Received June 27, 2021, accepted July 7, 2021, date of publication July 13, 2021, date of current version July 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3096819

Vocal-Accompaniment Compatibility Estimation Using Self-Supervised and Joint-Embedding Techniques

TAKAYUKI NAKATSUKA[1], (Member, IEEE), KENTO WATANABE[1], YUKI KOYAMA[1], MASAHIRO HAMASAKI[1], MASATAKA GOTO[1], AND SHIGEO MORISHIMA[2], (Member, IEEE)

[1]National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Ibaraki 305-8568, Japan
[2]Waseda University, Shinjuku, Tokyo 169-8050, Japan

Corresponding author: Takayuki Nakatsuka (takayuki.nakatsuka@aist.go.jp)

This work was supported in part by JST ACCEL Grant Number JPMJAC1602, JST CREST Grant Number JPMJCR20D4, and JSPS KAKENHI Grant Number 19H04137, Japan.

ABSTRACT We propose a learning-based method of estimating the compatibility between vocal and accompaniment audio tracks, *i.e.*, how well they go with each other when played simultaneously. This task is challenging because it is difficult to formulate hand-crafted rules or construct a large labeled dataset to perform supervised learning. Our method uses self-supervised and joint-embedding techniques for estimating vocal-accompaniment compatibility. We train vocal and accompaniment encoders to learn a joint-embedding space of vocal and accompaniment tracks, where the embedded feature vectors of a compatible pair of vocal and accompaniment tracks lie close to each other and those of an incompatible pair lie far from each other. To address the lack of large labeled datasets consisting of compatible and incompatible pairs of vocal and accompaniment tracks, we propose generating such a dataset from songs using singing voice separation techniques, with which songs are separated into pairs of vocal and accompaniment tracks, and then original pairs are assumed to be compatible, and other random pairs are not. We achieved this training by constructing a large dataset containing 910,803 songs and evaluated the effectiveness of our method using ranking-based evaluation methods.

INDEX TERMS Vocal-accompaniment compatibility, metric learning, music signal processing, music information retrieval.

I. INTRODUCTION

Compatibility is a concept describing how well two (or more) different items exist together without conflict. In this study, we are interested in the compatibility between vocal and accompaniment audio tracks (*i.e.*, *vocal-accompaniment compatibility*), that is, how well the tracks go with each other when playing them simultaneously.

Achieving the estimation of vocal-accompaniment compatibility has the potential to enhance various *music information retrieval* (MIR) applications. One example is an automatic mashup. A mashup is a blend of two songs created by superimposing the vocal track of one song over the accompaniment track of another, and the challenge is to find an appropriate pair of tracks [1]. If a vocal track and an accompaniment track are compatible, these tracks are likely to be suitable as a mashup. It will also be possible to develop an interactive composition tool that suggests candidates of different yet compatible accompaniment (vocal) tracks retrieved from a dataset once a user inputs a vocal (accompaniment) track. If compatibility estimation can be done in real time, it may improve the quality of an automatic mashup.

The concept of vocal-accompaniment compatibility involves many complex factors. For example, a vocal track and an accompaniment track may be considered compatible when they are chromatically in harmony, follow the same rhythmic patterns, or share similar timbral characteristics [2]–[6]. However, there are many exceptions, and no single aspect can fully define compatibility, which makes it difficult to formulate hand-crafted rules to achieve a general vocal-accompaniment compatibility estimation.

We propose a learning-based method of estimating the vocal-accompaniment compatibility instead of relying on specific rules. The main goal of this study is to estimate a

The associate editor coordinating the review of this manuscript and approving it for publication was Shuihua Wang.

compatibility value for a given pair of vocal and accompaniment tracks. One possible way to achieve this goal is to construct a large dataset for (fully) supervised learning. However, it is impractical to gather a sufficient number of vocal- and accompaniment-only tracks, which are unavailable for most songs on the market, and to manually annotate many possible combinations with compatibility labels.

To address the lack of such large datasets consisting of compatible and incompatible pairs of vocal and accompaniment tracks, our method uses a self-supervised technique with which such a dataset is generated from (polyphonic) songs using singing voice separation techniques [7]–[9] under the assumption that *a vocal track and an accompaniment track that are played simultaneously in songs are compatible*. That is, after songs are automatically separated into pairs of vocal and accompaniment tracks, the original pairs of these tracks in the same song are assumed to be compatible. On the other hand, other random pairs of the vocal track separated from a song and the accompaniment track separated from another song are assumed to be relatively incompatible.

With this self-supervised approach, our method applies a variant of deep metric learning. Unlike typical metric learning methods, which train a single encoder, our method simultaneously trains two encoders (*i.e.*, vocal and accompaniment) to achieve *joint embedding* of vocal and accompaniment tracks into a *shared* vector space. We use deep neural networks (DNNs) as the vocal and accompaniment encoders. Our loss function enables the embedded feature vectors of a compatible pair of vocal and accompaniment tracks to lie close to each other in the shared vector space, and those of a random pair to lie far from each other. Once the training is done, we can estimate the compatibility of a given pair of vocal and accompaniment tracks on the basis of the distance between embedded feature vectors of these tracks (Figure 1).

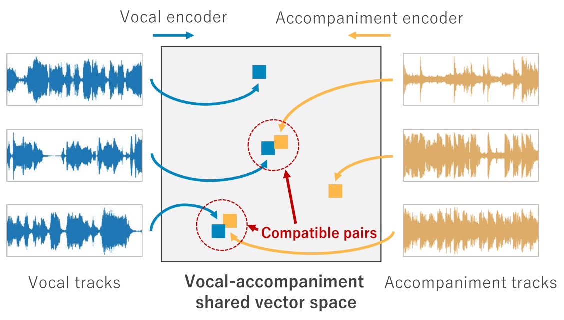


FIGURE 1. Concept of how our method estimates the compatibility between vocal and accompanied tracks. The tracks are jointly embedded in the same vector space (*i.e.*, the vocal-accompaniment shared vector space) using two different encoders (*i.e.*, vocal and accompaniment). These encoders lay the embedded feature vectors of a compatible pair of vocal and accompaniment tracks close to each other and those of an incompatible pair far from each other. Thus, we can estimate the compatibility between vocal and accompaniment tracks by calculating the distance of the embedded feature vectors of these tracks in this shared vector space.

We tested our method by constructing a large dataset called the *trial listening dataset* containing 910,803 songs available online for trial listening. We then quantitatively evaluated the effectiveness of our method using ranking-based evaluation methods, showing that our method outperformed rule-based baseline methods.

