\widehat{x}_{us} = \alpha+\beta_u+\beta_s+\beta_{a_s}+\langle \boldsymbol{\nu_u}, \boldsymbol{\nu_s} \rangle+\langle \boldsymbol{\nu_u}, \boldsymbol{\nu_{a_s}} \rangle+\langle \boldsymbol{\nu_{a_s}}, \boldsymbol{\nu_s} \rangle, \quad (1)$$

where α is the global offset, a_s is the artist of s, and β_u, β_s, and β_{a_s} are the user/song/artist bias terms. As can be seen in the model, the preference score is computed by a user's affinity with a song ($\langle \boldsymbol{\nu_u}, \boldsymbol{\nu_s} \rangle$), user's affinity with an artist ($\langle \boldsymbol{\nu_u}, \boldsymbol{\nu_{a_s}} \rangle$), and artist's affinity with a song ($\langle \boldsymbol{\nu_{a_s}}, \boldsymbol{\nu_s} \rangle$). Regarding the last term $\langle \boldsymbol{\nu_{a_s}}, \boldsymbol{\nu_s} \rangle$, if artist a_s tends to sing calm songs and song s is also calm, the value of $\langle \boldsymbol{\nu_{a_s}}, \boldsymbol{\nu_s} \rangle$ becomes high; while if s is exciting music, the value becomes low. Note that because a song's popularity is reflected in β_s (i.e., more popular songs tend to have higher values of β_s), popular songs do not always have high values of $\langle \boldsymbol{\nu_{a_s}}, \boldsymbol{\nu_s} \rangle$. Rather, the value of $\langle \boldsymbol{\nu_{a_s}}, \boldsymbol{\nu_s} \rangle$ is determined purely by the affinity between a and s.

Note that although we use *simple* FM just by adding artist information, our goal in this paper is not to improve recommendation accuracy. Rather, we aim to study how to leverage latent vectors of songs and artists. Although this simple FM can learn reliable latent vectors because it can achieve high enough recommendation accuracy, as we will report in Section 4.2, we can adopt more sophisticated FM (e.g., FM considering song co-occurrence [47] and audio information [11]); we leave this for future work.

We adopt Bayesian Personalized Ranking (BPR) [48] to learn latent vectors. BPR is a pairwise ranking optimization framework and is designed to deal with users' implicit consumption behaviors such as playing a song rather than explicit ones such as rating. In BPR, the training set $\mathcal{D}$ used for optimizing parameters is defined as follows:

$$\mathcal{D} = \{(u, i, j) \mid u \in \mathcal{U} \wedge i \in \mathcal{I}_u^+ \wedge j \in \mathcal{I} \setminus \mathcal{I}_u^+\}.$$

That is, a triad (u, i, j) means that user u prefers song i to song j. The optimization criterion for $\mathcal{D}$ is given by:

$$\sum_{(u,i,j)\in\mathcal{D}} \ln \sigma(\widehat{x}_{uij}) - \lambda_\Theta \|\Theta\|^2, \quad (2)$$

where σ is the sigmoid function, $\Theta = \{\beta_s, \beta_a, \boldsymbol{\nu_u}, \boldsymbol{\nu_s}, \boldsymbol{\nu_a}\}$ represents all model parameters, and λ_Θ is a regularization hyperparameter. $\widehat{x}_{uij}$ represents the difference between u's preference for i and that for j, which is defined as $\widehat{x}_{uij} = \widehat{x}_{ui} - \widehat{x}_{uj}$. Finally, we learn the parameters by using TensorFlow [49] with Adam optimizer [50].

3.3 Overall Similarity and Prominent Affinity

The learned model parameters are usually used for computing a user's preference score toward a song by using Eq. 1. In this paper, we propose using the learned model parameters (i.e., latent vectors) $\boldsymbol{\nu_a}$ and $\boldsymbol{\nu_s}$ for capturing the relationships between artists and their songs. In Eq. 1, because the affinity between $\boldsymbol{\nu_a}$ and $\boldsymbol{\nu_s}$ is considered, the nth ($1 \leq n \leq K$) dimension of $\boldsymbol{\nu_a}$ and that of $\boldsymbol{\nu_s}$ have the same meaning. For example, if the first dimension of $\boldsymbol{\nu_a}$ represents the calmness, the first dimension of $\boldsymbol{\nu_s}$ also represents the calmness of the song. Hence, we can embed $\boldsymbol{\nu_a}$ and $\boldsymbol{\nu_s}$ into the same feature space, as shown in Fig. 1 (a). In the feature space, the angle and length of a vector represents its qualitative and quantitative aspects, respectively. Specifically, the difference of the angle between two vectors represents the difference of their characteristics as a song or artist because each dimension represents a characteristic of songs and artists. While the length of a vector represents the degree of its characteristics: if the value of $\boldsymbol{\nu_s}$'s nth dimension is large, $\boldsymbol{\nu_s}$ represents nth characteristic well. By using the embedded feature space, we propose two concepts of the relationships between artists and songs: overall similarity and prominent affinity.

