TRANSDRUMS: A DRUM PATTERN TRANSFER SYSTEM PRESERVING GLOBAL PATTERN STRUCTURE

Shun Sawada[†] *Satoru Fukayama*[††] *Masataka Goto*[††] *Keiji Hirata*[†]

[†]Future University Hakodate, Japan
[††]National Institute of Advanced Industrial Science and Technology (AIST), Japan

ABSTRACT

This paper presents TransDrums, which is a system that transfers drum patterns from a drum-pattern-source song (D-song) to a base song (B-song) and synthesizes the audio with the substituted drum pattern. Typical drum parts consist of multiple drum patterns that are concatenated to form a structure by, for example, inserting fill-in patterns at structural boundaries. The previous system that replaced the drum parts was not able to form such a structure. Therefore, we propose TransDrums, which extracts and transfers multiple drum patterns to form the structure. It takes two songs as the input and extracts multiple typical drum patterns from each song. It then makes pairs of those patterns between B-song and D-song and replaces them using the counterpart drum patterns to synthesize audio with the altered drum pattern. To achieve the key idea of properly replacing a drum phrase in B-song with that in D-song, it is necessary to model the structure of the drum parts by analyzing the transition probabilities between the typical drum patterns. The appropriate pairs are determined so that the sum of the Jensen-Shannon divergence between the transition probabilities is minimized. Our experimental results show that TransDrums can generate audio to change by altering the drum patterns with the structure.

Index Terms— Music information retrieval, drum pattern, music structure, active music listening

1. INTRODUCTION

Conventional music listening was a passive experience that involved just playing music and enjoying it. Recently, research for active music listening [1] has been expanding. Several techniques have been proposed to satisfy the active requirements of listeners, such as listening to music while editing it to suit their taste. Nakamura proposed a timbre replacement system [2] that replaces the frequency characteristics of harmonic sounds and the timbres of drum sounds with those of another audio signal. Ono proposed a real-time equalizer of harmonic and percussive components in music signals that did not require any a priori knowledge of musical scores and included instruments [3]. Davies proposed AutoMashUpper [4] for making multi-song music mashups. Yoshii proposed Drumix [5] for controlling the rhythmic patterns of drums in audio signals in real time.

Following the previous systems, we present TransDrums, which is a system that transfers drum patterns from one song to another song and synthesizes the audio with the altered drum track while preserving the global pattern structure. Typical drum tracks consist of multiple drum patterns that are concatenated to form the structure by, for example, inserting fill-in patterns and changing patterns at structural boundaries. However, a previous system replacing the drum track with a four-bar drum pattern was not able to form such a structure [5]. In this paper, we propose a system that transfers drum patterns from one song to another song while preserving the global structure.

To achieve this goal, the role of each drum pattern, such as inserting fill-in patterns, should be detected, and each drum pattern should be substituted with the one that has the same role in the other song. The role of a drum pattern appears in the transition between the drum patterns. For example, drum patterns that have many self-transitions are likely to be a standard pattern occupying a large part in the song. Some of the fill-in patterns are also likely to have a transition from a particular drum pattern. In other words, the drum patterns that have the same role in each of the songs are considered to have a similar transition probability between drum patterns. To overcome difficulties finding appropriate drum pattern pairs to replace each other, we model the structure of the drum track by analyzing the transition probabilities between the typical drum patterns. Our experimental results show that TransDrums can generate audio to change by altering the drum track with the structure. Demonstrations are available at `https://shunsawada.github.io/transDrum/`.

2. DRUM PATTERN MAPPING ALGORITHM

Figure 1 shows the diagram for the entire processing. This system, which takes two songs (base-song (B-song) and drum-pattern-source song (D-song)) as inputs, transfers the drum patterns from D-song to B-song. The drum patterns in B-song are replaced by those in D-song according to a map preserving the global structure relating to the transition

between drum patterns in B-song. The problem with finding the appropriate relationship of each drum pattern is finding a mapping $M : \boldsymbol{B} \rightarrow \boldsymbol{D}$ from a set $\boldsymbol{B}$ of the drum patterns appearing in B-song to a set $\boldsymbol{D}$ of those appearing in D-song ((3) of Figure 1). The global structure of the drum pattern is modeled by analyzing the transition probabilities between typical drum patterns in B-song and D-song as a probability model. Figure 2 shows the processing procedure mapping that transfers drum patterns in D-song to B-song. First, this system performed clustering for each song using the bar-level features of a drum pattern. Second, it obtains a mapping between the clusters by clustering each of the two songs.