II. RELATED WORK

A. COMPATIBILITY ESTIMATION

Compatibility has been extensively discussed in various communities. For example, fashion is a domain that involves compatibility among articles of clothing. With image processing and machine learning techniques, researchers have developed fashion compatibility estimation and recommendation methods [10]–[14]. These methods depend on the availability of large datasets that describe co-occurrence relationships between fashion items (*e.g.*, some of those retrieved from the Amazon web store). Compatibility has also been of interest in computer graphics domains such as 3D indoor scene assembly [15] and color palettes [16]. Our proposed method is inspired by such previous studies [12], [15] but applied to a different domain.

Compatibility estimation has also been discussed in the MIR community and used for MIR applications such as automatic mashup and mixing (*e.g.*, remix and vocal mix). A standard approach to estimating compatibility for the mashup and mixing is to apply hand-crafted rules based on key insights for those applications to tracks. That is, the more strictly the tracks follow the rules, the more compatible they are assumed to be. Lee *et al.* [2] considered the similarity in chromagrams [17] and the harmonic change balance. Bernardes *et al.* [3] proposed a harmony-specific method that combines dissonance-based and perceptual relatedness-based approaches. Davies *et al.* [4] considered the similarity in chromatic harmony, rhythm, and spectral balance. Gebhardt *et al.* [5] used a psychoacoustic model of roughness and pitch commonality to estimate the similarities between music. Maçãs *et al.* [6] used the similarity in harmonic, rhythmic, spectral, and timbral characteristics. These well-crafted rules for mashup and mixing, however, address only certain aspects of the compatibility between vocal and accompaniment tracks.

We take a different approach that estimates the compatibility between vocal and accompaniment tracks by directly learning the compatibility using self-supervised and joint-embedding techniques, instead of taking the rule-based approach.

B. SELF-SUPERVISED AND JOINT-EMBEDDING TECHNIQUES

Self-supervised learning is a form of unsupervised learning in which a piece of data is used as self-supervision and is advantageous in that it does not require large datasets with annotations [18]. To take advantage of this, methods using temporal synchronization between audio and video tracks as the self-supervision have been proposed [19]–[22]. Inspired by these methods, we use a pair of vocal and accompaniment tracks, which are played simultaneously and temporally synchronized in a song, as self-supervision.

Joint-embedding techniques are widely used in metric learning for cross-modal tasks. Because data of different modalities can be treated as identical data in a joint-embedding space and trained under a common metric, deep metric learning and joint-embedding techniques perform well together. In MIR-related tasks, deep metric learning succeeds in learning joint representations over several modalities such as a vocal and mix [23], vocal imitation and sound recording [24], [25], animal sounds [26], sheet music and audio spectrograms [27], music and image [28]–[31], and music and video [21], [22]. The target pair for the metric learning described in this paper consists of a vocal track and an accompaniment track.

III. METHOD AND IMPLEMENTATION DETAILS

This section describes an overview of our learning-based method for vocal-accompaniment compatibility estimation. Because this method requires a large labeled dataset, we construct such a dataset in a self-supervised manner. Note that any song which includes vocal and accompaniment tracks can be used in our method. We used the trial listening dataset, which consists of 910,803 songs available on a music service described in Subsection IV-A, for our experiments. We generate training data from this dataset as follows. Every song in the dataset is separated into a pair of vocal and accompaniment tracks using a singing voice separation technique [9]. Because vocalists are not always singing, vocal tracks obtained from singing voice separation have non-vocal sections where no vocal activity is detected (*e.g.*, introduction, interlude, or instrumental solo). These sections disrupt stable training because most songs include non-vocal sections; thus, a non-vocal section and all possible sections of accompaniment tracks will be compatible, which is an undesired result. Therefore, non-vocal sections should be removed from training data. The problem is that vocal tracks often include artifacts in non-vocal sections after singing voice separation. These artifacts are difficult to remove by simply setting an amplitude threshold. Therefore, we apply a vocal activity detection method [32] to each vocal track to detect non-vocal sections and eliminate these sections from training data. After the removal of non-vocal sections, we convert both vocal and accompaniment tracks into constant-Q transform (CQT) representations [33] to feed them into encoders as images. The use of a CQT spectrogram as an input is widespread [34]–[37] to make effective use of DNNs with convolutional layers. Because songs have various tempos and non-aligned timings, previous studies have used beat synchronization [2]–[6] to align tempos. We thus apply a beat-tracking technique [38] to songs. Using the beat-tracking results, we apply bilinear filtering to both vocal and accompaniment CQTs such that each CQT has the same number of time samples per bar because each CQT of songs has a different number of time samples per bar. We then cut both vocal and accompaniment CQTs into multiple bars, *i.e.*, beat-synchronous bar-wise CQTs (BBCQTs), and use them as training data (see Subsection III-A for details).

Our proposed method is based on deep metric learning as follows. Inspired by previous studies [12], [15], [22], we consider two DNN-based encoders (vocal and accompaniment). These encoders consist of Visual Geometry Group (VGG)-like networks [39] and jointly embed BBCQT representations of vocal and accompaniment tracks (*i.e.*, vocal and accompaniment BBCQTs) into a shared vector space called *vocal-accompaniment shared vector space*. We design a loss function inspired by the multi-class N-pair loss [40], enabling the encoders to learn effective joint embeddings for estimating the compatibility between vocal and accompaniment tracks. That is, our loss function enables the two encoders to locate the embedded feature vectors of original pairs of vocal and accompaniment tracks nearby in the shared vector space and locate those of other random pairs far away, where the original pairs are assumed to be statistically more compatible than the random pairs (see Subsection III-B for details).

Once the training finishes, the distance within this space indicates the degree of compatibility, by which we can estimate the compatibility of a novel pair of vocal and accompaniment tracks (see Subsection III-C for details).

A. PRE-PROCESSING STEPS

We apply the following processing steps to songs. Figure 2 illustrates these steps.

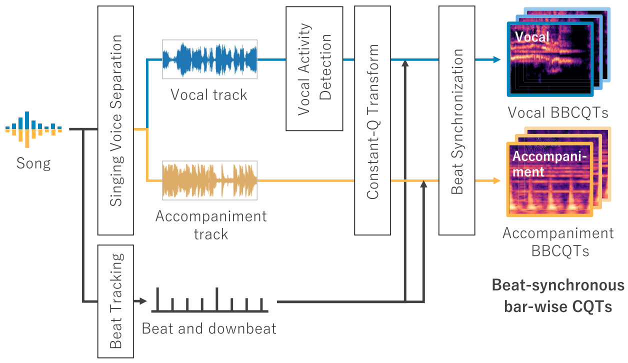


FIGURE 2. Pre-processing steps. All songs are separated into pairs of vocal and accompaniment tracks. The tracks are then converted into beat-synchronous bar-wise constant-Q transforms (BBCQTs).

1) SINGING VOICE SEPARATION

We perform singing voice separation to use a large collection of songs with vocal and accompaniment tracks that are not directly available. Specifically, our implementation uses the "2stems" model in Spleeter [9], which can separate polyphonic sound mixtures of a song into a pair of vocal and accompaniment tracks.