3.3.1 Overall Similarity

Suppose artist a_1's song s_1 is mapped fairly close to a_1 in the feature space, as shown in Fig. 1 (b). In this case, we can say that s_1 is similar overall to a_1. Thus, we define the closeness between an artist and a song in the feature space as the overall similarity between them. More formally, given artist a and song s, the overall similarity between a and s is computed based on the Euclidean distance between them: $f_{os}(a, s) = \frac{1}{\|\boldsymbol{\nu_a} - \boldsymbol{\nu_s}\|+1}$.

The concept of overall similarity can be used to search for an artist's song that represents the artist's characteristics well. Searching for such a song and listening to it is useful to quickly understand the artist's characteristic songs. In particular, when a user listens to an artist's song for the first time, it might be helpful for her to listen to the artist's characteristic song and then decide if she wants to listen to the artist's other songs.

3.3.2 Prominent Affinity

Now suppose a_1's song s_2 is mapped fairly close to the extended position of a_1 as shown in Fig. 1 (b). This means that s_2 prominently represents a_1's characteristics because the degree of each characteristic of s_2 is higher than that of a_1. Thus, we define the inner product of an artist vector and a song vector as the prominent affinity between them. More formally, given artist a and a's song s, the prominent affinity between a and s is given by $f_{pa}(a, s) = \langle \boldsymbol{\nu_a}, \boldsymbol{\nu_s} \rangle$.

The concept of prominent affinity can be used to search for an artist's song that strongly represents the artist's characteristics. Similar to the overall similarity, searching for such a song would be helpful when a user listens to an artist's song for the first time.

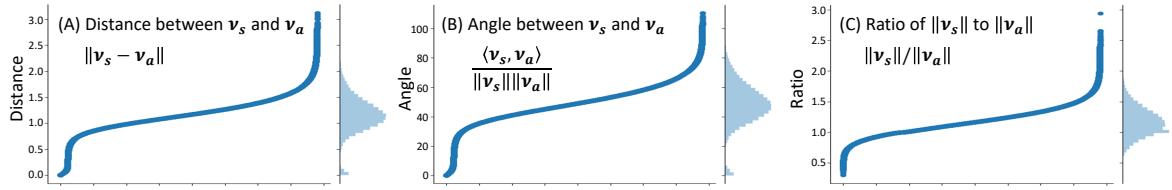

Figure 2. Distributions of (A) distance between ν_s and ν_a, (B) angle between ν_s and ν_a, and (C) ratio of $\|\nu_s\|$ to $\|\nu_a\|$. Each dot represents a song where songs are sorted in ascending order of the y-values in each graph.

Number of users ($\|\mathcal{U}\|$)	84,708
Number of songs ($\|\mathcal{I}\|$)	390,158
Number of artists ($\|\mathcal{A}\|$)	32,448
Sum of preferred songs ($\sum_{u \in \mathcal{U}} \|\mathcal{I}_u^+\|$)	18,478,304

Table 1. Dataset statistics.

4. ANALYSIS

In this section, we analyze the characteristics of overall similarity (OS) and prominent affinity (PA).

4.1 Dataset

We use the LFM-1b dataset that includes music listening logs on Last.fm [21]. Each listening log consists of user ID, song ID, artist ID, and timestamp. The dataset also includes data for converting song ID and artist ID to song name and artist name, respectively. We use logs for two years (between 1/1/2012 and 12/31/2013). The μ in Section 3.1 is set to five, where μ is the threshold to determine that song s is included in $\mathcal{I}_u^+$ when u listens to s equal to or more than μ times. To increase the reliability of learned model parameters, songs and artists that have been listened to by less than 10 different users and users who have listened to less than 10 different songs are discarded. Table 1 shows the dataset statistics after the preprocessing.

