Drum Synthesis and Rhythmic Transformation with Adversarial Autoencoders

Maciej Tomczak
Digital Media Technology Lab
Birmingham City University
maciej.tomczak@bcu.ac.uk

Masataka Goto
National Institute of Advanced Industrial Science and Technology (AIST)
m.goto@aist.go.jp

Jason Hockman
Digital Media Technology Lab
Birmingham City University
jason.hockman@bcu.ac.uk

ABSTRACT

Creative rhythmic transformations of musical audio refer to automated methods for manipulation of temporally-relevant sounds in time. This paper presents a method for joint synthesis and rhythm transformation of drum sounds through the use of adversarial autoencoders (AAE). Users may navigate both the timbre and rhythm of drum patterns in audio recordings through expressive control over a low-dimensional latent space. The model is based on an AAE with Gaussian mixture latent distributions that introduce rhythmic pattern conditioning to represent a wide variety of drum performances. The AAE is trained on a dataset of bar-length segments of percussion recordings, along with their clustered rhythmic pattern labels. The decoder is conditioned during adversarial training for mixing of data-driven rhythmic and timbral properties. The system is trained with over 500000 bars from 5418 tracks in popular datasets covering various musical genres. In an evaluation using real percussion recordings, the reconstruction accuracy and latent space interpolation between drum performances are investigated for audio generation conditioned by target rhythmic patterns.

KEYWORDS

Neural Drum Synthesis; Rhythmic Transformation; Adversarial Autoencoders

ACM Reference Format:
Maciej Tomczak, Masataka Goto, and Jason Hockman. 2020. Drum Synthesis and Rhythmic Transformation with Adversarial Autoencoders. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20), October 12–16, 2020, Seattle, WA, USA.* ACM, Seattle, WA, USA, 9 pages. https://doi.org/10.1145/3394171.3413519

1 INTRODUCTION

Creative rhythmic transformations of musical audio are computational approaches for manipulation of musical sounds of varying length (e.g., long, short) and accentuation (e.g., loud, soft) that are grouped into patterns. Taking inspiration from capabilities offered in digital audio workstations (e.g., [30][1]) and plugins (e.g., [4][2]), along with emerging research in the music and multimedia communities, these transformations have become entrenched within modern music production workflows. Recent advances in powerful machine learning algorithms have given rise to new modalities of synthesis and effects processing procedures, which in turn have afforded new musical supportive systems for pitch, timbre and rhythm manipulation for music arrangement and sound design. Although various neural audio synthesis systems have been proposed, the majority of these have focused on generation, interaction and visualisation of pitched instruments, and relatively few have explored generation of percussion instruments and transformations of the underlying rhythmic patterns.

In this study, an end-to-end model for neural audio synthesis is created with the intention of performing rhythmic transformation of drum sounds. System performance is measured in comparison to three pre-existing algorithms on a new database of 5418 percussion performances extracted from real music recordings. The aim of this transformation is to assist musicians and music producers during the composition and production processes and to develop an understanding of meaningful low-level features for expression in generation of target sound qualities.

1.1 Background

Early approaches for rhythmic transformation of audio signals [23, 28, 45, 55] relied on signal processing techniques for sound segmentation, pattern matching (i.e., segment alignment between different metrical levels such as beats or note onsets), and time-stretching (e.g., using a phase vocoder) to satisfy the target transformation. Such methods rely heavily on the initial analysis of the recording and are thus prone to artifacts (e.g., transient smearing) caused by incorrect demarcation of temporally-relevant event positions. In the recent years, deep generative models such as variational autoencoders (VAE) [33] and generative adversarial networks (GAN) [22] have seen increasing success in various fields through targeting the task of learning and manipulation of disentangled feature representations. Disentangled representations denote techniques that break down each input data feature into narrowly defined variables to be encoded into separate dimensions. Multiple machine learning models have been proposed in various domains, such as computer vision (e.g., semantic analysis of images [10]), natural language processing (e.g., sentiment modification in text [20]), or speech synthesis (e.g., speaker identity modelling [29]).