First, the bar-level features of the drum patterns are labeled by clustering. The number of clusters is N_{cluster}, and the label set is $\boldsymbol{C} = \{c_n\}_{n=1}^{N_{\text{cluster}}}$. The system converts the song to a bar-level label sequence $\boldsymbol{l} = \{l_n\}_{n=1}^{N_{\text{bar}}}$ using these labels. The label l_n belongs to class set $\boldsymbol{C}$ ($l_n \in \boldsymbol{C}$), and N_{bar} represents the number of bars in the song. The system calculates the transition probability of the label transition from l_{n-1} to l_n in label sequence $\boldsymbol{l} = \{l_n\}_{n=1}^{N_{\text{bar}}}$. An element t_{ij} of the transition probability matrix $\mathbf{T}_X$ ($N_{\text{cluster}} \times N_{\text{cluster}}$ matrix) of the transition between the drum patterns in X-song represents transition probability $P(c_j \mid c_i)$ of the transition from label c_i to label c_j. The n-th row of matrix $\mathbf{T}_X$ is the discrete probability distribution of c_n. A probability P_n is the n-th row of the $\mathbf{T}_B$ of B-song, and a probability Q_m is the m-th row of the $\mathbf{T}_D$ of D-song.

Second, to map the pairs of drum patterns preserving the global structure, we set a metric for the difference between structures. Since the structure here is represented with transition probability, the difference can be calculated based on Kullback-Leibler divergence. Assuming that the drum pattern with the same role in a song has a similar transition probability, the pair of drum patterns with the same role can be obtained through minimizing the sum of the divergences. Kullback-Leibler divergence $\mathrm{KL}(P_n \mid\mid Q_m)$ from a probability distribution Q_m to P_n is defined as:

$$\mathrm{KL}(P_n \mid\mid Q_m) = \sum_i P_n(i) \log \frac{P_n(i)}{Q_m(i)}. \quad (1)$$

In this study, we use Jensen-Shannon divergence $\mathrm{JS}(P_n \mid\mid Q_m)$ obtained by symmetrizing Kullback-Leibler divergence as follows:

$$\mathrm{JS}(P_n \mid\mid Q_m) = \frac{1}{2}(\mathrm{KL}(P_n \mid\mid M) + \mathrm{KL}(Q_m \mid\mid M)), \quad (2)$$

where $M = \frac{1}{2}(P_n + Q_m)$.

The system calculates the Jensen-Shannon divergence between all possible pairs of transition probabilities of D-song and B-song, and searched for a map minimizing the sum of all the mappings. By using drum pattern pairs obtained from this method, the label sequence of B-song ($\boldsymbol{l}_B$) is replaced by the labels of D-song using the mapping M, and a new label sequence $\boldsymbol{l}'_B$ is generated.

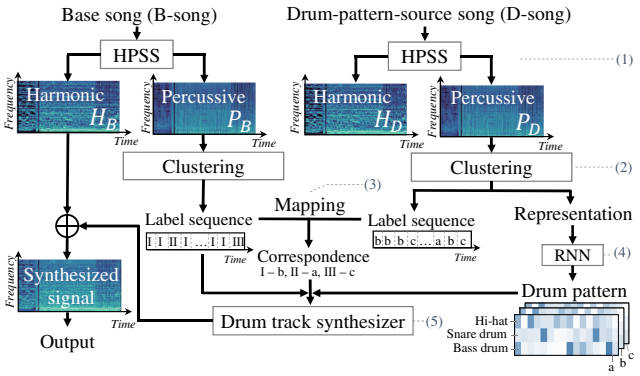

Fig. 1. Overview of our method.

3. IMPLEMENTATION OF TRANSDRUMS

3.1. Feature Extractor

In this section, we present the method to extract the bar-level drum pattern features of the input song. The input song consists of a non-harmonic-sound (percussive-sound) component and a harmonic-sound component. The drum pattern contains many of the non-harmonic-sound components. The input spectrogram is separated into the harmonic-sound and the non-harmonic-sound components ((1) of Figure 1), and we handle the non-harmonic-sound components in this study. We used onset saliency as feature of drum patterns and calculated the difference (increase) of the power spectrum for one frame against that for the previous frame in the non-harmonic-sound components. We reduced these dimensions of features to 16-dimensions corresponding to sixteenth-note grid.