4.2 Model Development

For each user, we split $\mathcal{I}_u^+$ into training/validation/test sets. To this end, we randomly select one preferred song (i.e., $i \in \mathcal{I}_u^+$) for validation $\mathcal{V}_u$ and another for testing $\mathcal{T}_u$ [51]. All the remaining songs are used for training $\mathcal{R}_u$. The recommendation performance is evaluated by the AUC (Area Under the ROC Curve), which is widely used to evaluate whether model parameters are appropriately learned [52, 53]: $AUC = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{|\mathcal{D}_u|} \sum_{(i,j) \in \mathcal{D}_u} \delta\left(\widehat{x}_{ui} > \widehat{x}_{uj}\right)$, where $\mathcal{D}_u = \{(i,j) \mid i \in \mathcal{T}_u \wedge j \in \mathcal{I} \setminus \mathcal{I}_u^+\}$, and $\delta(z)$ is 1 when z is true and 0 otherwise. The AUC value ranges between 0 and 1, and a higher value represents better performance (i.e., the model parameters are appropriately learned). The hyperparameters (i.e., λ_Θ in Eq. 2 and the learning rate) are tuned on a validation set in terms of the AUC, where the hyperparameters are selected from {0.0001, 0.001, 0.01, 0.1, 1}. We set the latent dimensionality K to 50. [1] We emphasize that our goal here is not to evaluate the recommendation accuracy of FM by comparing with other methods. Rather, we aim to evaluate whether the parameters in FM developed by our dataset are appropriately learned by showing that the AUC is close to 1 after the learning process.

[1] We evaluated the AUC on the validation dataset by changing K from 10 to 100 in increments of 10. The AUC saturated when $K = 50$.

Rank	OS	PA
1	I'm So Tired	Something
2	Get Back	All You Need Is Love
3	The End	Come Together
4	Sun King	Hey Jude
5	Here Comes the Sun	I Am the Walrus
6	She's Leaving Home	Lucy in the Sky with Diamonds
7	Glass Onion	Eleanor Rigby
8	You Like Me Too Much	A Hard Day's Night
9	You Never Give Me Your Money	I Want to Hold Your Hand
10	The Night Before	Golden Slumbers

Rank	OS	PA
1	Minor Thing	Californication
2	Police Station	Scar Tissue
3	If	Dani California
4	Can't Stop	Snow (Hey Oh)
5	Fortune Faded	By the Way
6	Annie Wants a Baby	The Adventures of Rain Dance Maggie
7	Happiness Loves Company	The Zephyr Song
8	Brendan's Death Song	Brendan's Death Song
9	Give It Away	Torture Me
10	Emit Remmus	Dosed

Table 2. Ranking results of songs in terms of OS and PA (top: "The Beatles," bottom: "Red Hot Chili Peppers").

The AUC on the test set achieved a high value: 0.973. This result means that the model parameters and the embedded feature space are appropriately learned and the artist-song relations based on OS and PA are reliable. We also computed the Spearman rank correlation between the values of β_s and song popularities to evaluate if β_s correctly reflects song popularity, as mentioned in Section 3.2. The popularity of song s was measured by the number of different users who have played s. The popularity ranking of songs are obtained by sorting them in descending order of their popularities. For each artist, we compute the correlation between the popularities of the artist's all songs and their values of β_s. The average of the correlations over all artists was relatively high: 0.502. When we define the popularity of artist a as the number of different users who have played at least one of a's songs, the average of the correlations became high with increasing artist popularity: artists whose popularities are $\geq$ 100 and $\geq$ 300 had correlation values of 0.682 and 0.755 on average, respectively. From these results, we can say that the bias term β_s certainly reflects the song's popularity, and the value of $\langle \nu_{a_s}, \nu_s \rangle$ in Eq. 1 is determined mainly by the affinity between a_s and s especially for popular artists. Hereafter, the parameter values computed in this section are used for artist latent vectors and song latent vectors.

4.3 Analysis Results

Although we showed that the model parameters are appropriately learned, if most of an artist's songs are mapped

Rank	Area (a)	Area (b)	Area (c)	Area (d)
1	Gold on the Ceiling	Have Love Will Travel	Yearnin'	Can't Find My Mind
2	Lonely Boy	Same Old Thing	Keep Your Hands Off Her	Her Eyes Are A Blue Million Miles
3	Tighten Up	I Got Mine	Grown So Ugly	Howlin For You
4	Little Black Submarines	Run Right Back	Till I Get My Way	The Wicked Messenger
5	Everlasting Light	Your Touch	Just Got To Be	The Baddest Man Alive

Table 3. Ranking results of familiarity-oriented search for "The Black Keys" in terms of PA.

very close to the artist, it would be useless to rank songs according to the subtle difference of their positions. To confirm whether artists' songs are well distributed, we evaluate the distributions of (A) distance between $\boldsymbol{\nu_s}$ and $\boldsymbol{\nu_a}$, (B) angle between $\boldsymbol{\nu_s}$ and $\boldsymbol{\nu_a}$, and (C) ratio of $\|\boldsymbol{\nu_s}\|$ to $\|\boldsymbol{\nu_a}\|$. Fig. 2 shows the results. In all cases, the value distribution is close to the Gaussian distribution. These results indicate that songs are well distributed in the feature space and it is meaningful to rank songs according to OS, which is affected by (A), and PA, which is affected by (B) and (C).