In music, deep generative models have been applied to symbolic music representations in [47, 51, 53, 54]; however, modelling of symbolic music operates within a lower-dimensional space than raw audio signals and constrains the output generations to sequences

[1] https://support.apple.com/en-gb/HT207864
[2] https://www.xlnaudio.com/products/xo

MM '20, October 12–16, 2020, Seattle, WA, USA

ACM ISBN 978-1-4503-7988-5/20/10…$15.00
https://doi.org/10.1145/3394171.3413519

of instructions for a fixed set of sounds. Concurrently, substantial work has been performed in neural audio synthesis, where the task constitutes modelling of higher-dimensional information captured in the music content. Approaches to neural audio synthesis can be divided into two categories: (1) time-domain (i.e., raw audio) based, in which audio samples are optimised directly; and (2) time-frequency domain (i.e., spectrogram) based, in which log magnitudes of a short-time Fourier transform (STFT) are used as input to a network, with the requirement of phase reconstruction process during the inference time of the model.

The authors in [13] adapted the WaveNet architecture for raw audio generation of piano performances at longer timescales across tens of seconds. More recently, [12] presented a model based on vector-quantised variational autoencoder [46] for generating musical performances in the raw audio domain conditioned on styles learned from different artists and genres. Engel et al. [19] conditioned an autoregressive WaveNet [52] autoencoder on raw audio to synthesise meaningful instrument note embeddings as portrayed in the NSynth dataset.

Approaches operating in the time-frequency domain include the work by [36] who trained a Gaussian mixture VAE to learn disentangled representations of pitch and timbre for the synthesis of concert instruments. Similarly, the authors in [5] synthesised orchestral instruments through adversarial latent training with Wasserstein autoencoders (WAE) [48]. A similar approach was used by [1] for synthesis of short (i.e., one-shot) percussion samples.

Alternatively, GANs have been used to jointly model log magnitude spectrograms and phases for a fast neural audio synthesis [18] as well as for raw audio synthesis of one-shot drum sounds in [17].

1.2 Motivation

The motivation of this work is derived from the popular task of *redrumming* [35, 49] that is present in a professional music production setting. Here musicians, desiring a certain sound or aesthetic influenced by the style of artists they admire, replace the rhythmic pattern of drums in their recordings (i.e., *source*) with that from an idealised recording (i.e., *target*). Previous methods used for achieving this effect relied on signal processing procedures that would ultimately cascade errors through various stages of the transformation. Modern advances in neural audio synthesis allow for generation of new audio sequences trained on large quantities of data; however, not many focus on the rhythmic aspects of such transformations.

The goal of this work is to extend the possibilities of the current redrumming procedure to facilitate the creation of new drum arrangements from arbitrary audio inputs. The proposed model achieves redrumming by synthesising the individual drum instruments to imitate the source recordings with the rhythmic pattern of the target.

To achieve a redrumming effect, the proposed model seeks not only to synthesise individual drum instruments, but also to extend neural audio synthesis to include the manipulation of rhythmic patterns within bar-length segments of arbitrary percussion recordings. A major contribution of this paper is the development of a system that does not require tedious discretised note segmentation or rhythmic event selection prior to transformation. A user is given the freedom to manipulate the structure within a bar without reliance on discrete identification of rhythmic boundaries towards a continuous transformation. This is achieved with the proposed framework based on Gaussian mixture adversarial autoencoders (`AAE-GM`) conditioned on rhythmic patterns present in real music recordings.