First, the two songs input (B-song and D-song) were converted to spectrograms with a window size of 2048 samples and a hop size of 512 samples. Each spectrogram $\mathbf{S}$ was separated into the harmonic-sound components $\mathbf{H}$ and the non-harmonic-sound components $\mathbf{P}$ by using harmonic-percussive sound separation (HPSS). We used the HPSS of the `librosa.decompose.hpss` function implemented in librosa (Version 0.6.2) [6]. In the HPSS [7] used in this study, a technique [8] has been extended, and the parameters (margins) of the mask thresholds that separate components into harmonic and non-harmonic components are introduced. In our research, we set both (H and P) parameters (margins) to 2.0.

Second, we calculated the spectral flux of the non-harmonic-sound components $\mathbf{P}$ of each song. The spectral flux is used as a measure of temporal change of a spectrum. It was obtained by calculating the sum of the difference of the power spectrum densities between time t and time $t-1$ over all the frequency bins. The spectral flux $\mathbf{SF}_t$ at time t is discretized into the sixteenth-note level by calculating the sum of the m frames around the beat positions. The 16-dimensional vector $\boldsymbol{f} = \{f_i\}_{i=1}^{16}$ obtained this way was used as the feature of the drum pattern. We set $m = 8$ based on

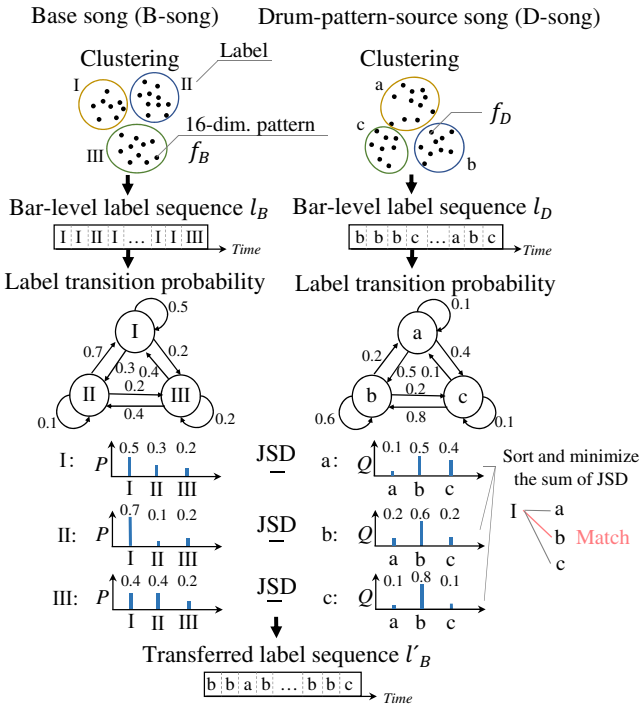

Fig. 2. Drum pattern mapping algorithm of TransDrums

a preliminary experiment. The drum pattern feature f_i was given as follows:

$$f_i = \sum_{\tau=-m}^{m} \mathbf{SF}_{u_i+\tau}, \tag{3}$$

where $\boldsymbol{u} = \{u_i\}_{i=1}^{16}$ represents the beat-time sequence of the sixteenth-note level.

The label sequence was generated by clustering these drum pattern features $\{f_i\}_{i=1}^{16}$ ((2) of Figure 1). We used k-means clustering with Lloyd's algorithm implemented as the Python library `scikit-learn` (Version 0.19.1) [9].

3.2. Drum Transcription and Synthesis

A drum pattern audio track of D-song was generated according to label sequence $\boldsymbol{l}'_B$ and mixed with the harmonic-sound components of B-song. First, we extracted the typical drum patterns from D-song. The drum pattern that was the nearest centroid of the cluster of D-song was extracted as a typical drum pattern. Second, we transcribed the typical drum pattern from audio $\mathbf{P}_D$ ((4) of Figure 1). Finally, we synthesized the drum track using the result of the drum transcription. Since many consonants of singing voices were contained in the non-harmonic-sound components, we decided not to use them to generate a drum track.

We used an algorithm based on a recurrent neural network (RNN) as the drum transcription. The RNN system has achieved the highest evaluation accuracies for polyphonic recordings and has been used in many drum transcription researches [10, 11, 12]. We used the automatic drum transcription library (ADTLib), which is the current state-of-the-art system for polyphonic music [12]. This system outputs the onset times of the bass drum, snare drum, and hi-hat cymbals. We used these onset times to generate the drum track. At each onset, we used the drum sound samples from RWC-MDB-I-2001 (snare drum: 431SD7N3, bass drum: 432BD2N3, and hi-hat cymbals: 432HHCC3) [13]. Finally, we mixed the drum track obtained in this way with the harmonic-sound components of B-song according to label sequence $\boldsymbol{l}'_B$ ((5) of Figure 1).

