QUERY-BY-BLENDING: A MUSIC EXPLORATION SYSTEM BLENDING LATENT VECTOR REPRESENTATIONS OF LYRIC WORD, SONG AUDIO, AND ARTIST

Kento Watanabe **Masataka Goto**
National Institute of Advanced Industrial Science and Technology (AIST), Japan
`{kento.watanabe, m.goto}@aist.go.jp`

ABSTRACT

This paper presents *Query-by-Blending*, a novel music exploration system that enables users to find unfamiliar music content by flexibly combining three musical aspects: lyric word, song audio, and artist. Although there are various systems for music retrieval based on the similarity between songs or artists and for music browsing based on visualized songs, it is still difficult to explore unfamiliar content by flexibly combining multiple musical aspects. Query-by-Blending overcomes this difficulty by representing each of the aspects as a latent vector representation (called a "flavor" in this paper) that is a distinctive quality felt to be characteristic of a given word/song/artist. By giving a lyric word as a query, for example, a user can find songs and artists whose flavors are similar to the flavor of the query word. Moreover, by giving a query combining (blending) lyric-word and song-audio flavors, the user can interactively explore unfamiliar content containing the blended flavor. This multi-aspect blending was achieved by constructing a novel vector space model into which all of the lyric words, song audio tracks, and artist IDs of a collection can be embedded. In our experiments, we embedded 14,505 lyric words, 433,936 songs, and 44,696 artists into the same shared vector space and found that the system can appropriately calculate similarities between different aspects and blend flavors to find related lyric words, songs, and artists.

1. INTRODUCTION

Given a huge collection of musical pieces such as those provided by online music services, conventional music access based on bibliographic metadata like song titles and artist names has not been sufficient. The Music Information Retrieval (MIR) community, therefore, has covered various types of music retrieval and exploration. When a listener wants to listen to familiar musical pieces, a popular approach is content-based music retrieval [4, 15, 26, 31] based on music similarity. When a query is entered by humming [5, 8, 12, 24, 29, 33], for example, similarities between the query and melody lines in a database are computed to show its title. When a query is given by a musical piece [7, 17, 19, 25, 28], a ranked list of similar musical pieces is shown. Music retrieval based on various kinds of metadata [3, 6, 14, 27] is also proposed. Although music would have multiple aspects, previous approaches typically assume a single aspect as a query. Moreover, it is necessary for users to conceive appropriate queries, which is sometimes not easy.

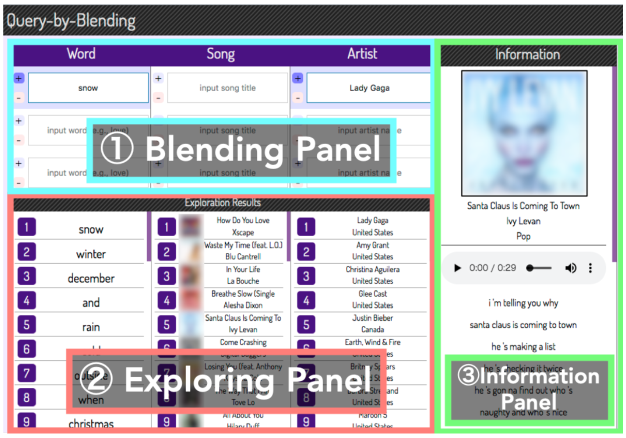


Figure 1. Query-by-Blending interface that consists of blending, exploring, and information panels.

When a user wants to discover unfamiliar musical pieces, an approach of music exploration is important. Music exploration systems typically provide interfaces that visualize a music collection by embedding musical pieces or artists into a 2D or 3D space and let users explore the collection to find favorite pieces [11, 21, 25, 30] or artists [1, 25, 27, 32]. However, it is difficult for previous music exploration systems to take different aspects of music into account. Another approach is to help users flexibly conceive a variety of queries for music discovery [9]. Such assistance, however, has not been investigated much in the MIR community.

We therefore propose a novel music exploration system, *Query-by-Blending*, that can embed three musical aspects – lyric word (word in lyrics), song audio (audio signal of a song), and artist (represented as artist ID) – into a unified high-dimensional latent vector space and enable users to find unfamiliar but interesting music content by flexibly combining those aspects (Figure 1). Query-by-Blending

Attribution: Kento Watanabe, Masataka Goto. "Query-by-Blending: A Music Exploration System Blending Latent Vector Representations of Lyric Word, Song Audio, and Artist", 20th International Society for Music Information Retrieval Conference, Delft, The Netherlands, 2019.