2) VOCAL ACTIVITY DETECTION

We use the vocal activity detection method proposed by Kum and Nam [32], the detection of which is conducted in 0.01-second increments. We discard bars in which non-vocal sections occupy more than half the length of a bar.

3) CONSTANT-Q TRANSFORM

We transform each waveform of both vocal and accompaniment tracks into CQT representations to handle them as images. We use librosa [41] to calculate the CQTs, where we set the number of frequency bins to 84 (*i.e.*, 7 octaves) and hop length to 256.

4) BEAT TRACKING

We track beats and downbeats to divide both vocal and accompaniment tracks into bars. To do this, we use the method proposed by Böck *et al.* [38] for jointly detecting beats and downbeats. This method is available as an open-source library called madmom [42], the calculation of which is conducted in 0.01-second increments. Only bars consisting of four beats are used as training data.

5) BEAT SYNCHRONIZATION

We divide each CQT into BBCQTs using the results of beat tracking. To feed the divided CQTs to the encoders, we also need to ensure their dimensions are always the same across bars and songs. To achieve this, we use bilinear filtering to resize every CQT such that the number of time samples becomes 32 per bar (*i.e.*, beat synchronization). Note that we apply the filter only to CQTs along the time axis, so that each frequency bin of a CQT maintains its independence. We then divide each CQT into bar-wise CQTs (*i.e.*, BBCQTs).

B. TRAINING

1) ENCODER ARCHITECTURE

We use a VGG-like network [39] (see Table 1) to encode BBCQTs into feature vectors. We design the same architecture for both the vocal and accompaniment encoders (note that they do not share their weights). Each encoder embeds either vocal or accompaniment BBCQTs into the same 32-dimensional feature space called the vocal-accompaniment shared vector space.

As shown in Table 1, each of the first five blocks of our encoder consists of a max-pooling layer and two convolution layers. Each convolution layer is followed by batch normalization [43] and a rectified linear unit (ReLU) [44]. An average pooling (avgpool) layer is applied to the tensor output from the first five blocks. Through a fully connected (fc) layer, we finally obtain a time-dependent embedded feature vector (32 dimensions), where each dimension corresponds to a time sample.

2) TRAINING STRATEGY

Inspired by the multi-class N-pair loss [40], we designed a loss function called *compatibility loss*. While training, we use N original pairs of vocal and accompaniment BBCQTs to construct mini-batches of both BBCQTs. When we construct mini-batches, we first choose N random songs from the training data and then choose a random bar from each song. Therefore, we can obtain N compatible pairs and $N(N-1)$ random pairs from N original pairs by making all possible combinations of them (*i.e.*, N^2 pairs in total). Each mini-batch of vocal and accompaniment BBCQTs is encoded into N vocal-embedded feature vectors, $\{\mathbf{v}_n \in \mathbb{R}^{32}\}_{n=1}^{N}$, and N accompaniment-embedded feature vectors, $\{\mathbf{a}_n \in \mathbb{R}^{32}\}_{n=1}^{N}$, using our encoders. Having the same indices within each mini-batch indicates that they are originally from the same data sample (*e.g.*, $\mathbf{v}_1$ and $\mathbf{a}_1$ are generated from the same bar in the same song).

Using vocal- and accompaniment-embedded feature vectors, $\{\mathbf{v}_n\}_{n=1}^{N}$ and $\{\mathbf{a}_n\}_{n=1}^{N}$, we calculate a *compatibility matrix*, $\mathbf{M} \in \mathbb{R}^{N\times N}$, which is defined as

$$m_{i,j} = f(\mathbf{v}_i, \mathbf{a}_j), \tag{1}$$

where $m_{i,j}$ is the entry in the i-th row and j-th column of the compatibility matrix $\mathbf{M}$, and f is a scalar-valued function that reflects the compatibility between the two vectors. As embedded feature vectors lie in the shared vector space, we can simply define f as a similarity function between the embedded feature vectors of vocal and accompaniment BBCQTs. We compare the cosine similarity $f_{\cos}$ and dot product f_{dot} on the basis of the definition of f in Section IV as follows:

$$f_{\cos}(\mathbf{v}_i, \mathbf{a}_j) = \frac{\mathbf{v}_i \cdot \mathbf{a}_j}{\|\mathbf{v}_i\|\,\|\mathbf{a}_j\|}, \tag{2}$$

$$f_{\mathrm{dot}}(\mathbf{v}_i, \mathbf{a}_j) = \mathbf{v}_i \cdot \mathbf{a}_j. \tag{3}$$

TABLE 1. Our encoder architecture, which is a VGG-like network [39]. It encodes a BBCQT (of either a vocal or an accompaniment) into a time-dependent embedded feature vector (32 dimensions).

Layer	Size	kernel size	Stride
Input (BBCQT)	$1 \times 84 \times 32$	—	—
maxpool1	$1 \times 42 \times 32$	2×1	2×1
conv1_1	$32 \times 42 \times 32$	3×3	1×1
conv1_2	$32 \times 42 \times 32$	3×3	1×1
maxpool2	$32 \times 21 \times 32$	2×1	2×1
conv2_1	$64 \times 21 \times 32$	3×3	1×1
conv2_2	$64 \times 21 \times 32$	3×3	1×1
maxpool3	$128 \times 10 \times 32$	2×1	2×1
conv3_1	$128 \times 10 \times 32$	3×3	1×1
conv3_2	$128 \times 10 \times 32$	3×3	1×1
maxpool4	$128 \times 5 \times 32$	2×1	2×1
conv4_1	$128 \times 5 \times 32$	3×3	1×1
conv4_2	$128 \times 5 \times 32$	3×3	1×1
maxpool5	$128 \times 2 \times 32$	2×1	2×1
conv5_1	$128 \times 2 \times 32$	3×3	1×1
conv5_2	$128 \times 2 \times 32$	3×3	1×1
avgpool1	128×32	2×1	1×1
fc1	1×32	—	—
Output (vector)	32	—	—

We then calculate compatibility loss $\mathcal{L}$ from the compatibility matrix $\mathbf{M}$ as follows:

$$\mathcal{L} = -\frac{1}{N}\sum_{j=1}^{N}\log\frac{\exp(m_{j,j})}{\sum_{i=1}^{N}\exp(m_{i,j})} - \frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(m_{i,i})}{\sum_{j=1}^{N}\exp(m_{i,j})}. \quad (4)$$

The first term is considered the average of the *cross-entropy* (CE) values for each column (*i.e.*, accompaniment-wise CE). Similarly, the second term is considered the average of the CE values for each row (*i.e.*, vocal-wise CE). Both terms enable the diagonal entries of the compatibility matrix (*i.e.*, those calculated from original pairs of vocal and accompaniment BBCQTs) to be larger. We train our encoders by minimizing this loss function value. Figure 3 illustrates this process.