4. EVALUATION

4.1. Experimental Conditions

To evaluate whether the global drum pattern is recognized and the drum pattern is properly mapped, we conducted the following two experiments.

Experiment 1 : evaluating how much the fill-in patterns in B-song are reproduced by substituting the corresponding fill-in patterns in D-song.

Experiment 2 : evaluating the degree of correspondence between the transition label of the drum pattern in B-song and that in D-song.

We used songs in the RWC Music Database (RWC-MDB-P-2001) [14] to evaluate our system and used the label sequence, which has been made manually by two people who have more than three years' experience of drum performance, as a ground truth. An example of ground truth is shown in Figure 3. These experiments were conducted using pairs of songs with the same number of clusters of drum patterns (122 pairs). We used a beat structure of the AIST Annotation for the RWC Music Database [15] to divide a spectrogram by bar position. The sixteenth-note grid for calculating the spectral flux was obtained by dividing each beat equally into four segments.

First of all, we explain the symbols for defining the evaluation measure as follows. The label set of B-song represents $\boldsymbol{B} = \{b_n\}_{n=1}^{N_{\text{cluster}}}$, and that of D-song represents $\boldsymbol{D} = \{d_n\}_{n=1}^{N_{\text{cluster}}}$. The mapping obtained by the proposed method represents $M(b_i) = d_j$.

In Experiment 1, we defined the fill-in pattern mapping rate $F(M)$ as a measure to evaluate how much the fill-in patterns in two songs map to each other. $F(M)$ is defined as follows:

$$F(M) = \frac{\sum_i^{N_{\text{cluster}}} f(c_i) f(M(c_i))}{N_{\text{fill-in}}}, \tag{4}$$

where $N_{\text{fill-in}}$ represents the number of types of fill-in patterns contained in the song, and $f(c_i)$ represents the ground

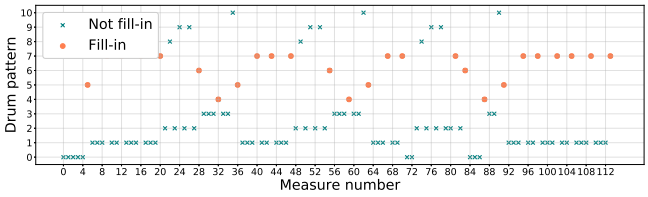

Fig. 3. Ground truth (RWC-MDB-P-2001 No. 1): horizontal axis represents bar number, and vertical axis represents label of drum patterns. Fill-in pattern positions are plotted as orange circles while other (not-fill-in) pattern positions are plotted as green crosses.

truth of the fill-in pattern positions. $f(c_i)$ takes the binary value $\{1, 0\}$, which may or may not be a fill-in pattern. $F(M)$ takes the maximum value 1 when all the fill-in patterns of the two songs map to each other ($0 \leq F(M) \leq 1$).

In Experiment 2, we defined the frequency consistency $E(M)$ as a measure to evaluate the degree of the correspondence of the mapping and the bigram frequency the drum pattern labels. For example, when highly frequent label sequences of B-song and D-song are "12" and "ab", respectively, according to the bigram frequency, the mapping is expected to be $M(1) = a,\ \ M(2) = b$. However, the mapping patterns could be inconsistent under the constraint that all patterns should have one-to-one mapping. $E(M)$ is defined as follows:

$$E(M) = \sum_{r=1}^{10} w(r) \frac{1}{2} (R_0(r) + R_1(r)). \tag{5}$$

Here, $R_i(r) = 1$ if x_{ri} in B-song matches x_{ri} in D-song where $\{x_{ri}\}_{i=0}^{1} = x_{r0}x_{r1}$ denotes the bigram with the r-th highest frequency, otherwise $R_i(r) = 0$. The function $w_x(r)$ is defined as:

$$w(r) = \frac{v_{x_{r0}x_{r1}}}{\sum_r v_{x_{r0}x_{r1}}}, \tag{6}$$

where $v_{x_{r0}x_{r1}}$ denotes the number of occurrences of bigram $x_{r0}x_{r1}$ in the B-song. The measure $E(M)$ indicates how invariant the order of the bigram frequencies is with respect to the mapping obtained by the proposed method ($0 \leq E(M) \leq 1$).

4.2. Experimental Results and Discussion

The top five and worst five of the results of $E(M)$ and $F(M)$ are shown in Table 1. For example, the first row of Table 1 (left) indicates that $E(M)$ is 0.96 and $F(M)$ is 1.0 when the drum pattern of RWC-MDB-P-2001 No. 17 (B-song) is replaced with that of RWC-MDB-P-2001 No. 15 (D-song). The average of $E(M)$ for all 122 pairs of the song is 0.37, which indicates the system could not create a full-match among all of the drum patterns. On the other hand, the average of

Table 1. Top 5 results (left) and worst 5 results (right) with regard to frequency consistency $E(M)$, and the corresponding fill-in pattern mapping rate $F(M)$.