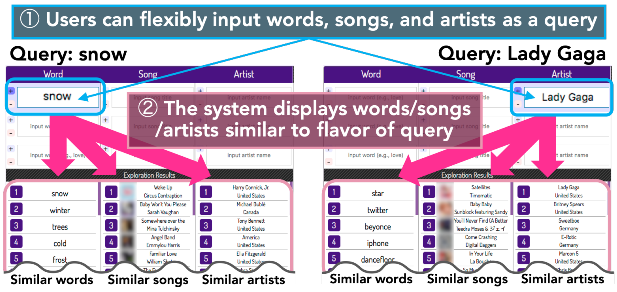


Figure 2. Query-by-Blending enables the exploration of three musical aspects: lyric word, song audio, and artist.

uses a "flavor" and "blending flavors" metaphor to help users combine different aspects to flexibly conceive a variety of queries. The terms "lyric-word flavor", "song-audio flavor", and "artist flavor" denote latent vector representations that are distinctive qualities felt to be characteristic of a given word, song, and artist. Each of the three kinds of flavors can be used as a query to retrieve and display three kinds of ranked lists: related lyric words, titles of songs having related song audio, and related artist names.

By giving a favorite artist as a query, for example, a user can not only listen to various songs containing its artist flavor but also see lists of lyric words and artists containing its artist flavor. Since the retrieved songs are not necessarily songs by the query artist, a user can explore a variety of unfamiliar music content. Since all three of the musical aspects are embedded into the same latent vector space as flavors, a user can add another flavor to "blend flavors" (i.e., give a query combining multiple musical aspects). Adding a lyric-word flavor, for example, causes the displayed lists to be interactively updated to give every musical content containing that flavor as well as the previous flavor a higher rank. Query-by-Blending can thus provide novel interactive, incremental, and iterative exploration experiences based on multiple musical aspects.

In implementing Query-by-Blending, it is difficult to calculate similarities among the three musical aspects because there are no large-scale annotations for supervised learning of such similarities. To overcome this difficulty, we propose a method of constructing a latent vector space model that can be trained with unsupervised learning under the assumption that a lyric word, song audio, and artist sampled from the same song tend to have similar meanings and are mapped to positions close to each other in the unified vector space. This method is based on multi-task learning, which has been used successfully across various applications of neural networks, and uses a vector model that can learn shared representations of music content by separately training each aspect of a large music collection (one including 14,505 words, 433,936 songs, and 44,696 artists).

2. QUERY-BY-BLENDING

Query-by-Blending enables a user to iteratively issue a query of any combination of lyric words, song titles, and artist names and obtain ranked lists of lyric words, song titles, and artist names that are similar to the query. The interface of Query-by-Blending is shown in Figure 1 and consists of (1) a blending panel for issuing the query, (2) an exploring panel displaying the retrieved ranked lists and allowing the user to select a song, and (3) an information panel displaying the title, artist name, and lyrics of the selected song and allowing the user to play back a short excerpt of the song for trial listening.

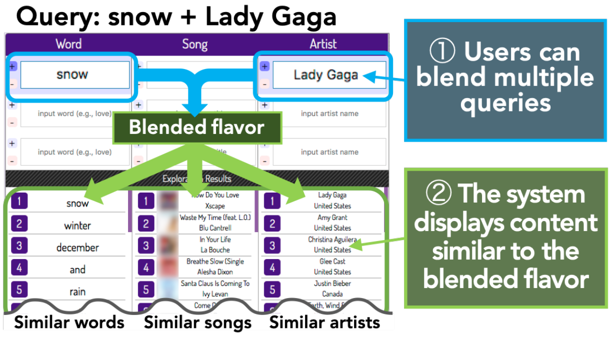


Figure 3. By blending multiple flavors, Query-by-Blending can display music content similar to the blended flavor. The blended flavor "snow + Lady Gaga" is similar to Christmas, Pop, and Soul songs/artists.

2.1 Exploring Three Musical Aspects

When a user enters a lyric word, song title, or artist name as a query on the blending panel, the exploring panel displays the retrieved lists of lyric words, song audio tracks, and artists whose flavors (latent vector representations) are similar to the flavor of the query. The lists are sorted in the order of the similarity. Figure 2 shows two screenshots of the interface. As shown in the left side of Figure 2, for example, when a user types "snow" into the lyric word text field on the blending panel, the exploring panel displays the lyric word "winter", "Angel Band" (the title of a song containing the word "snow"), and "Harry Connick, Jr." (who released the Christmas song album "When My Heart Finds Christmas", which can be considered to be related to "snow"). On the other hand, as shown in the right side of Figure 2, when the user types "Lady Gaga" into the artist text field on the blending panel, the exploring panel displays music content with "Lady Gaga" flavor – lyric words "star" and "dance floor", songs of Pop and Dance music sung by female singers, and an American female singer "Britney Spears". The user can then click one of the displayed songs to listen to its excerpt. These results and their music excerpts are available at a web page (`https://kentow.github.io/qbb/`). Query-by-Blending thus enables the user to flexibly issue a query to explore music content that is similar to the query.