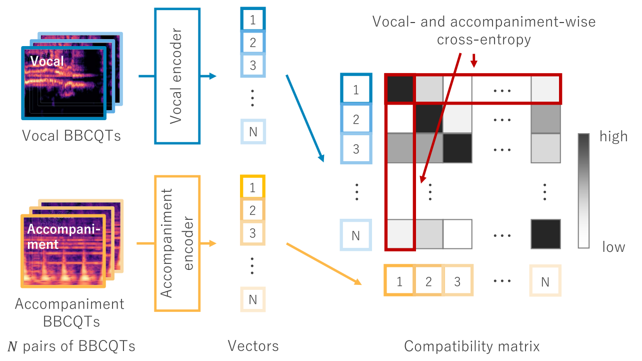


FIGURE 3. Details of our training process. We first construct the compatibility matrix from *N* pairs of vocal and accompaniment BBCQTs. We then calculate the compatibility loss on the basis of the cross-entropy (CE) values for each row and column of the compatibility matrix.

We implemented the encoders using PyTorch [45] and optimized them using Adam [46]. We set the learning rate to 1×10^{-3} initially and decreased it to 1×10^{-4} and 1×10^{-5} at the 80th and 160th epochs, respectively. The training process was completed in 250 epochs. We used $N = 2048$ for our experiments.

C. VOCAL-ACCOMPANIMENT COMPATIBILITY ESTIMATION

Once the training of the encoders has been conducted, we can estimate the compatibility of a novel pair of vocal and accompaniment tracks as follows. First, we perform the pre-processing steps in the same manner as the training (Subsection III-A). We then encode the obtained BBCQTs into embedded feature vectors using the trained encoders. Finally, we calculate a compatibility value from the f in Eq. (1).

Note that we estimate compatibility values in a bar-wise manner. If we need a compatibility value between tracks that are longer than a single bar (*e.g.*, automatic mashup), we merely average the compatibility values across all bars.

IV. EVALUATION

We conducted comparative experiments to evaluate the effectiveness of our method. We took a ranking-based evaluation approach [25], [47]–[49] to quantitatively evaluate the performance of compatibility estimation. Ranking-based evaluation is used to assess how accurately a target can be found from multiple candidates. We set up two tasks to evaluate vocal-accompaniment compatibility estimation: *query-by-vocal*, which uses a vocal track as a query for retrieving an original accompaniment track, and *query-by-accompaniment*, which uses an accompaniment track as a query for retrieving an original vocal track.

A. DATASET

We constructed the trial listening dataset from songs available on a music service for trial listening. This dataset consists of 910,803 songs of 64,535 artists. Each song file represents a short music excerpt (30 sec, 44.1 kHz). Table 2 lists the music genres of this dataset.

TABLE 2. List of primary music genres and the number of songs in the trial listening dataset.

Music Genre	#Songs
Pop	269,560 (29.6%)
Rock	217,366 (23.9%)
Alternative	94,076 (10.3%)
Hip-Hop/Rap	39,073 (4.3%)
Country	34,282 (3.8%)
R&B/Soul	32,079 (3.5%)
Others (Anime, Metal, Singer/Songwriter, etc.)	224,367 (24.6%)

We divided this dataset into training, validation, and test sets in a ratio of 8:1:1. The songs in each set are constructed in equal percentages with respect to the music genres, as shown in Table 2, such that each set has the same weight in music styles. After applying the pre-processing steps described in Subsection III-A, we obtained 8,105,813 (training: 6,488,005, validation: 808,887, test: 808,921) bars of each of vocal and accompaniment tracks.

B. RANKING-BASED EVALUATION METRICS

We used the mean reciprocal rank (MRR) [50] and top-k for comparative experiments. We define a rank as the rank position of the compatibility value derived from a query vocal (accompaniment) track and its original accompaniment (vocal) track. MRR is used to compute an average of the reciprocal ranks as follows.

$$\mathrm{MRR} = \frac{1}{|Q|}\sum_{l=1}^{|Q|}\frac{1}{\mathrm{rank}_l}, \quad (5)$$

where Q is a set of queries, and rank_l is the rank of the l-th query. A larger MRR indicates that the compatibility is more successfully estimated. Top-k counts the number of times when the original pair of vocal and accompaniment tracks is

TABLE 3. Results of ranking-based evaluations on the test set of the trial listening dataset. For comparative experiments, k was set to 1, 10, and 100. A larger value indicates that the compatibility was more successfully estimated. Our method with dot product (Dot) performed the best in both query-by-vocal and query-by-accompaniment settings. Random means the expected value when we use purely random estimation and is shown just for reference.

	Query-by-Vocal				Query-by-Accompaniment			
	MRR	Top-1 (%)	Top-10 (%)	Top-100 (%)	MRR	Top-1 (%)	Top-10 (%)	Top-100 (%)
Random	1.75×10^{-5}	1.24×10^{-4}	1.24×10^{-3}	1.24×10^{-2}	1.75×10^{-5}	1.24×10^{-4}	1.24×10^{-3}	1.24×10^{-2}
Cos-Chromagram	1.90×10^{-4}	8.04×10^{-3}	3.07×10^{-2}	1.21×10^{-1}	2.44×10^{-4}	1.09×10^{-2}	4.22×10^{-2}	1.37×10^{-1}
Dot-Chromagram	2.40×10^{-5}	2.47×10^{-4}	2.35×10^{-3}	2.05×10^{-2}	1.89×10^{-5}	2.47×10^{-4}	9.89×10^{-4}	1.19×10^{-2}
Cos (Ours)	1.01×10^{-2}	4.45×10^{-1}	1.93×10^{0}	6.40×10^{0}	8.89×10^{-3}	3.81×10^{-1}	1.71×10^{0}	5.37×10^{0}
Dot (Ours)	$\mathbf{2.46 \times 10^{-1}}$	$\mathbf{1.88 \times 10^{1}}$	$\mathbf{3.55 \times 10^{1}}$	$\mathbf{5.03 \times 10^{1}}$	$\mathbf{2.38 \times 10^{-1}}$	$\mathbf{1.84 \times 10^{1}}$	$\mathbf{3.39 \times 10^{1}}$	$\mathbf{4.75 \times 10^{1}}$

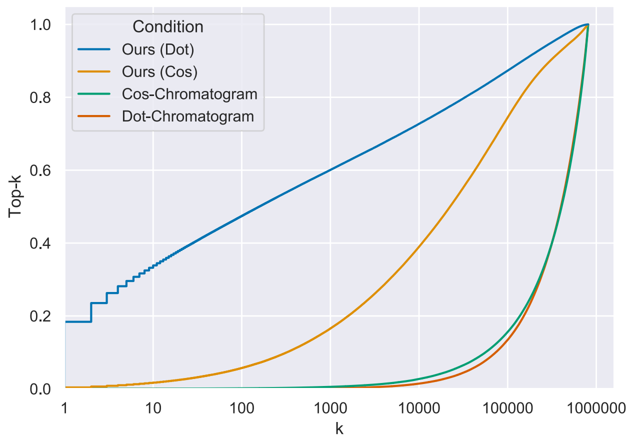


FIGURE 4. Empirical cumulative distribution functions (CDFs) of the ranks in the query-by-vocal setting.