Next, we analyze the ranked results of songs by OS/PA. Table 2 shows the top 10 songs of "The Beatles" and "Red Hot Chili Peppers." Since the top ranked songs are largely different between the two concepts, it would be meaningful to generate two ranked lists so that we can show one of them (or both of them) to a user according to her intent. We can also see that the top ranked songs in PA tend to be more popular than those in OS. To evaluate this, we compute the Spearman rank correlation between OS-based/PA-based song ranking and popularity-based song ranking. As expected, the correlation of PA-based ranking is high (0.577), while that of OS-based ranking is low (-0.147). As mentioned in Section 4.2, the effect of song popularity is eliminated by the bias terms β_s. Therefore, this high correlation of PA-based ranking is caused purely by the characteristics of the songs: songs strongly reflecting the artist's characteristics tend to be popular.

5. APPLICATIONS

In Section 4.3, we showed how to directly use the concepts of OS and PA to rank an artist's songs. In this section, by leveraging the concepts and learned parameters of $\boldsymbol{\nu_a}$ and $\boldsymbol{\nu_s}$, we demonstrate three applications: familiarity-oriented search, typicality-oriented search, and analogy search. Through these demonstrations, we show that this paper brings reusable insights in that our proposed concepts can be used for various kinds of MIR applications.

5.1 Familiarity-oriented Search

By considering PA and song popularity, an application to search for songs according to a user's familiarity with an artist can be realized. As we mentioned in Section 4.3, songs with a high PA tend to be popular, but some of them are unpopular. Similarly, some popular songs have a low PA. Hence, according to the degree of PA and the degree of popularity, an artist's songs can be classified into four areas, as shown in Fig. 3. Searching for songs in area (a) would be beneficial especially for a user who listens to the artist's song for the first time because these songs are popular and represent the artist's typical characteristics well; she can decide if she wants to listen to the artist's other

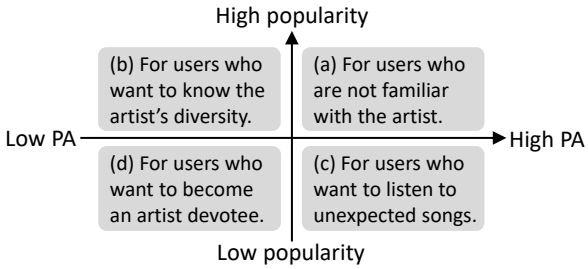

Figure 3. Properties of songs classified by the degree of PA and the degree of popularity.

songs by listening to the searched songs. After listening to such songs, searching for songs in area (b) would be useful to let her understand the diversity of the artist because area (b) includes songs that are popular but different from the artist's typical characteristics. Moreover, searching for songs in area (c) enables her to listen to unexpected songs in terms of the fact that the searched songs match the artist's typical characteristics well but are not known by many people. Finally, searching for songs in area (d) would be helpful for her to become an artist devotee by listening to them.

In light of the above, given artist a, we rank all a's songs for each area in terms of PA as follows. For area (a), the score of song s is computed by $r_{ap}(s, \mathcal{I}_a) + r_{pop}(s, \mathcal{I}_a)$, where $\mathcal{I}_a$ is a set of all songs of a and $r_{ap}(s, \mathcal{I}_a)$ and $r_{pop}(s, \mathcal{I}_a)$ represent the ranks of s among $\mathcal{I}_a$ in terms of AP and popularity, respectively. The songs are then ranked in ascending order of score. For area (b), (c), and (d), the score is given by $r_{pop}(s, \mathcal{I}_a) - r_{ap}(s, \mathcal{I}_a)$, $r_{ap}(s, \mathcal{I}_a) - r_{pop}(s, \mathcal{I}_a)$, and $-r_{ap}(s, \mathcal{I}_a) - r_{pop}(s, \mathcal{I}_a)$, respectively. In these areas, the songs are also ranked in ascending order. Regarding OS, songs in each area has the same meaning as with PA and we can also make a song ranking for each area in the same manner as with PA.

Table 3 shows example results of the top five song rankings in each area for "The Black Keys" in terms of PA. It would be beneficial to show each ranking to a user according to her familiarity with "The Black Keys."