2.2 Blending and Subtracting Flavors

Although issuing a single-flavor query with one of the three aspects (flavors) has already been useful, the blending panel further enables a user to issue a query blending multiple flavors. As shown in Figure 3, for example, when the user

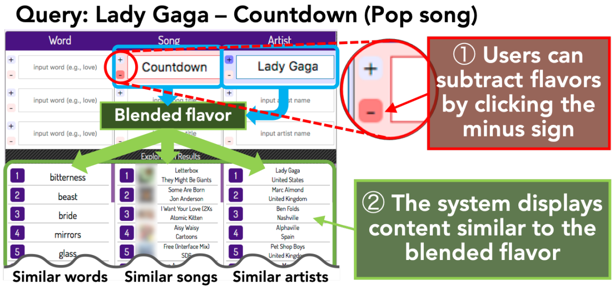

Query: Lady Gaga – Countdown (Pop song)
Word
Song
Artist
input word (e.g., love)
Countdown
Lady Gaga
input artist name
Blended flavor
Exploration Results
1 bitterness
2 beast
3 bride
4 mirrors
5 glass
1 Letterbox They Might Be Giants
2 Some Are Born Jon Anderson
3 I Want Your Love (2Xs Atomic Kitten
4 Aisy Waisy Cartoons
5 Free (Interface Mix) SD6
1 Lady Gaga United States
2 Marc Almond United Kingdom
3 Ben Folds Nashville
4 Alphaville Spain
5 Pet Shop Boys United Kingdom
Similar words
Similar songs
Similar artists
① Users can subtract flavors by clicking the minus sign
② The system displays content similar to the blended flavor

Figure 4. Query-by-Blending allows the user to subtract flavors from others. The flavor “Lady Gaga – Countdown (Pop song)” is similar to Alternative and Rock songs/artists.

types both “snow” and “Lady Gaga” into the two text fields on the blending panel, the retrieved ranked lists are updated to be more similar to the blended flavor of both of them. The exploring panel displays a Christmas song sung by the female Soul singer “Ivy Levan”. Then the user can feel free to iteratively add more flavors.

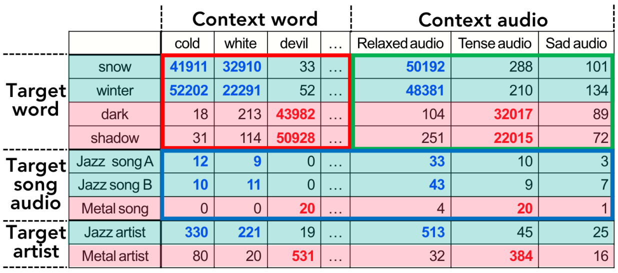
Moreover, when a user issues a query and finds some of the listed songs or artists unappealing, the blending panel enables the user to subtract a flavor related to them from the current query. As shown in Figure 4, when the user subtracts a Pop song “Countdown” having a “Pop song” flavor from the original query of “Lady Gaga” (by clicking the minus sign (−) located beside the text field of the song “Countdown”), the exploring panel updates the lists to include Alternative and Rock music songs (e.g., “Letterbox”) and artists (e.g., “Ben Folds”) that are similar to the flavor “Lady Gaga − Countdown”. The user can also subtract some flavors after blending multiple flavors.

When a user issues a query and finds unfamiliar songs or artists interesting after trial listening on the information panel, the user can add (blend) some of them to the current query to update the retrieved lists. Query-by-Blending thus provides novel exploration experiences, such as being able to incrementally conceive a variety of queries including both familiar and unfamiliar songs and artists. It enables users to flexibly update their queries by blending and subtracting various flavors to explore unfamiliar but interesting music content interactively in a trial-and-error manner.

3. IMPLEMENTATION

We implemented Query-by-Blending by developing a novel unsupervised method of constructing the unified latent vector space in which similar aspects are located nearby. Since similarities related to the three musical aspects are not annotated in a typical music collection, we leveraged the *Distributional Hypothesis* [10] that is well-known in the field of Natural Language Processing and has successfully been used in *word2vec* [23].

To explain the mechanism of capturing similarities, we first focus on the similarity between two lyric words without using audio and artist aspects. According to the Distributional Hypothesis, we assume that *words that occur in similar contexts tend to have similar topics*. As an example, consider the following two lyrics:



		Context word				Context audio		
		cold	white	devil	…	Relaxed audio	Tense audio	Sad audio
Target word	snow	41911	32910	33	…	50192	288	101
	winter	52202	22291	52	…	48381	210	134
	dark	18	213	43982	…	104	32017	89
	shadow	31	114	50928	…	251	22015	72
Target song audio	Jazz song A	12	9	0	…	33	10	3
	Jazz song B	10	11	0	…	43	9	7
	Metal song	0	0	20	…	4	20	1
Target artist	Jazz artist	330	221	19	…	513	45	25
	Metal artist	80	20	531	…	32	384	16

Figure 5. The Distributional Hypothesis for multiple aspects.

> “**Snow** *white side street of cold NewYork City.*”
>
> *“But here in the white of a cold* **winter** *night, ...”*

Since “snow” and “winter” each co-occur with “cold” and “white” (i.e., the same context), we can assume that “snow” and “winter” have a similar topic. In this paper, we define a *context word* (e.g., “cold” or “white”) to be a word co-occurring with a *target word* (e.g., “snow” or “winter”) in the same song. This mechanism is expressed by a *co-occurrence matrix* where each row corresponds to a target word and columns give the context words (the red frame in Figure 5). In this matrix, each cell contains the frequency of co-occurrence of the target word and the context word in all the songs. In Figure 5, the target words “snow” and “winter” have high co-occurrence with the context words “cold” and “white” but low co-occurrence with “devil”. Other target words “dark” and “shadow” frequently co-occur with the context word “devil”. By calculating the co-occurrence matrix, we can estimate that “snow” and “winter” have similar topics and that “dark” and “shadow” have similar topics. We call each row of this matrix a *target word vector* and calculate the similarity between target words as the distance between their target word vectors.