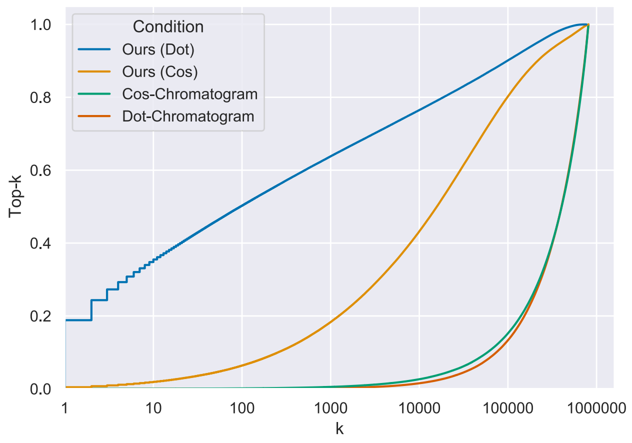


FIGURE 5. Empirical cumulative distribution functions (CDFs) of the ranks in the query-by-accompaniment setting.

among the top k ranks as follows.

$$\text{top-}k = \frac{1}{|Q|} \sum_{l=1}^{|Q|} \alpha_l, \tag{6}$$

$$\alpha_l = \begin{cases} 1 & \text{if rank}_l \leq k, \\ 0 & \text{if rank}_l > k. \end{cases} \tag{7}$$

We added the normalization factor $1/|Q|$ to Eq. (6) for comparison. A larger top-k indicates that the compatibility is more successfully estimated. We used all bars in the test set as queries for comparative experiments ($|Q| = 808,921$).

C. CONDITIONS

We compared four conditions: a baseline method based on a chromagram with (1) cosine similarity (Cos-Chromagram) and (2) dot product (Dot-Chromagram), and our method with (3) cosine similarity (Cos) and (4) dot product (Dot).

The use of a chromagram as a feature vector can be considered a typical rule-based approach, and previous studies took this approach in combination with either cosine similarity or dot product [2], [4], [25], [51], [52]. This is why we included conditions (1) and (2) as the baseline methods. For these conditions, the dataset was processed similar to but slightly different from the pre-processing steps in Subsection III-A: we calculated chromagrams using librosa [41] instead of CQTs.

D. RESULTS

Table 3 shows the results. Our methods (*i.e.*, Cos and Dot) performed much better than the baselines (*i.e.*, Cos-Chromagram, Dot-Chromagram) in both the query-by-vocal and query-by-accompaniment settings. From these results, our method successfully estimated that original (thus assumed to be compatible) pairs are generally more compatible than random pairs, which validates our method. In terms of the used metrics, our method using Dot achieved better results than using Cos in both the query-by-vocal and query-by-accompaniment settings. Figures 4 and 5 visualize empirical cumulative distribution functions (CDFs) of the ranks among top k ($1 \leq k \leq 808,921$) in the query-by-vocal and query-by-accompaniment settings, respectively, for better understanding. The empirical CDFs clearly show the advantage of our method.

From these observations, we consider that the proposed method, especially using dot product, can successfully estimate compatibility between vocal and accompaniment tracks.

V. DISCUSSION AND FUTURE WORK

A. INPUT REPRESENTATION

We used the CQTs (vocal and accompaniment BBC-QTs) as the inputs of our encoders. Other possible inputs include short-time Fourier transform (STFT) [53],

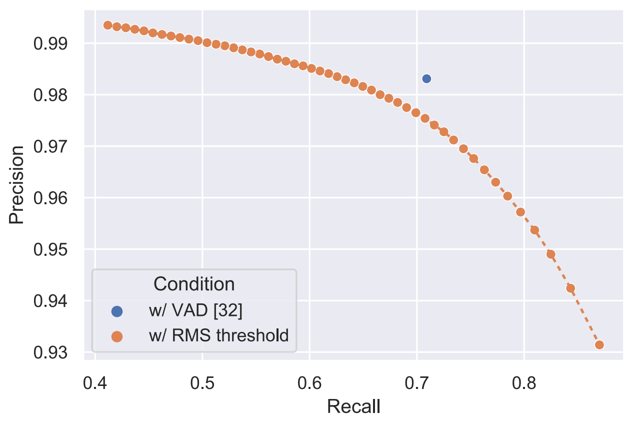


FIGURE 6. Precision-recall of vocal activity detection. Our method (w/ VAD [32]) is superior to the method merely applying thresholds to RMS (w/ RMS threshold).

mel-scaled [54] STFT (Mel-STFT), continuous wavelet transform (CWT) [55], mel-frequency cepstrum coefficient (MFCC) [56], and chromagram representations. Several studies using DNNs with convolutional layers have discussed which input representation is effective. Huzaifah [35] revealed that the use of an STFT spectrogram, a Mel-STFT spectrogram, or a CQT spectrogram was superior to that of a CWT spectrogram or MFCCs in environmental sound classification tasks. Dubey *et al.* [37] showed that the use of a CQT spectrogram outperformed that of a STFT spectrogram in music source separation tasks. From these discussions, we believe that the use of a CQT spectrogram as an input is also beneficial in the vocal-accompaniment compatibility estimation task. Both a CQT spectrogram and a chromagram are based on the twelve equal temperament, which divides an octave into twelve parts by log-scale. A CQT spectrogram includes more detailed features about pitch compared with a chromagram. This difference will lead to a trade-off between computational cost and performance. A chromagram will be a good candidate when computational resources are limited.

B. THE USAGE OF BEAT TRACKING

We adopted beat-tracking results for audio tracks and aligned the number of time samples per bar using bilinear filtering. Another possible approach is to use fixed- (or variable-) length audio tracks directly. In several MIR tasks (*e.g.*, genre classification [57], [58], and singing voice separation [7]–[9]), fixed- (or variable-) length audio tracks are used as input to the DNNs. Note that the proposed method does not prevent the use of fixed- (or variable-) length audio tracks: it can handle those audio tracks by using bilinear filtering to match a specific number of time samples while requiring re-training our encoders with those audio tracks. Moreover, the motivation for using beat tracking is that beat synchronization is an important factor in MIR applications (*e.g.*, mashup and mixing) [2]–[6].