5.2 Typicality-oriented Search

Because the parameters of all artists and all of their songs are learned by using the same optimization criterion (Eq. 2), all artists and all songs can be embedded into the same feature space. This means that given artist a, all songs in the dataset can be ranked in terms of OS or PA. In other words, we can search for songs that represent a's typical characteristics well even if they were not a's songs. By showing such songs to a user who is a fan of a, she may be willing to listen to unfamiliar songs because they are highly related to her favorite artist. This is also benefi-

Rank	OS	PA
1	California Gurls / Katy Perry	Circus / Britney Spears
2	Racy Lacey / Girls Aloud	...Baby One More Time / Britney Spears
3	Gimme More / Britney Spears	I Wanna Go / Britney Spears
4	Cannibal / Ke$ha	Hung Up / Madonna
5	Piece of Me / Britney Spears	I Kissed a Girl / Katy Perry

Rank	OS	PA
1	Cymbal Rush / Thom Yorke	Untitled / Interpol
2	Atoms for Peace / Thom Yorke	The Clock / Thom Yorke
3	And It Rained All Night / Thom Yorke	I've Seen It All / Björk
4	Convergence / Jonny Greenwood	Svefn-g-englar / Sigur Rós
5	Quick Canal / Atlas Sound	Kingdom of Rust / Doves

Table 4. Ranking results of typicality-oriented search (top: "Lady Gaga," bottom: "Radiohead").

cial for online music streaming services because they can expand users' interests and let users listen to more songs.

Given artist a, when we generate a ranked list of all songs in $\mathcal{I}$ in terms of OS, we compute the score of each song by $f_{os}(a, s)$, which was defined in Section 3.3.1, and rank all songs in descending order of scores. Similarly, for PA, we use $f_{pa}(a, s)$, which was defined in Section 3.3.2, and rank all songs in descending order of scores.

Table 4 shows the top five song rankings for "Lady Gaga" and "Radiohead." Although we do not use acoustic features and metadata of songs, in the case of "Lady Gaga," all artists in the table are female artists. Similarly, in the case of "Radiohead," members of the band such as "Thom Yorke" and "Jonny Greenwood" are retrieved. In addition, because "Radiohead" is a rock band, rock bands such as "Interpol," "Sigur Rós," and "Doves" are ranked at higher positions. These results show the potential of this application to enable users to find new attractive songs.

5.3 Analogy Search

When a user searches for an object in an unfamiliar domain and obtains the desired search results, it is helpful to give an example object in her familiar domain to the search system. This kind of search is known as analogy search and its usefulness to search for persons [54] and restaurants [55] has been studied. An analogy search for MIR can also be an attractive application as follows. Suppose a user is a big fan of "Eminem" and likes his song "Not Afraid." She has recently become interested in "Lady Gaga." However, because she has little knowledge on "Lady Gaga" and there are too many "Lady Gaga" songs, she is at a loss as to which song she should listen to. In such a case, by using an analogy, she can ask something like "what Lady Gaga song corresponds to Not Afraid by Eminem?" If we can return search results for such a query, it would be helpful for her to try some songs of "Lady Gaga."

In an analogy search for MIR, given a query consisting of source artist a^s, source song s^s, and target artist a^t, our goal is to return a^t's song s^t where the relation between a^t and s^t corresponds to that between a^s and s^s. We compute the relation between an artist and a song based on the angle between their latent vectors and the ratio of their lengths because they respectively represent the qualitative and quantitative aspects of artists/songs as we described in Section 3.3. Intuitively, if the angle between a^s and s^s is similar to that between a^t and s^t, and the ratio of $\|\boldsymbol{\nu}_{\boldsymbol{s}^s}\|$ to $\|\boldsymbol{\nu}_{\boldsymbol{a}^s}\|$ is also similar to that of $\|\boldsymbol{\nu}_{\boldsymbol{s}^t}\|$ to $\|\boldsymbol{\nu}_{\boldsymbol{a}^t}\|$, we regard

Query
Source artist: Madonna
Source song: Like a Prayer
Target artist: Lady Gaga

Rank	Song title
1	Yoü and I
2	Poker Face
3	Telephone

Query
Source artist: Eminem
Source song: Not Afraid
Target artist: Lady Gaga

Rank	Song title
1	The Edge of Glory
2	Bad Romance
3	Yoü and I

Query
Source artist: The Beatles
Source song: That Means A Lot
Target artist: Aerosmith

Rank	Song title
1	Milk Cow Blues
2	Face
3	Temperature

Table 5. Ranking results of analogy search.

s^t as a good analogy search result of the query. However, because the degree of the scatter of songs in the feature space is different from one artist to another, we need to normalize the angles and ratios between an artist and its songs. Formally, given a^s, we first compute the angle between $\boldsymbol{\nu}_{\boldsymbol{a}^s}$ and the vector of each of a^s's songs. The angles are then normalized to fit into the interval $[0, 1]$ by min-max normalization. Let θ_{s^s} denote the normalized angle of s^s. We also compute the ratio of the length of each of a^s's songs to $\boldsymbol{\nu}_{\boldsymbol{a}^s}$'s length (i.e., $\|\boldsymbol{\nu}_{\boldsymbol{s}}\|/\|\boldsymbol{\nu}_{\boldsymbol{a}^s}\|$ where $s \in \mathcal{I}_{a^s}$). Again, min-max normalization is applied and let r_{s^s} be the normalized ratio of s^s. Similarly, we compute the normalized angles and ratios for each of a^t's songs. The score of $s^t \in \mathcal{I}_{a^t}$ for analogy search is then computed as follows:

$$f_{analogy}(a^s, s^s, a^t, s^t) = \gamma|\theta_{s^s} - \theta_{s^t}| + (1-\gamma)|r_{s^s} - r_{s^t}|,$$

where γ is a parameter to determine the weights on angle similarity and length-ratio similarity. Finally, a^t's songs are ranked in ascending order of $f_{analogy}(a^s, s^s, a^t, s^t)$.