We then extend this co-occurrence matrix to *context audio tracks* so that we can utilize audio signals for capturing the similarity between target words. Each context audio is a short fragment of song audio and is represented as an *audio-word* defined in Section 3.2. In the green frame of Figure 5, the target words “snow” and “winter” tend to co-occur with some short fragments of song audio and are considered similar, but other target words “dark” and “shadow” do not co-occur with those short fragments. Both the context words and context audio tracks can thus be used to capture the similarity and are included in each *target word vector*. As with the similarity between target words, we assume that *songs that occur in similar contexts tend to have similar topics*. In the blue frame of Figure 5, we call each row of a target song audio in the co-occurrence matrix a *target audio vector*. Each cell of a target audio vector contains the number of occurrences of a context word in lyrics of the song corresponding to the target song audio, or contains the number of occurrences of a context audio in the target song audio. In Figure 5, the target vectors of “snow”, “Jazz song audio”, and “Jazz artist” are close to context vectors of "cold" and "white"; thus these multiple aspects can be located near each other. Moreover, we can also naturally calculate the similarity between any target word vector and any target audio vector as the distance between them since those vectors are represented in the same matrix.

Finally, in the same way, we can extend the co-occurrence matrix to *target artists*. In the bottom part of Figure 5, we call each row of a target artist in the co-occurrence matrix a *target artist vector*. Each cell of a target artist vector contains the number of occurrences of a context word in lyrics of all songs by its artist, or contains the number of occurrences of a context audio in all song audio tracks by its artist. By extending the Distributional Hypothesis, we can calculate similarities related to the three aspects.

3.1 Dataset

To calculate the above similarities, we made a dataset containing 433,936 songs by 44,696 artists. The dataset item for each song consists of a text file of English lyrics provided by a lyrics distribution company, an audio file of a short music excerpt (30 sec, 44.1kHz) available for trial listening on a music service, and an artist ID. Here, each text file contains all sentences of the lyrics of a song. We extracted 14,505 frequent lyric words from all the text files and did not use words that appeared less than 100 times.

3.2 Creating Audio-word Representation

To represent a short fragment of song audio for a context audio, we use a discrete symbol called an *audio-word*. The audio-word can be obtained by a *bag of audio-words* (BoAW) model [18] as follows. (1) Each music excerpt is downsampled to 22,050 Hz. (2) We use *LibROSA*, a python package for music and audio analysis, to extract 20-dimensional mel-frequency cepstral coefficients (MFCCs) with the FFT size of 2048 samples and the hop size of 512 samples. This result is represented as an MFCC matrix (20 × 1280). (3) The MFCC matrix is divided into 128 submatrices (20 × 10) without overlap. (4) Each submatrix including 10 frames of MFCCs is flattened into a 200-dimensional vector that represents local temporal dynamics of MFCCs. (5) We use the *k-means++* algorithm [2] to group all 200-dimensional vectors of all 433,936 songs into 3,000 clusters. (6) Each cluster is regarded as a discrete audio-word. We thus obtained 3,000 audio-words.

3.3 Vector Space Model for Multiple Musical Aspects

Since the co-occurrence matrix is huge and extremely sparse, it is too computationally expensive to deal with. To overcome this problem, our latent vector space model uses a neural network to reduce the huge matrix to a dense matrix as *word2vec* [23] also does.

The structure of the model is illustrated in Figure 6. Let w_t denote the target word, let aw_m denote the audio-word, let a denote the artist ID, and let c denote the context consisting of the context word and context audio-word. To obtain a D-dimensional latent vector space/representation after dimension reduction, we define an embedding function $\boldsymbol{v_w}(\cdot)$ that maps the target word to a D-dimensional vector and define an embedding function $\boldsymbol{u}(\cdot)$ that maps the context word/audio-word to a D-dimensional vector.

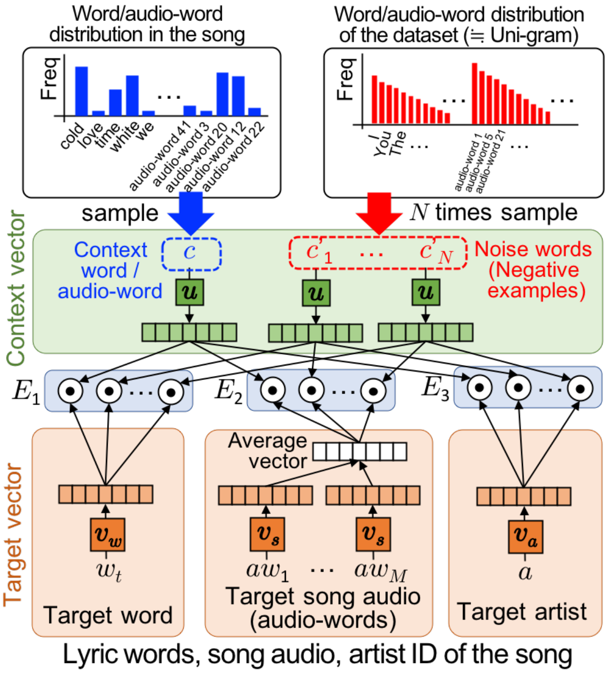


Figure 6. Multiple musical aspect vector space model.