C. EFFECTIVENESS OF VOCAL ACTIVITY DETECTION

To validate the effectiveness of our method, which uses vocal activity detection [32] in the pre-processing steps, we conducted an additional experiment. We used the MUSDB18 dataset [59], which provides 150 polyphonic sound mixtures and their isolated tracks including vocal tracks. We calculated the precision and recall of the estimated vocal activity by considering the vocal activity of the vocal tracks in the MUSDB18 dataset as ground truth. We compared vocal activity detection [32] with root mean square (RMS) thresholds when applying the pre-processing steps to polyphonic sound mixtures of the MUSDB18 dataset. We set the RMS thresholds in a range from 0.001 to 0.05 at an interval of 0.001. Figure 6 shows the precision-recall under each condition. The results clearly indicate the effectiveness of our pre-processing.

D. MULTI-CLASS N-PAIR LOSS VS. TRIPLET LOSS

Our current implementation uses the compatibility loss inspired by multi-class N-pair loss [40]. That is, each original pair of vocal and accompaniment tracks has a unique class. Conventional triplet loss takes into account only one class of a positive sample versus one class of a negative sample, making the training process slow and inefficient. By contrast, multi-class N-pair loss takes into account one class of a positive sample versus multiple classes of negative samples simultaneously, enabling more stable embedding.

E. EFFECTIVENESS OF COMPATIBILITY LOSS

To validate the effectiveness of our loss function (*i.e.*, the sum of vocal- and accompaniment-wise CE values), we conducted an ablation study. We used only a vocal- or accompaniment-wise CE value as a loss function. Under all conditions, we used dot product for both training the vocal and accompaniment encoders and calculating compatibility values. Table 4 lists the results, which clearly show the effectiveness of the design of the loss function.

TABLE 4. Mean reciprocal rank (MRR) on the test set of the trial listening dataset with respect to the ablation study on the loss functions. Our compatibility loss function achieved good performance for both query-by-vocal and query-by-accompaniment settings while the vocal- and accompaniment-wise CE loss functions achieved good performance only for one of these two settings.

	Query-by-Vocal	Query-by-Accompaniment
Vocal-wise CE	$\mathbf{2.40 \times 10^{-1}}$	2.63×10^{-5}
Accompaniment-wise CE	3.16×10^{-5}	$\mathbf{2.54 \times 10^{-1}}$
Compatibility loss function	$\mathbf{2.46 \times 10^{-1}}$	$\mathbf{2.38 \times 10^{-1}}$

F. COMPATIBILITY VALUE

Our method can estimate a compatibility value of a new pair of vocal and accompaniment tracks. As shown in Figure 7, our method (especially using dot product) succeeded in estimating a compatibility value of any pair so that the

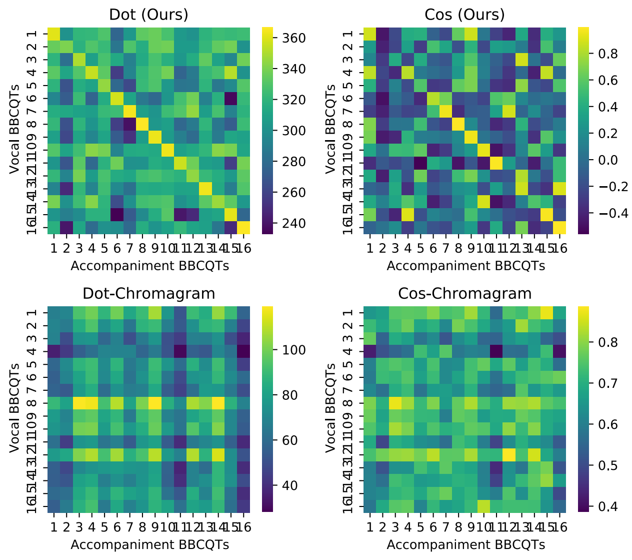


FIGURE 7. A part of the compatibility matrix under each condition. We randomly sampled 16 songs (*i.e.*, 16 original pairs of vocal and accompaniment BBCQTs) from the test set. Each diagonal entry of the compatibility matrix indicates the compatibility value of an original pair. Our method (top) could estimate that original pairs are more compatible than other random pairs.

compatibility value of an original pair was higher than those of most other random pairs. These results suggest that our method successfully captures the relevance of the estimated compatibility values as expected. The next important step is to investigate how well the estimated compatibility values agree with human perception by conducting a large perceptual study. It would also be interesting to conduct a deeper analysis of what musical factors (and human factors) are important in understanding the compatibility between vocal and accompaniment tracks, and we believe that the findings from this study can be used in this analysis.

G. EXAMPLE APPLICATION: AUTOMATIC MASHUP

As described in Section I, vocal-accompaniment compatibility estimation can be used for creating mashup songs. We describe an automatic mashup tool using our compatibility estimation as an example application. The tool requires a dataset containing vocal and accompaniment tracks and a trained model using this dataset. As an input to the tool, a user can select an excerpt of a vocal track or that of an accompaniment track (*e.g.*, verse, bridge, or chorus). Note that such excerpts need to be bar-wise. When the tool takes an excerpt of a vocal track as the input, it estimates the compatibility between the excerpt and all excerpts from accompaniment tracks in the dataset. When the tool applies an excerpt of an accompaniment track as the input, it estimates the compatibility between that excerpt and all excerpts from vocal tracks in the dataset. After selecting the excerpt that should be paired to the input excerpt, the tool stretches the time of the selected excerpt to fit the input excerpt without pitch shifting. Finally, the tool mixes them and creates a new mashup song, the duration of which is the same as that of the input.

H. POSSIBILITIES OF COMPATIBILITY ESTIMATION

We believe that compatibility estimation has the potential to enhance many other MIR applications (*e.g.*, composition support) and yield new research directions. This study can be extended to compatibility between tracks of a non-vocal melody and an accompaniment; that between tracks of a non-audio-signal melody and an accompaniment such as those stored in MIDI representations; and that between any non-melody track and another track.

VI. CONCLUSION

We proposed a method for estimating the compatibility between vocal and accompaniment tracks. This method is based on deep metric learning: it embeds both vocal and accompaniment tracks into the shared vector space using two encoders, which lay the embedded feature vectors of a compatible pair of vocal and accompaniment tracks close to each other and those of an incompatible pair far from each other. To enable this, we constructed a large dataset called the trial listening dataset containing 910,803 songs available on a music service and generated the training data in a self-supervised manner. Our comparative experiments using ranking-based evaluation clarified that our method is superior to baseline methods.

REFERENCES

[1] C. Boone, "Mashing: Toward a typology of recycled music," *Music Theory Online*, vol. 19, no. 3, Sep. 2013. [Online]. Available: https://mtosmt.org/issues/mto.13.19.3/mto.13.19.3.boone.html

[2] C.-L. Lee, Y.-T. Lin, Z.-R. Yao, F.-Y. Lee, and J.-L. Wu, "Automatic mashup creation by considering both vertical and horizontal mashabilities," in *Proc. Int. Soc. Music Inf. Retr. (ISMIR) Conf.*, Oct. 2015, pp. 399–405.