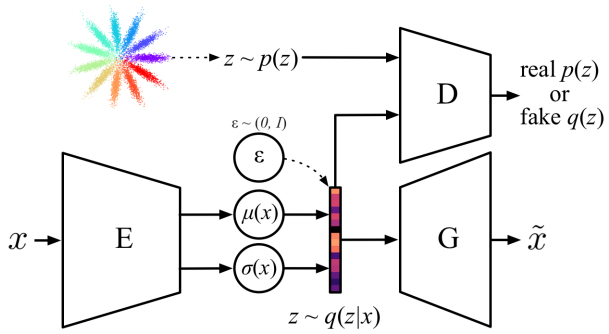

Figure 1: *Proposed architecture for joint drum synthesis and rhythm transformation. Input data x is mapped onto a latent variable $z \sim q(z|x)$. Encoder E tries to trick discriminator D with artificially generated latent samples and generator G outputs spectrograms $\tilde{x}$. A Gaussian prior distribution $z \sim p(z)$ (star) allows the model to juxtapose similar rhythmic patterns in the latent space. Solid lines represent deterministic operations of the network and dashed lines represent stochastic operations.*

The remainder of this paper is structured as follows: Section 2 presents the proposed method for combined drum synthesis and rhythmic transformation of audio using adversarial autoencoders. Experimentation methodology and the dataset used for the study are detailed in Section 3 and results and discussion are provided in Section 4. Conclusions and future work are presented in Section 5.

2 METHOD

An overview of the proposed method for joint drum synthesis and rhythmic transformation is presented in Figure 1. The system is based on adversarial autoencoders introduced in [37] and is inspired by adversarial audio synthesis approaches in [5, 16, 18]. To achieve both drum synthesis and rhythmic transformation in a unified architecture, the proposed model originally extends adversarial audio synthesis to include a regularisation based on a Wasserstein GAN adversarial framework for the transformation of rhythmic and timbral qualities of drum recordings. It supports an AAE with gradient penalty and Guassian mixture prior for conditional disentanglement of rhythmic pattern styles.

2.1 Adversarial Autoencoders (AAE)

While similar in design to VAE [33], adversarial autoencoders (AAE) appropriate the additional discriminator network D from GANs, which aims to distinguish between real and fake (i.e., synthesised) samples. Real samples are sampled from an assumed prior distribution $p(z)$ imposed on the latent variables z, while fake samples are generated through the use of an encoder E conditional distribution $q(z|x)$. The decoder (i.e., generator network G) conditional

distribution is denoted by $p(x|z)$. In practice $p(x|z)$ and $q(z|x)$ are parameterised with neural networks and sampling from $q(z|x)$ is performed using a reparameterisation trick [33]. Let $p_d(x)$ be the data distribution of data sample x, and $p_g(x)$ be the distribution of data generated by the model. The encoder defines an aggregated posterior distribution $q(z)$ on the z as follows:

$$q(z) = \int q(z|x)p_d(x). \tag{1}$$

Following the more general formulation for GANs [40], the adversarial component of an AAE can be trained as:

$$\begin{aligned}\min_E \max_D V(E,D) &= \mathbb{E}_{z\sim p(z)}[f(D(z))] \\ &+ \mathbb{E}_{x\sim p_d(x)}[f(-D(E(x)))],\end{aligned} \tag{2}$$

where $\mathbb{E}[\cdot]$ denotes expectation and objective V is optimised by alternating parameter updates of encoder E and discriminator D in a min-max game characteristic of GAN models. When the concave function $f : \mathbb{R} \rightarrow \mathbb{R}$ is set to $f(x) = -\log(1 + \exp(-x))$, the formulation resembles that of the GAN by [22]. The Wasserstein GAN (WGAN) criterion [2] can be obtained by setting $f(x) = x$.

The parameters of the autoencoder are optimised by the reconstruction error, while the adversarial network guides the encoder to match the imposed prior. Thus, the encoder plays the role of the generator during the adversarial part of training, while the discriminator represents the adversarial network of GANs. After training, decoder G acts as a generative model that maps the imposed prior to the data distribution. Training of an AAE is performed in two phases: (1) the reconstruction phase and (2) the regularisation phase. In the reconstruction phase the reconstruction error of E and G is minimised together and in the regularisation phase, the parameters of the discriminator D are first updated by minimising $\mathcal{L}_D = -V(E,D)$ (i.e., to distinguish true samples generated by the prior from the generated codes processed by the autoencoder). The adversarial network then updates the encoder to confuse the discriminator. When combined, the two terms represent $\mathcal{L}_{total}$ as follows:

$$\mathcal{L}_{total} = BCE(p_d, p_g) + \beta\mathcal{L}_{WGAN-GP}, \tag{3}$$

where BCE denotes binary cross-entropy reconstruction cost between the original data samples $|S|$ (i.e., magnitude spectrograms in this study) and their reconstructions $|\hat{S}|$ as:

$$BCE(S,\hat{S}) = -[S\log\hat{S} + (1-S)\log(1-\hat{S})]; |S| < 1. \tag{4}$$

The second term in Equation (3) is the WGAN with gradient penalty (WGAN-GP) loss with weighting β from [25], proposed as an improved solution to gradient clipping in adversarial training of the discriminator, computed as:

$$\mathcal{L}_{WGAN-GP} = \mathcal{L}_D + \lambda\mathbb{E}_{\hat{x}\sim p_{\hat{x}}}[(||\nabla_{\hat{x}}D(E(x))||-1)^2], \tag{5}$$

where $\hat{x}$ represents a randomly weighted average between real and generated samples. Following [25], we set $\lambda = 10$. During the regularisation phase, this term imposes regularisation on latent variables z and can be trained end-to-end with gradient descent.

2.2 Implementation

Details for the proposed adversarial autoencoder with Gaussian mixture prior (`AAE-GM`) are presented in this section. All neural network models are implemented using the TensorFlow Python library.[3]

2.2.1 ***Input Features.*** Following the approach in [5], input audio (16-bit 22.05 kHz mono WAV files) is transformed with short-time Fourier transform (STFT) using a Hanning window with a window length of 2048 samples and a hop size of 324 samples to facilitate the desired temporal resolution of the network input. Mel-spectrograms are created from audio inputs of length of 41344 samples corresponding to a bar segment of 1.87s duration at 128 beats per minute (BPM), resulting in the network input S of size 512 bins by 128 STFT frames. Every bar is normalized to this duration by a time-stretching algorithm. Magnitudes of S are floored to 1e−3 and log-scaled in [0,1] according to the *BCE* range. Rhythmic pattern styles $\xi = 11$ (i.e., classes as clustered attributes of S) are used in supervised training and are defined in Sections 2.3 and 2.3.4.

2.2.2 ***Architecture.*** The selected models are based on architectures used by [5, 16, 18], and are modified accordingly to facilitate modelling of rhythmic patterns studied in this work. All convolution layers are 2-d with square kernels and zero-padding of half the kernel size (i.e., *same* padding) and 2-d feature normalization. All fully-connected layers are followed by 1-d feature normalization. All normalizations use the batch normalization algorithm by [31]. The non-linear activations are leaky rectified linear units (LeakyReLU with slope of 0.2). The deterministic encoder has 5 convolution layers with [16, 32, 64, 128, 256] output channels, a kernel size of 7 and stride 2. This downsampled representation with 256 feature maps of the input spectrograms is reshaped to 16384 values and followed by a bottleneck of 3 fully-connected dense layers with output sizes [2048, 1024, 512]. Two fully-connected layers μ and σ are used for sampling z with the reparametrisation trick [33], thus mapping the input to the latent space $z \in \mathbb{R}^{N_z}$, where $N_z = 64$. The decoder mirrors the structure of the encoder with 3 linear layers of output sizes [512, 1024, 2048] and a layer reshaping the vector into 256 feature maps. The following convolution layers use nearest-neighbour upsampling, a fast solution that was demonstrated to mitigate the creation of known checkerboard artifacts of transposed convolutions [42]. These maps are processed through 5 layers with an upsampling factor set to 4 and convolution layers with [128, 64, 32, 16, 1] output channels, kernel sizes [7, 7, 7, 9, 9] and stride 2. The last layer reconstructs the input shape of S (128×512) followed by a sigmoid activation function bounding the output to the *BCE* range. The adversarial discriminator consists of 3 fully-connected layers with output channels [2048, 2048, 1], where the last linear layer outputs the final score used to compute a scalar measure of how well the latent variable resembles the imposed prior distribution.