We formulate this as an optimization problem that minimizes the distance between $\boldsymbol{v_w}(w_t)$, the target vector of a target word w_t in a song, and $\boldsymbol{u}(c)$, the context vector of another context word or a context audio-word c randomly sampled from the same song. By iterating this sampling and minimization for every target word, the model captures the similarity between target words. In Figure 5, for example, "snow" and "winter" are frequently sampled for the target words, and "cold" and "white" are frequently sampled for the context words. In other words, the vectors of the target words "snow" and "winter" are close to both of the vectors of context words "cold" and "white" in the embedded vector space. Thus, these target word vectors can be located near each other. In the training phase to minimize the above distance, we maximize $\boldsymbol{u}(c)^{\mathsf{T}} \cdot \boldsymbol{v_w}(w_t)$, the dot product of the context word/audio-word vector $\boldsymbol{u}(c)$ and the target word vector $\boldsymbol{v}(w_t)$ in the lyrics of a song. This maximization can be done by optimizing parameters of $\boldsymbol{u}(\cdot)$ and $\boldsymbol{v_w}(\cdot)$.

To accelerate training, we not only maximize the dot product of co-occurring w and c but also minimize the dot product of w and c', where c' is a noise word. This technique is called *Negative Sampling* and is known to be useful in training word2vec [23]. We define and minimize the objective function E_1 to maximize $\boldsymbol{u}(c)^{\mathsf{T}} \cdot \boldsymbol{v_w}(w_t)$ and minimize $\boldsymbol{u}(c')^{\mathsf{T}} \cdot \boldsymbol{v_w}(w_t)$:

$$E_1 = -\log\sigma\Big(\boldsymbol{u}(c)^{\mathsf{T}} \cdot \boldsymbol{v_w}(w_t)\Big) - \sum_{n=1}^{N} \log\sigma\Big(-\boldsymbol{u}(c'_n)^{\mathsf{T}} \cdot \boldsymbol{v_w}(w_t)\Big), \quad (1)$$

where $\sigma(\cdot)$ is a sigmoid function. N is the number of negative words/audio-words $c'_n (1 \leq n \leq N)$ sampled from

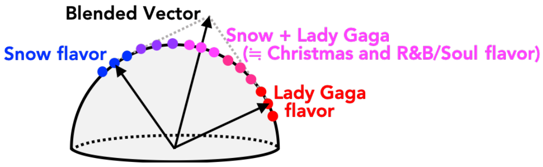


Figure 7. Normalized multiple aspect vector on hypersphere. Each circle denotes an embedded aspect. The cosine similarity between similar aspects is large.

the following noise distribution:

$$P(c'_n) = \#(c'_n)^{0.75} / \textstyle\sum_{c' \in V} (\#(c')^{0.75}), \tag{2}$$

where V is the word and audio-word vocabulary and $\#(c')$ is the global frequency of a word and audio-word c' in the whole dataset. This is for maximizing the distance between the target word and common words, such as "I" and "You", that occur frequently in the dataset.

In our current implementation, we set the vector dimension D to 400, the number of samplings for context words and context audio-words in each song to 400, and the number of negative samplings to 10. The objective function E_1 is optimized using stochastic gradient descent with a learning rate of 0.025, and training was run for 5 epochs.

We can also use the same idea to handle the similarities related to song audio tracks and artists. For song audio tracks, we define and minimize the loss function E_2:

$$\begin{aligned} E_2 = -\log\sigma\Big(\boldsymbol{u}(c)^\top \cdot \tfrac{1}{M} \textstyle\sum_{m=1}^{M} \boldsymbol{v_s}(aw_m)\Big) \\ - \textstyle\sum_{n=1}^{N} \log\sigma\Big(-\boldsymbol{u}(c'_n)^\top \cdot \tfrac{1}{M} \textstyle\sum_{m=1}^{M} \boldsymbol{v_s}(aw_m)\Big), \end{aligned} \tag{3}$$

where M is the number of audio-words in the song, $\boldsymbol{v_s}(\cdot)$ is an embedding function that maps the one-hot representation of every audio-word in the song to a D-dimensional vector, and the vector of the target song audio is represented by averaging all the audio-word vectors $\boldsymbol{v_s}(aw_m)$ in the song. For artists, we define and minimize the loss function E_3:

$$E_3 = -\log\sigma\Big(\boldsymbol{u}(c)^\top \cdot \boldsymbol{v_a}(a)\Big) - \sum_{n=1}^{N} \log\sigma\Big(-\boldsymbol{u}(c'_n)^\top \cdot \boldsymbol{v_a}(a)\Big). \tag{4}$$

where $\boldsymbol{v_a}(\cdot)$ is an embedding function that maps the one-hot representation of the artist ID to a D-dimensional vector. In the training phase, these objective functions E_2 and E_3 are optimized with the same settings as E_1.