Pairs	$E(M)$	$F(M)$	Pairs	$E(M)$	$F(M)$
17 - 15	.96	1.0	17 - 14	.01	.75
17 - 12	.90	1.0	8 - 18	.02	1.0
17 - 5	.89	.50	18 - 8	.02	.40
17 - 7	.89	.75	7 - 18	.02	1.0
15 - 17	.86	1.0	18 - 14	.03	.60

$F(M) = 0.73$ is relatively higher than $E(M)$, which shows that matching between fill-in patterns is easier for our system.

In the third row of Table 1 (left), the value of $F(M)$ was low compared to other cases. $N_{\text{fill-in}}$ of the RWC-MDB-P-2001 No. 17 is 4, whereas $N_{\text{fill-in}}$ of RWC-MDB-P-2001 No. 5 is 2. Therefore, all of the fill-in patterns existing in RWC-MDB-P-2001 No. 5 were mapped with those existing in RWC-MDB-P-2001 No. 17. Most of the song pairs having low $F(M)$ value are due to difference of $N_{\text{fill-in}}$. When the song pairs have similar $N_{\text{fill-in}}$, the system is able to replace the drum patterns preserving the global structure. Most of the song pairs having a low $E(M)$ value are due to the greatly different structures of songs. When the pairs of songs had a similar global drum-pattern structure, the mapping between patterns were found appropriately and the bigram patterns which were frequently observed in each song were consistent. However, since $N_{\text{fill-in}}$ and repeating patterns could be different depending on the songs, the system has the problem of mapping fill-in patterns to not-fill-in patterns because of the one-to-one mapping constraint.

5. CONCLUSION

We described TransDrums, which can transfer drum patterns in a song into another song while considering the global drum-pattern structure. To analyze the global structure for the drum pattern transfer, we proposed a novel approach of focusing on the transition probability between the drum patterns and mapping drum patterns by minimizing the distance between the transition probability distributions. Our experimental results show that the system was able to grasp the global structure of drum patterns. Future work includes improving the sound synthesis quality and the clustering of drum patterns.

Acknowledgement

This research was supported in part by JST ACCEL Grant Number JPMJAC1602 and JSPS KAKENHI Grant Number 16H01744.

6. REFERENCES

[1] M. Goto, "Active music listening interfaces based on signal processing," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007, pp. 1441–1444.

[2] T. Nakamura, H. Kameoka, K. Yoshii, and M. Goto, "Timbre replacement of harmonic and drum components for music audio signals," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 7470–7474.

[3] N. Ono, K. Miyamoto, H. Kameoka, and S. Sagayama, "A real-time equalizer of harmonic and percussive components in music signals," in *Proceedings of the 9th International Conference for Music Information Retrieval Conference (ISMIR)*, 2008, pp. 139–144.

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

[5] K. Yoshii, M. Goto, K. Komatani, T. Ogata, and G. H. Okuno, "Drumix: An audio player with real-time drum-part rearrangement functions for active music listening," *IPSJ Journal*, vol. 48, no. 3, pp. 1229–1239, 2007.

[6] B. McFee, C. Raffel, D. Liang, P. W. D. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th Python in Science Conference (SciPy)*, 2015, pp. 18–25.

[7] J. Driedger, M. Müller, and S. Disch, "Extending harmonic-percussive separation of audio signals," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 611–616.

[8] D. Fitzgerald, "Harmonic/percussive separation using median filtering," in *Proceedings of the 13th International Conference on Digital Audio Effects (DAFx-10)*, 2010.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, J. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[10] C. Southall, R. Stables, and J. Hockman, "Automatic drum transcription using bi-directional recurrent neural networks," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016, pp. 591–597.

[11] R. Vogl, M. Dorfer, and P. Knees, "Recurrent neural networks for drum transcription," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016, pp. 730–736.

[12] C. Southall, R. Stables, and J. Hockman, "Automatic drum transcription for polyphonic recordings using soft attention mechanisms and convolutional neural networks," in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017, pp. 606–612.

[13] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Music genre database and musical instrument sound database," in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, 2003, pp. 229–230.

[14] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Popular, classical, and jazz music databases," in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, 2002, pp. 287–288.

[15] M. Goto, "AIST annotation for the RWC Music Database," in *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, 2006, pp. 359–360.