2.2.3 ***Representation of Prior Distribution.*** The authors in [37] observed that VAEs are largely limited by the Gaussian prior, and thus relaxed this constraint by allowing $p(z)$ to be any distribution by replacing the KL divergence with an adversarial loss

[3] https://www.tensorflow.org/

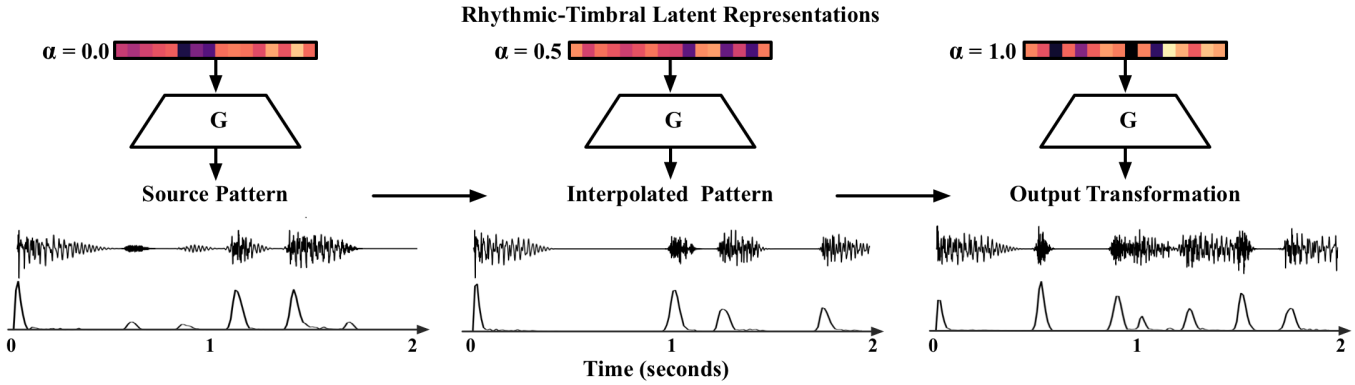

Figure 2: ***Rhythmic transformation of source (left) with intermediate pattern (middle) and resulting output transformation (right). Rhythmic envelopes (bottom) show changes to the rhythmic pattern as the latent code is manipulated via parameter α.***

imposed on the encoder output. Thus the latent variable z is required to have the same aggregated posterior distribution as the prior $p(z)$. The AAE framework makes it possible to leverage any prior knowledge that may be specific to the studied application. In this work, an isotropic Gaussian with 0 mean and I variance is used as a baseline, and compared against a prior distribution that is a mixture of ξ N_z-dimensional Gaussians, where ξ denotes the number of rhythmic pattern styles as defined in Section 2.3. This distribution can be depicted with a 2-d flower-like shape and allows modelling of a variety of similar rhythmic styles by pushing their latent codes to the center of the N_z-dimensional distribution. Following the notation by authors in [51], the means of ξ Gaussians are placed on a 2-d circle as:

$$\mu_i = \left[cos\left(\frac{2\pi i}{\xi}\right), sin\left(\frac{2\pi i}{\xi}\right), 0, ..., 0 \right], \tag{6}$$

where μ_i has a total of N_z dimensions. The covariance matrix Σ_i is calculated as:

$$v_i = \begin{bmatrix} cos(\frac{2\pi i}{\xi}) & sin(\frac{2\pi i}{\xi}), \\ -sin(\frac{2\pi i}{\xi}) & cos(\frac{2\pi i}{\xi}) \end{bmatrix}, \tag{7}$$

$$U_i = \begin{bmatrix} v_i^T & 0 \\ 0 & I \end{bmatrix}, \tag{8}$$

$$\Lambda = \begin{bmatrix} a_1 & 0 \\ 0 & \mathrm{diag}(a_2) \end{bmatrix}, \tag{9}$$

$$\Sigma_i = U_i \Lambda U_i^{-1}, \tag{10}$$

where U_i and Λ are $N_z \times N_z$ matrices. The variance $a_1 = 0.1$ for the radial (i.e., center-to-outer) dimension, and variance $a_2 = 0.001$ for the remaining dimensions. The matrix U_i is used to rotate Λ with respect to the position of the specific Gaussian. Thus, each of the ξ rhythmic pattern styles is associated with a separate Gaussian where patterns that are more similar are still able to be organised closer to each other.