The proposed model can capture similarities between the *same aspects* by minimizing E_1, E_2, and E_3. In addition, iterative optimization of the three objective functions enables training of the similarity between *multiple aspects* because these three objective functions share the embedding function for the context vector $\boldsymbol{u}(\cdot)$. In Figure 5, the target vectors of "snow", "Jazz song audio", and "Jazz artist" get closer to the context vectors such as $\boldsymbol{u}$(white); thus these multiple aspects can be located near each other.

3.4 Similarity Calculation

When calculating similarity, we use vectors obtained using $\boldsymbol{v_w}(\cdot)$, $\boldsymbol{v_s}(\cdot)$, and $\boldsymbol{v_a}(\cdot)$ without $\boldsymbol{u}(\cdot)$. We can use the cosine similarity as a measure of the similarity of two vectors. According to Levy et al. [16], multiple aspect vectors are normalized to unit length before they are used for similarity calculation, making cosine similarity and dot product equivalent. By this constraint, all the words, songs, and artists are located on a hypersphere and the system finds music content considering two flavors by calculating the content close to the blended vector on the hypersphere (Figure 7).

4. QUALITATIVE ANALYSIS

We investigated whether the trained vector space model appropriately captures the similarity between multiple musical aspects. Table 1 shows the five most similar lyric words, song audio tracks [1] , and artists – as well as their cosine similarities – that were obtained when we issued three different queries written at the top.

We can see in Table 1 that the query word "death" is similar to the words "blood" and "dead", which is reasonable since those words are often used in metal songs. Moreover, "death" is similar to song audio tracks having aggressive sounds using electric guitars and to Heavy Metal artists such as "Savatage". Metal songs like "Neuro Osmosis" were found even though their lyrics do not include the word "death".

Table 1 also shows that the query song "Amazing Grace" is similar to clean words such as "meadow" and "lullaby" and to relaxation songs such as New Age and Holiday music. Interestingly, the query artist "Michael Jackson" is similar to the artist "Janet Jackson" who is his sister even though the model does not know that they are siblings. This query is also similar to rhythmic songs by other artists.

These results indicate that the multi-aspect vector space model makes it possible to find related lyric words, songs, and artists that are hard to find otherwise. Furthermore, we tried to issue various queries by blending and subtracting flavors of multiple aspects and confirmed the usefulness of this blending and subtracting. Some results were illustrated in Section 2.2. Another interesting result is that the blended flavor "Stevie Wonder" + "snow" − "spring" is similar to "California Christmas".

5. QUANTITATIVE EVALUATION

Since ground-truth annotations of similarities between different aspects do not exist, it is not easy to quantitatively evaluate the model in general, but we tried to evaluate the effectiveness of blending latent vector representations by comparing content-based distributions of retrieved results. In Table 1, for example, the results retrieved for the word "death" and the song "Amazing Grace" are contrasting and have different impressions, which means that the distance between their distributions is large. If we issue a query blending them and it works effectively, we expect to see somewhat intermediate results between them, which means that the distance between one of the above distributions and the new distribution after blending becomes smaller.

[1] The song audio tracks used in this table can be listened to at our demo page (`https://kentow.github.io/qbb/`).

	Input word: death		Input song: Amazing Grace		Input artist: Michael Jackson	
Similar words	blood	0.70	may	0.37	hypnotized	0.41
	dead	0.65	sadly	0.35	maurice	0.40
	flesh	0.65	gentle	0.34	babygirl	0.39
	reborn	0.63	meadow	0.34	confused	0.39
	mortal	0.62	lullaby	0.34	emotions	0.39
Similar songs	Burning Sermon (Rock)	0.49	I Wish I Was In England (World)	0.85	It's a Man's Man's Man's World (Pop)	0.50
	Neuro Osmosis (Metal)	0.48	North Country Maid (Pop)	0.84	Tight (Gospel)	0.49
	Scraping the Barrel (Metal)	0.48	Mary, Did You Know (Holiday)	0.83	Play with Bootsy (Rock)	0.48
	Coins Upon the Eyes (Metal)	0.47	No Turning Back (Alternative)	0.83	Unspeakable (Pop)	0.48
	The Harlot Ov the Saints (Rock)	0.47	Beloved (NewAge)	0.83	Dead Heat (Pop)	0.48
Similar artists	Gary Numan	0.40	Barbra Streisand	0.40	Janet Jackson	0.61
	L'Âme Immortelle	0.39	Ella Fitzgerald	0.39	Sarah Connor	0.52
	Cowboy Junkies	0.38	Linda Ronstadt	0.38	Luther Vandross	0.52
	Don Moen	0.38	Debby Boone	0.37	Kylie Minogue	0.52
	Savatage	0.37	Nana Mouskouri	0.35	Faith Evans	0.51

Table 1. The most similar words, songs, and artists obtained by Query-by-Blending. The genre tags are shown in parentheses.