In Table 5, we show example results where the top three songs are listed for each query. The value of γ is set to 0.5. In the top table, the source song is "Like a Prayer," which is a signature piece for "Madonna," and the target artist is "Lady Gaga." In this case, the signature pieces for "Lady Gaga" are ranked higher. Even when the source artist is "Eminem," whose music is largely different from that of "Lady Gaga," her signature pieces are retrieved for his signature piece "Not Afraid." When the source song is "That Means A Lot," which has a low PA with "The Beatles," songs with a low PA with "Aerosmith" are retrieved. From these results, we can say that this application can search for the target artist's songs that have a similar relationship between the source artist and the source song.

6. CONCLUSION

This paper analyzed song/artist latent features by embedding them into a same feature space. Based on the analysis results, we suggested three applications for song search. We acknowledge a limitation of this paper in that we did not quantitatively evaluate the search results of those applications. Nonetheless, we believe this study is worthwhile contribution as a first step toward leveraging latent vectors of songs and artists. In future work, we plan to quantitatively evaluate the usefulness of our proposed applications by conducting user studies. We also want other researchers to leverage the concepts of OS/PA and realize useful music information retrieval systems.

7. ACKNOWLEDGMENTS

This work was supported in part by JSPS KAKENHI Grant Number 20K19934 and JST ACCEL Grant Number JPMJAC1602, Japan.

8. REFERENCES

[1] M. Levy and M. Sandler, "A semantic space for music derived from social tags," in *Proceedings of the 8th International Conference on Music Information Retrieval*, ser. ISMIR '07, 2007, pp. 411–416.

[2] J. A. Russell, "A circumplex model of affect," *Journal of personality and social psychology*, vol. 39, no. 6, pp. 1161–1178, 1980.

[3] S. Chapaneri and D. Jayaswal, "Structured prediction of music mood with twin gaussian processes," in *Proceedings of the 7th International Conference on Pattern Recognition and Machine Intelligence*, ser. PReMI '17, 2017, pp. 647–654.

[4] K. Tsukuda, K. Ishida, and M. Goto, "Lyric Jumper: A lyrics-based music exploratory web service by modeling lyrics generative process," in *Proceedings of the 18th International Society for Music Information Retrieval Conference*, ser. ISMIR '17, 2017, pp. 544–551.

[5] J. L. Moore, S. Chen, T. Joachims, and D. Turnbull, "Learning to embed songs and tags for playlist prediction," in *Proceedings of the 13th International Society for Music Information Retrieval Conference*, ser. ISMIR '12, 2012, pp. 349–354.

[6] K. Pugatschewski, T. Köllmer, and A. M. Kruspe, "Playlists from the matrix - combining audio and metadata in semantic embeddings," in *Proceedings of the 17th International Society for Music Information Retrieval Conference*, ser. ISMIR '16, 2016.

[7] G. Karamanolakis, E. Iosif, A. Zlatintsi, A. Pikrakis, and A. Potamianos, "Audio-based distributional representations of meaning using a fusion of feature encodings," in *Proceedings of the 17th Annual Conference of the International Speech Communication Association*, ser. INTERSPEECH '16, 2016, pp. 3658–3662.

[8] J. Weston, S. Bengio, and P. Hamel, "Multi-tasking with joint semantic spaces for large-scale music annotation and retrieval," *Journal of New Music Research*, vol. 40, no. 4, pp. 337–348, 2011.

[9] S. Rendle, "Factorization machines," in *Proceedings of the 2010 IEEE International Conference on Data Mining*, ser. ICDM '10, 2010, pp. 995–1000.

[10] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[11] C.-M. Chen, M.-F. Tsai, J.-Y. Liu, and Y.-H. Yang, "Using emotional context from article for contextual music recommendation," in *Proceedings of the 21st ACM International Conference on Multimedia*, ser. MM '13, 2013, pp. 649–652.

[12] A. Vall, M. Skowron, P. Knees, and M. Schedl, "Improving music recommendations with a weighted factorization of the tagging activity," in *Proceedings of the 16th International Society for Music Information Retrieval Conference*, ser. ISMIR '15, 2015, pp. 65–71.