2.2.4 ***Training.*** The model is trained using Adam optimiser [32] with an initial learning rate of 1e−4. All model weights use Xavier uniform initialisation [21]. The model is trained for around 100000 iterations for approximately 2 days using 4 Tesla V100 GPUs with a total batch size of 128. When training `AAE-GM`, the β parameter is gradually increased by 0.1 every 5000 iterations.

2.2.5 ***Signal Reconstruction.*** Mel-spectrograms generated by the trained model can be approximated back to the linear frequency scale and iteratively inverted with the Griffin-Lim algorithm [24] for 100 to 300 iterations.

2.3 Rhythmic Transformation

An overview of the rhythmic transformation is shown in Figure 2. A source recording is reduced to rhythmic-timbral representation output from a deterministic encoder and is passed to the generator together with a target pattern label. This latent code can be used to manipulate metrically relevant positions of drum instruments within a bar with mixing parameter α.

2.3.1 Representation of Rhythmic Patterns. Information related to rhythmic patterns is introduced during model training in order to guide the output generations towards particular target patterns. Audio tracks are first separated into a drums component and music parts (e.g., vocals, bass, other) with the Spleeter source separation library [26].[4] Next, audio tracks are segmented into bars b using the state-of-the-art beat and downbeat tracking algorithm [9] included in the madmom Python library [7].[5] Rhythmic patterns are represented with rhythmic envelope features processed with *LogFiltSpecFlux* from madmom, which performed well in onset detection function comparisons conducted in [8], for N ($N = 3$) frequency bands representing *low* (lowpass: 120 Hz), *mid* (bandpass: 120–2500 Hz) and *high* (highpass: 2500 Hz) contents of drum performances in each bar b. Following the authors in [14, 28], b features are resampled to a length of 144 time steps t and normalised to ranges between 0 and 1. The resulting M number of patterns is represented by a template matrix $\tau \in \mathbb{R}^{M \times N \times t}$. Figure 3 shows an example bar-length drum recording with the proposed representation of three rhythmic envelopes plotted together.

[4] https://github.com/deezer/spleeter
[5] https://github.com/CPJKU/madmom

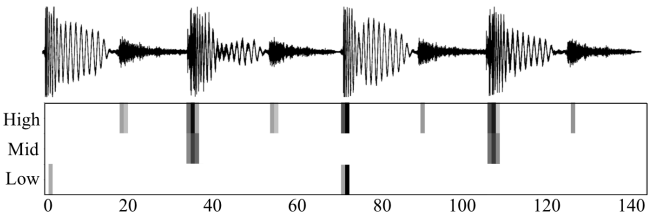

Figure 3: *Bar-length drum pattern definition using three frequency bands (low, mid and high).*

2.3.2 Clustering of Pattern Styles. Building on past research for rhythmic pattern modelling [15, 34, 43], an unsupervised clustering strategy via X-means [44] algorithm is proposed in this work. X-means is an unsupervised extension of the popular K-means algorithm, which does not require the predetermined K number of clusters prior to classification. The framework requires specification of the range within which K reasonably lies, and then jointly outputs the number of centroids together with a value for K that scores best by a model selection criterion such as Bayesian information criterion (BIC).

Centroid initialisation is known to influence clustering results in both K- and X-means algorithms, and as such results can be improved through informed initialisation. All experiments in this study incorporate K-means++ initialisation [3] with prior knowledge of rhythmic patterns extracted from transcriptions of the 50 most frequent kick, snare and hi-hats patterns from over 4.8 million bar-length drum patterns [38].[6] Patterns are resampled to satisfy the structure of rhythmic template matrix $\tau \in \mathbb{R}^{50\times3\times144}$.