We therefore quantitatively examined how much this distance decreases after blending as follows. (1) A word query w and a song query s for evaluation are sampled from the dataset. (2) We get the 100 most similar songs retrieved by the word query w and compute ψ_w that denotes a content-based distribution of all lyric words and audio-words (i.e., a histogram of them) appearing in those 100 songs. (3) We get the 100 most similar songs retrieved for the song query s and compute ψ_s that denotes a content-based distribution obtained from those 100 songs in the same way. (4) We then get the 100 most similar songs retrieved for the query blending w and s and compute ψ_{w+s} that denotes a content-based distribution obtained from those 100 songs in the same way. (5) We calculate the Jensen-Shannon (JS) divergence between every pair of distributions: (ψ_w, ψ_s), (ψ_w, ψ_{w+s}), and (ψ_s, ψ_{w+s}). If the JS divergences of (ψ_w, ψ_{w+s}) and (ψ_s, ψ_{w+s}) are smaller than that of (ψ_w, ψ_s), we can confirm that our query blending lyric word and song audio works effectively. In addition, we replace the song s with the artist a and repeat the above procedure.

[Experimental Setup] For the above step (1), we used 1,000 word queries (most frequent nouns, verbs, adjectives, and adverbs), 1,000 song queries, and 1,000 artist queries. For the step (4), we made one million word-song pairs and one million word-artist pairs for the blended queries.

[Results] Table 2 shows JS divergences that are averaged over pairs. We can see that, as expected, the JS divergences of (ψ_w, ψ_{w+s}) and (ψ_s, ψ_{w+s}) are smaller than the JS divergence of (ψ_w, ψ_s). The same applies to (ψ_w, ψ_a). We thus confirmed that our blended queries work effectively with regard to content-based distributions of retrieved results.

Distributions		JS divergence	Distributions		JS divergence
ψ_w	ψ_s	0.261	ψ_w	ψ_a	0.224
ψ_w	ψ_{w+s}	**0.057**	ψ_w	ψ_{w+a}	**0.147**
ψ_s	ψ_{w+s}	**0.227**	ψ_a	ψ_{w+a}	**0.119**

Table 2. Quantitative evaluation of blending flavors.

6. RELATED WORK

Several studies have dealt with the similarity between different aspects of music. McFee and Lanckriet [22], for example, developed a hypergraph of song nodes whose edges capture multi-aspect relationships. Although it can be used to calculate similarities between songs while considering multiple aspects, it does not deal with similarities between the multiple musical aspects.

Some studies embedded musical aspects into a high-dimensional vector space in an unsupervised manner. Weston et al. [35] proposed a model by which musical audio signals and artist tags are embedded assuming that songs created by the same artist are correlated. Wang et al. [34] modeled the relationships between songs by using the same architecture as word2vec under the assumption that songs played by the same listener are similar. There are also studies that embedded vectorized audio-words and social tags by using the singular value decomposition (SVD) [13, 20]. All these studies shared with ours the motivation of embedding multiple aspects into a vector space but dealt only with audio signals and metadata without lyrics even though lyrics are an important element that conveys messages and emotions of music. To the best of our knowledge, there is no study in which lyric word, song audio, and artist ID are embedded into the same vector space.

7. CONCLUSION

In this paper we proposed a novel interface, Query-by-Blending, that enables users to find unfamiliar but interesting music content by flexibly combining (blending) three musical aspects: lyric word, song audio, and artist. Our contributions are summarized as follows: (1) Query-by-Blending is the first interface that lets users iteratively issue various queries by blending and subtracting multiple musical aspects. (2) We developed the novel embedding method of constructing the unified latent multi-aspect vector space by using unsupervised learning. (3) We demonstrated that our vector space model captures the similarities between multiple aspects. We plan to conduct a user study evaluating the Query-by-Blending interface. We also plan to extend our model to blend other aspects, such as genre tags and album cover images.

8. ACKNOWLEDGMENTS

The authors appreciate SyncPower Corporation for providing lyrics data. This work was supported in part by JST ACCEL Grant Number JPMJAC1602, Japan.

9. REFERENCES

[1] Alo Allik, Florian Thalmann, and Mark B. Sandler. Musiclynx: Exploring music through artist similarity graphs. In *Proceedings of the Web Conference 2018 (WWW)*, pages 167–170, 2018.

[2] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM SIAM symposium on Discrete algorithms*, pages 1027–1035, 2007.

[3] David Bretherton, Daniel A. Smith, Monica M. C. Schraefel, Richard Polfreman, Mark Everist, Jeanice Brooks, and Joe Lambert. Integrating musicology's heterogeneous data sources for better exploration. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 27–32, 2009.

[4] Michael Casey, Remco Veltkamp, Masataka Goto, Marc Leman, Christophe Rhodes, and Malcolm Slaney. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4):668–696, 2008.

[5] Roger B. Dannenberg and Ning Hu. Understanding search performance in query-by-humming systems. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, 2004.

[6] Daniel P. W. Ellis, Brian Whitman, Adam Berenzweig, and Steve Lawrence. The quest for ground truth in musical artist similarity. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, 2002.