[13] O. Gouvert, T. Oberlin, and C. Févotte, "Matrix cofactorization for cold-start recommendation," in *Proceedings of the 19th International Society for Music Information Retrieval Conference*, ser. ISMIR '18, 2018, pp. 792–798.

[14] D. Yang, T. Chen, W. Zhang, Q. Lu, and Y. Yu, "Local implicit feedback mining for music recommendation," in *Proceedings of the 6th ACM Conference on Recommender Systems*, ser. RecSys '12, 2012, pp. 91–98.

[15] F. Yuan, G. Guo, J. M. Jose, L. Chen, H. Yu, and W. Zhang, "Boostfm: Boosted factorization machines for top-N feature-based recommendation," in *Proceedings of the 22Nd International Conference on Intelligent User Interfaces*, ser. IUI '17, 2017, pp. 45–54.

[16] ——, "Optimizing factorization machines for top-N context-aware recommendations," in *Proceedings of the 17th International Conference on Web Information Systems Engineering*, ser. WISE '16, 2016, pp. 278–293.

[17] M. Pichl, E. Zangerle, and G. Specht, "Improving context-aware music recommender systems: Beyond the pre-filtering approach," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, ser. ICMR '17, 2017, pp. 201–208.

[18] G. Vigliensoni and I. Fujinaga, "Automatic music recommendation systems: Do demographic, profiling, and contextual features improve their performance?" in *Proceedings of the 17th International Society for Music Information Retrieval Conference*, ser. ISMIR '16, 2016, pp. 94–100.

[19] J. L. Moore, S. Chen, D. Turnbull, and T. Joachims, "Taste over time: The temporal dynamics of user preferences," in *Proceedings of the 14th International Society for Music Information Retrieval Conference*, ser. ISMIR '13, 2013, pp. 401–406.

[20] C. Chung, Y. Chen, and H. H. Chen, "Exploiting playlists for representation of songs and words for text-based music retrieval," in *Proceedings of the 18th International Society for Music Information Retrieval Conference*, ser. ISMIR '17, 2017, pp. 478–485.

[21] M. Schedl, "The LFM-1b dataset for music retrieval and recommendation," in *Proceedings of the 2016*

ACM on International Conference on Multimedia Retrieval, ser. ICMR '16, 2016, pp. 103–110.

[22] G. Vigliensoni and I. Fujinaga, "The music listening histories dataset," in *Proceedings of the 18th International Society for Music Information Retrieval Conference*, ser. ISMIR '17, 2017, pp. 96–102.

[23] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th International Society for Music Information Retrieval Conference*, ser. ISMIR '11, 2011, pp. 591–596.

[24] Y. Labs, "R2 - Yahoo! Music user ratings of songs with artist, album, and genre metainformation, v.1.0," https://webscope.sandbox.yahoo.com/catalog.php?datatype=r.

[25] Ò. Celma, *Music Recommendation and Discovery - The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010.

[26] E. Zangerle, M. Pichl, W. Gassler, and G. Specht, "#nowplaying music dataset: Extracting listening behavior from twitter," in *Proceedings of the 1st International Workshop on Internet-Scale Multimedia Management*, ser. WISMM '14, 2014, pp. 21–26.

[27] D. Hauger, M. Schedl, A. Kosir, and M. Tkalcic, "The million musical tweet dataset - what we can learn from microblogs," in *Proceedings of the 14th International Society for Music Information Retrieval Conference*, ser. ISMIR '13, 2013, pp. 189–194.

[28] E. Pollastri, "An audio front end for query-by-humming systems," in *Proceedings of the 2nd International Symposium on Music Information Retrieval*, ser. ISMIR '01, 2001, pp. 65–72.

[29] T. Sorsa and K. Halonen, "Mobile melody recognition system with voice-only user interface," in *Proceedings of the 3rd International Conference on Music Information Retrieval*, ser. ISMIR '02, 2002, pp. 279–280.

[30] S. Pauws, "CubyHum: A fully operational "query by humming" system," in *Proceedings of the 3rd International Conference on Music Information Retrieval*, ser. ISMIR '02, 2002, pp. 187–196.

[31] A. Ito, S. Heo, M. Suzuki, and S. Makino, "Comparison of features for dp-matching based query-by-humming system," in *Proceedings of the 5th International Conference on Music Information Retrieval*, ser. ISMIR '04, 2004, pp. 297–303.

[32] D. Little, D. Raffensperger, and B. Pardo, "A query by humming system that learns from experience," in *Proceedings of the 8th International Conference on Music Information Retrieval*, ser. ISMIR '07, 2007, pp. 335–338.

[33] P. Ferraro, P. Hanna, L. Imbert, and T. Izard, "Accelerating query-by-humming on GPU," in *Proceedings of the 10th International Society for Music Information Retrieval Conference*, ser. ISMIR '09, 2009, pp. 279–284.