[3] G. Bernardes, M. E. P. Davies, and C. Guedes, "A hierarchical harmonic mixing method," in *Proc. Int. Symp. Comput. Music Multidisciplinary Res. (CMMR)*, Sep. 2017, pp. 151–170.

[4] M. E. P. Davies, P. Hamel, K. Yoshii, and M. Goto, "AutoMashUpper: Automatic creation of multi-song music mashups," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 12, pp. 1726–1737, Dec. 2014.

[5] R. Gebhardt, M. Davies, and B. Seeber, "Psychoacoustic approaches for harmonic music mixing," *Appl. Sci.*, vol. 6, no. 5, p. 123, May 2016.

[6] C. Maçãs, A. Rodrigues, G. Bernardes, and P. Machado, "MixMash: An assistive tool for music mashup creation from large music collections," *Int. J. Art, Culture Design Technol.*, vol. 8, no. 2, pp. 20–40, Jul. 2019.

[7] D. Stoller, S. Ewert, and S. Dixon, "Wave-U-Net: A multi-scale neural network for end-to-end audio source separation," in *Proc. Int. Soc. Music Inf. Retr. (ISMIR) Conf.*, Sep. 2018, pp. 334–340.

[8] F.-R. Stöter, S. Uhlich, A. Liutkus, and Y. Mitsufuji, "Open-unmix—A reference implementation for music source separation," *J. Open Source Softw.*, vol. 4, no. 41, p. 1667, Sep. 2019.

[9] R. Hennequin, A. Khlif, F. Voituret, and M. Moussallam, "Spleeter: A fast and efficient music source separation tool with pre-trained models," *J. Open Source Softw.*, vol. 5, no. 50, p. 2154, Jun. 2020.

[10] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel, "Image-based recommendations on styles and substitutes," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2015, pp. 43–52.

[11] X. Song, F. Feng, J. Liu, Z. Li, L. Nie, and J. Ma, "NeuroStylist: Neural compatibility modeling for clothing matching," in *Proc. 25th ACM Int. Conf. Multimedia*, Oct. 2017, pp. 753–761.

[12] A. Veit, B. Kovacs, S. Bell, J. McAuley, K. Bala, and S. Belongie, "Learning visual clothing style with heterogeneous dyadic co-occurrences," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4642–4650.

[13] X. Han, Z. Wu, Y.-G. Jiang, and L. S. Davis, "Learning fashion compatibility with bidirectional LSTMs," in *Proc. 25th ACM Int. Conf. Multimedia*, Oct. 2017, pp. 1078–1086.

[14] X. Song, X. Han, Y. Li, J. Chen, X.-S. Xu, and L. Nie, "GP-BPR: Personalized compatibility modeling for clothing matching," in *Proc. 27th ACM Int. Conf. Multimedia*, Oct. 2019, pp. 320–328.

[15] T. Liu, A. Hertzmann, W. Li, and T. Funkhouser, "Style compatibility for 3D furniture models," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 85:1–85:9, Jul. 2015.

[16] P. O'Donovan, A. Agarwala, and A. Hertzmann, "Color compatibility from large datasets," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 63:1–63:12, Jul. 2011.

[17] T. Fujishima, "Real-time chord recognition of musical sound: A system using common lisp music," in *Proc. Int. Comput. Music Conf. (ICMC)*, Oct. 1999, pp. 464–467.

[18] V. R. de Sa, "Learning classification with unlabeled data," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Nov. 1994, pp. 112–119.

[19] A. Owens and A. A. Efros, "Audio-visual scene analysis with self-supervised multisensory features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 631–648.

[20] B. Korbar, D. Tran, and L. Torresani, "Cooperative learning of audio and video models from self-supervised synchronization," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Dec. 2018, pp. 7763–7774.

[21] Y. Tian, J. Shi, B. Li, Z. Duan, and C. Xu, "Audio-visual event localization in unconstrained videos," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 247–263.

[22] B. Li and A. Kumar, "Query by video: Cross-modal music retrieval," in *Proc. Int. Soc. Music Inf. Retr. (ISMIR) Conf.*, Nov. 2019, pp. 604–611.

[23] K. Lee and J. Nam, "Learning a joint embedding space of monophonic and mixed music signals for singing voice," in *Proc. Int. Soc. Music Inf. Retr. (ISMIR) Conf.*, Nov. 2019, pp. 295–302.

[24] Y. Zhang, B. Pardo, and Z. Duan, "Siamese style convolutional neural networks for sound search by vocal imitation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 2, pp. 429–441, Feb. 2019.

[25] X. Qi, D. Yang, and X. Chen, "Triplet convolutional network for music version identification," in *Proc. Int. Conf. Multimedia Modeling (MMM)*, Oct. 2018, pp. 544–555.

[26] L. Nanni, A. Rigo, A. Lumini, and S. Brahnam, "Spectrogram classification using dissimilarity space," *Appl. Sci.*, vol. 10, no. 12, p. 4176, Jun. 2020.

[27] M. Mueller, A. Arzt, S. Balke, M. Dorfer, and G. Widmer, "Cross-modal music retrieval and applications: An overview of key methodologies," *IEEE Signal Process. Mag.*, vol. 36, no. 1, pp. 52–62, Jan. 2019.

[28] R. Arandjelovic and A. Zisserman, "Look, listen and learn," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 609–617.

[29] R. Arandjelovic and A. Zisserman, "Objects that sound," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 435–451.

[30] Y. Aytar, T. Pfaff, D. Budden, T. Paine, Z. Wang, and N. de Freitas, "Playing hard exploration games by watching youtube," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Dec. 2018, pp. 2930–2941.

[31] D. Harwath, A. Torralba, and J. Glass, "Unsupervised learning of spoken language with visual context," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Dec. 2016, pp. 1858–1866.

[32] S. Kum and J. Nam, "Joint detection and classification of singing voice melody using convolutional recurrent neural networks," *Appl. Sci.*, vol. 9, no. 7, p. 1324, Mar. 2019.

[33] J. C. Brown, "Calculation of a constant Q spectral transform," *J. Acoust. Soc. Amer.*, vol. 89, no. 1, pp. 425–434, 1991.

[34] R. M. Bittner, B. McFee, J. Salamon, P. Li, and J. P. Bello, "Deep salience representations for F_0 estimation in polyphonic music," in *Proc. Int. Soc. Music Inf. Retr. (ISMIR) Conf.*, Oct. 2017, pp. 63–70.

[35] M. Huzaifah, "Comparison of time-frequency representations for environmental sound classification using convolutional neural networks," 2017, *arXiv:1706.07156*. [Online]. Available: http://arxiv.org/abs/1706.07156

[36] S. Huang, Q. Li, C. Anil, X. Bao, S. Oore, and R. B. Grosse, "TimbreTron: A WaveNet (cycleGAN (CQT (audio))) pipeline for musical timbre transfer," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2019, pp. 1–17.