[7] Jonathan Foote, Matthew L. Cooper, and Unjung Nam. Audio retrieval by rhythmic similarity. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, pages 265–266, 2002.

[8] Asif Ghias, Jonathan Logan, David Chamberlin, and Brian C. Smith. Query by humming: Musical information retrieval in an audio database. In *Proceedings of the 3rd ACM International Conference on Multimedia (ACM Multimedia)*, pages 231–236, 1995.

[9] Masataka Goto and Takayuki Goto. Musicream: Integrated music-listening interface for active, flexible, and unexpected encounters with musical pieces. *Journal of Information Processing*, 17:292–305, 2009.

[10] Zellig S Harris. Distributional structure. *Word*, 10(2–3):146–162, 1954.

[11] Carles Fernandes Julià and Sergi Jordà. Songexplorer: A tabletop application for exploring large collections of songs. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 675–680, 2009.

[12] Tetsuya Kageyama, Kazuhiro Mochizuki, and Yosuke Takashima. Melody retrieval with humming. In *Proceedings of the 1993 International Computer Music Conference (ICMC)*, pages 349–351, 1993.

[13] Giannis Karamanolakis, Elias Iosif, Athanasia Zlatintsi, Aggelos Pikrakis, and Alexandros 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 (Interspeech)*, pages 3658–3662, 2016.

[14] Joon Hee Kim, Brian Tomasik, and Douglas Turnbull. Using artist similarity to propagate semantic information. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 375–380, 2009.

[15] Peter Knees, Tim Pohle, Markus Schedl, and Gerhard Widmer. A music search engine built upon audio-based and web-based similarity measures. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 447–454, 2007.

[16] Omer Levy, Yoav Goldberg, and Ido Dagan. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics (TACL)*, 3:211–225, 2015.

[17] Tao Li and Mitsunori Ogihara. Content-based music similarity search and emotion detection. In *Proceedings of the 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 705–708, 2004.

[18] Yang Liu, Wan-Lei Zhao, Chong-Wah Ngo, Chang-Sheng Xu, and Han-Qing Lu. Coherent bag-of audio words model for efficient large-scale video copy detection. In *Proceedings of the ACM International Conference on Image and Video Retrieval (ACM CIVR)*, pages 89–96, 2010.

[19] Beth Logan and Ariel Salomon. A music similarity function based on signal analysis. In *Proceedings of the 2001 IEEE International Conference on Multimedia and Expo (ICME)*, 2001.

[20] Alessandro Lopopolo and Emiel van Miltenburg. Sound-based distributional models. In *Proceedings of the 11th International Conference on Computational Semantics (IWCS)*, pages 70–75, 2015.

[21] Dominik Lübbers and Matthias Jarke. Adaptive multimodal exploration of music collections. In *Proceedings*

of the 10th International Society for Music Information Retrieval Conference (ISMIR), pages 195–200, 2009.

[22] Brian McFee and Gert RG Lanckriet. Hypergraph models of playlist dialects. In *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, pages 343–348, 2012.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th Annual Conference on Neural Information Processing Systems*, pages 3111–3119, 2013.

[24] Emilio Molina, Lorenzo J. Tardón, Isabel Barbancho, and Ana M. Barbancho. The importance of F0 tracking in query-by-singing-humming. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, pages 277–282, 2014.

[25] Elias Pampalk, Simon Dixon, and Gerhard Widmer. Exploring music collections by browsing different views. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, 2003.

[26] Elias Pampalk and Masataka Goto. MusicRainbow: A new user interface to discover artists using audio-based similarity and web-based labeling. In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, pages 367–370, 2006.

[27] Markus Schedl, Peter Knees, and Gerhard Widmer. Discovering and visualizing prototypical artists by web-based co-occurrence analysis. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 21–28, 2005.

[28] Malcolm Slaney and William White. Similarity based on rating data. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 479–484, 2007.

[29] Tomonari Sonoda, Masataka Goto, and Yoichi Muraoka. A www-based melody retrieval system. In *Proceedings of the 1998 International Computer Music Conference, (ICMC)*, 1998.

[30] Sebastian Stober and Andreas Nürnberger. Musicgalaxy – an adaptive user-interface for exploratory music retrieval. In *Proceedings of 7th Sound and Music Computing Conference (SMC)*, pages 23–30, 2011.

[31] Kosetsu Tsukuda, Keisuke Ishida, and Masataka 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 (ISMIR)*, pages 544–551, 2017.

[32] Fabio Vignoli, i Rob van Gulik, and Huub van de Wetering. Mapping music in the palm of your hand, explore and discover your collection. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, 2004.

[33] Chung-Che Wang, Jyh-Shing Roger Jang, and Wennen 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, (ISMIR)*, pages 45–50, 2010.

[34] Dongjing Wang, Shuiguang Deng, Xin Zhang, and Guandong Xu. Learning music embedding with metadata for context aware recommendation. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval (ACM ICMR)*, pages 249–253, 2016.

[35] Jason Weston, Samy Bengio, and Philippe Hamel. Multi-tasking with joint semantic spaces for large-scale music annotation and retrieval. *Journal of New Music Research*, 40(4):337–348, 2011.