[34] C. de la Bandera, A. M. Barbancho, L. J. Tardón, S. Sammartino, and I. Barbancho, "Humming method for content-based music information retrieval," in *Proceedings of the 12th International Society for Music Information Retrieval Conference*, ser. ISMIR '11, 2011, pp. 49–54.

[35] E. Molina, L. J. Tardón, I. Barbancho, and A. M. Barbancho, "The importance of F0 tracking in query-by-singing-humming," in *Proceedings of the 15th International Society for Music Information Retrieval Conference*, ser. ISMIR '14, 2014, pp. 277–282.

[36] C. Wang, J. R. Jang, and W. Wang, "An improved query by singing/humming system using melody and lyrics information," in *Proceedings of the 11th International Society for Music Information Retrieval Conference*, ser. ISMIR '10, 2010, pp. 45–50.

[37] A. Duda, A. Nürnberger, and S. Stober, "Towards query by singing/humming on audio databases," in *Proceedings of the 8th International Conference on Music Information Retrieval*, ser. ISMIR '07, 2007, pp. 331–334.

[38] J. R. Jang, C. Hsu, and H. Lee, "Continuous HMM and its enhancement for singing/humming query retrieval," in *Proceedings of the 6th International Conference on Music Information Retrieval*, ser. ISMIR '05, 2005, pp. 546–551.

[39] E. Allamanche, J. Herre, O. Hellmuth, T. Kastner, and C. Ertel, "A multiple feature model for musical similarity retrieval," in *Proceedings of the 4th International Conference on Music Information Retrieval*, ser. ISMIR '03, 2003, pp. 217–218.

[40] J. Aucouturier and F. Pachet, "Music similarity measures: What's the use?" in *Proceedings of the 3rd International Conference on Music Information Retrieval*, ser. ISMIR '02, 2002, pp. 157–163.

[41] H. Fujihara and M. Goto, "A music information retrieval system based on singing voice timbre," in *Proceedings of the 8th International Conference on Music Information Retrieval*, ser. ISMIR '07, 2007, pp. 467–470.

[42] F. Vignoli and S. Pauws, "A music retrieval system based on user driven similarity and its evaluation," in *Proceedings of the 6th International Conference on Music Information Retrieval*, ser. ISMIR '05, 2005, pp. 272–279.

[43] D. R. Turnbull, L. Barrington, G. Lanckriet, and M. Yazdani, "Combining audio content and social context for semantic music discovery," in *Proceedings of*

the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, ser. SIGIR '09, 2009, pp. 387–394.

[44] P. Knees, T. Pohle, M. Schedl, and G. Widmer, "A music search engine built upon audio-based and web-based similarity measures," in *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '07, 2007, pp. 447–454.

[45] E. Brochu and N. d. Freitas, ""Name that song!": A probabilistic approach to querying on music and text," in *Proceedings of the 15th International Conference on Neural Information Processing Systems*, ser. NIPS'02, 2002, pp. 1529–1536.

[46] D. Lim, J. McAuley, and G. Lanckriet, "Top-N recommendation with missing implicit feedback," in *Proceedings of the 9th ACM Conference on Recommender Systems*, ser. RecSys '15, 2015, pp. 309–312.

[47] D. Liang, J. Altosaar, L. Charlin, and D. M. Blei, "Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence," in *Proceedings of the 10th ACM Conference on Recommender Systems*, ser. RecSys '16, 2016, pp. 59–66.

[48] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, ser. UAI '09, 2009, pp. 452–461.

[49] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: A system for large-scale machine learning," in *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'16, 2016, pp. 265–283.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations*, ser. ICLR '15, 2015.

[51] K. Tsukuda, S. Fukayama, and M. Goto, "ABCPRec: Adaptively bridging consumer and producer roles for user-generated content recommendation," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR'19, 2019, p. 1197–1200.

[52] W. Pan and L. Chen, "GBPR: Group preference based bayesian personalized ranking for one-class collaborative filtering," in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, ser. IJCAI '13, 2013, pp. 2691–2697.

[53] R. He and J. McAuley, "VBPR: Visual bayesian personalized ranking from implicit feedback," in *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, ser. AAAI'16, 2016, pp. 144–150.

[54] Y. Zhang, A. Jatowt, and K. Tanaka, "Is tofu the cheese of Asia?: Searching for corresponding objects across geographical areas," in *Proceedings of the 26th International Conference on World Wide Web Companion*, ser. WWW '17 Companion, 2017, pp. 1033–1042.

[55] M. P. Kato, H. Ohshima, and K. Tanaka, "Content-based retrieval for heterogeneous domains: Domain adaptation by relative aggregation points," in *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '12, 2012, pp. 811–820.