[37] M. Dubey, H. Jones, A. Thresher, and G. Kenyon, "Separating musical sources with convolutional sparse coding," in *Proc. 2nd Int. Conf. Appl. Intell. Syst. (APPIS)*, Jan. 2016, pp. 1–5.

[38] S. Böck, F. Krebs, and G. Widmer, "Joint beat and downbeat tracking with recurrent neural networks," in *Proc. Int. Soc. Music Inf. Retr. (ISMIR) Conf.*, Aug. 2016, pp. 255–261.

[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2015, pp. 1–14.

[40] K. Sohn, "Improved deep metric learning with multi-class N-pair loss objective," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Dec. 2016, pp. 1857–1865.

[41] B. McFee *et al.* (Jan. 2020). LibROSA/LibROSA: 0.7.2. Zenodo. [Online]. Available: https://zenodo.org/record/3606573

[42] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, "madmom: A new Python audio and music signal processing library," in *Proc. ACM Int. Conf. Multimedia (MM)*, Amsterdam, The Netherlands, Oct. 2016, pp. 1174–1178.

[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Jul. 2015, pp. 448–456.

[44] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. Int. Conf. Mach. Learn. (ICML)*, J. Fürnkranz and T. Joachims, Eds., Jun. 2010, pp. 807–814.

[45] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and A. Desmaison, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Dec. 2019, pp. 8024–8035.

[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, May 2015, pp. 1–13.

[47] M. Bretan, G. Weinberg, and L. Heck, "A unit selection methodology for music generation using deep neural networks," in *Proc. Int. Conf. Comput. Creativity (ICCC)*, Jun. 2017, pp. 72–79.

[48] A. Jansen, M. Plakal, R. Pandya, D. P. W. Ellis, S. Hershey, J. Liu, R. C. Moore, and R. A. Saurous, "Unsupervised learning of semantic audio representations," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 126–130.

[49] M. Bretan and L. Heck, "Learning semantic similarity in music via self-supervision," in *Proc. Int. Soc. Music Inf. Retr. (ISMIR) Conf.*, Nov. 2019, pp. 446–453.

[50] N. Craswell, *Mean Reciprocal Rank*. New York, NY, USA: Springer, 2009, p. 1703.

[51] D. P. W. Ellis and G. E. Poliner, "Identifying 'cover songs' with chroma features and dynamic programming beat tracking," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2007, pp. 1429–1432.

[52] T. Bertin-Mahieux and D. P. W. Ellis, "Large-scale cover song recognition using hashed chroma landmarks," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust. (WASPAA)*, Oct. 2011, pp. 117–120.

[53] D. Gabor, "Theory of communication," *J. Inst. Electr. Eng.-I Gen.*, vol. 94, no. 73, p. 58, 1947.

[54] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," *J. Acoust. Soc. Amer.*, vol. 8, no. 3, pp. 185–190, Jan. 1937.

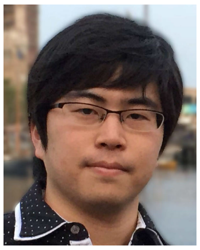
[55] C. K. Chui, *An Introduction to Wavelets*. New York, NY, USA: Academic, 1992.

[56] P. Mermelstein, "Distance measures for speech recognition, psychological and instrumental," *Pattern Recognit. Artif. Intell.*, vol. 116, no. 1, pp. 374–388, 1976.

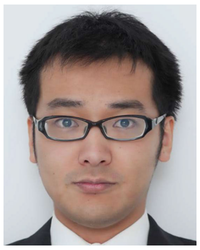
[57] W. Zhang, W. Lei, X. Xu, and X. Xing, "Improved music genre classification with convolutional neural networks," in *Proc. INTERSPEECH*, Sep. 2016, pp. 3304–3308.

[58] L. Nanni, Y. M. G. Costa, D. R. Lucio, C. N. Silla, Jr., and S. Brahnam, "Combining visual and acoustic features for audio classification tasks," *Pattern Recognit. Lett.*, vol. 88, pp. 49–56, Mar. 2017.

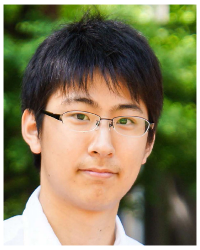
[59] Z. Rai, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner. (Dec. 2017). The MUSDB18 Corpus for Music Separation. Zenodo. [Online]. Available: https://zenodo.org/record/1117372



TAKAYUKI NAKATSUKA (Member, IEEE) received the Ph.D. degree from the Department of Pure and Applied Physics, Waseda University, in 2021. He is currently a Researcher with the National Institute of Advanced Industrial Science and Technology (AIST), Japan. His research interests include physically-based animation, human motion analysis, human–computer interaction, music information retrieval, and virtual reality.



KENTO WATANABE received the B.E. and Ph.D. degrees from Tohoku University, in 2013 and 2018, respectively. He is currently a Researcher with the National Institute of Advanced Industrial Science and Technology (AIST), Japan. His research interests include lyrics information processing, natural language processing, machine learning, and human–computer iteration.



YUKI KOYAMA received the Ph.D. degree from The University of Tokyo, in 2017. He is currently a Researcher with the National Institute of Advanced Industrial Science and Technology (AIST), Japan. His research interests include computer graphics and human–computer interaction.

MASAHIRO HAMASAKI received the Ph.D. degree in informatics from Soken University, in 2005. He is currently the Group Leader of the Media Interaction Group, National Institute of Advanced Industrial Science and Technology (AIST), Japan. He is also an Associate Professor with the University of Tsukuba. His research interests include online community assistance, knowledge construction, social media analysis, and web intelligence.

MASATAKA GOTO received the Doctor of Engineering degree from Waseda University, in 1998. He is currently a Prime Senior Researcher with the National Institute of Advanced Industrial Science and Technology (AIST), Japan. Over the past 29 years, he has published more than 300 articles in refereed journals and international conferences and has received 54 awards, including several best paper awards, best presentation awards, the Tenth Japan Academy Medal, and the Tenth JSPS PRIZE. As the Research Director, he began the OngaACCEL Project, in 2016, and the RecMus Project, in 2021, which are five-year JST-funded research projects (ACCEL and CREST) related to music technologies.

SHIGEO MORISHIMA (Member, IEEE) was born in August 1959. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from The University of Tokyo, in 1982, 1984, and 1987, respectively. He was a Visiting Professor with the University of Toronto, from 1994 to 1995, and an Invited Researcher with the Advanced Telecommunication Research Institute, from 1999 to 2011. He is currently a Professor with the School of Advanced Science and Engineering, Waseda University. He received many awards and takes an administration board member of several societies. He was the General Chair of ACM VRST2018 and VR/AR Adviser of SIGGRAPH ASIA 2018.