2.3.3 Pattern Conditioning and Interpolation. In order to introduce conditioning based on rhythmic pattern styles, each input feature S used during training is assigned a categorical variable taking one of the ξ number of style states found through X-means clustering. During the reconstruction phase, one-hot encoded conditioning vectors for ξ rhythmic styles are concatenated with inputs to the generator G. The basis for a suitable ξ number of rhythmic pattern styles is presented in Section 2.3.4.

Interpolations in the latent space allows for the mixing of two different drum patterns. As the transformation is continuous, a gradual change is achievable from the source rhythmic pattern to the target pattern. The intermediate latent codes are produced using a linear interpolation between source and target latent codes such that:

$$\tilde{z} = \alpha z_{target} + (1-\alpha) z_{source} \tag{11}$$

where α is an interval between [0,1]. The interpolated codes $\tilde{z}$ are fed into the generator, which outputs the mixed bar-length drum performances.

2.3.4 Pattern Style Definition via X-means. Determination of a suitable number of rhythmic patterns ξ is achieved through the X-means algorithm using BIC scores calculated across $K = [5, 50]$ with a maximum number of clusters set to 100. As in [14], a pattern resolution of $t = 144$ is used. Rhythmic envelopes are smoothed for different standard deviations $\varsigma = [0.2, 0.6, 0.8]$ covering a range of 4 timesteps at a time. Convergence was most frequently observed at $K = 11$ with $\varsigma = 0.2$.

3 EXPERIMENTS

The model proposed in Section 2 is assessed through an experiment to determine (1) the rhythmic pattern organisation in the *latent space structure*, (2) an evaluation of the audio *reconstruction performance* compared with similar AE models, and (3) an evaluation of the transformation quality between source and target patterns through *latent space interpolation*. In this section the dataset, experimental methodology and baseline systems under evaluation are presented.

3.1 Data

This project makes use of three publicly available datasets: (1) `DALI` (4116 tracks) [39], (2) Harmonix (`HMX` 807 tracks) [41], and (3) `HJDB` (227 tracks) [27], as well as a private collection of 268 jazz, funk and R&B (`JFRB`) recordings. The resulting dataset contains 5418 musical pieces of polyphonic sound mixtures having various kinds of instruments and represents a wide variety of genres and rhythmic patterns. All audio recordings are in 16-bit mono WAV format and resampled to 22.05 kHz. To facilitate modelling of rhythmic patterns, those tracks are segmented into bars using the state-of-the-art downbeat tracking algorithm by [9].

In order to model rhythmic patterns from percussion instruments present in the dataset, source separation is performed with the pre-trained *4stems* model provided in the Spleeter library [26] to extract drum sounds from music sound mixtures. The resultant drum parts are used in two ways: (1) as training inputs described in Section 2.2.1, and (2) for rhythmic pattern modelling described in Section 2.3. In both scenarios, tracks with time signatures other than 4/4 or with an amplitude < 0.2—due either to empty bars or poor source separation—are excluded. After filtering, the data is represented by 5418 tracks with a total of 510859 bars. Assessment of the dataset tempi results in a median tempo of 128 BPM. To facilitate appropriate representation of a wide range of rhythmic patterns, all bar-length segments are time-stretched to a fixed tempo of 128 BPM with the Rubberband library.[7] The dataset samples are distributed among training (80%), validation (10%) and test sets (10%) with an equal distribution of bars per ξ rhythmic pattern styles throughout all sets during training.

3.2 Experimental Methodology

In order to view the organisation of the learned latent space, its structure is visualised with 2-d and 3-d plots for each of the rhythmic classes ξ with principal component analysis (PCA) portraying differences between two different prior distributions.

The ability of the proposed model to generate spectrograms is evaluated using both timbral and temporal reconstruction metrics: root-mean squared error (RMSE), log-spectral distance (LSD) and cosine similarity (CS). The LSD is calculated as follows:

$$LSD = \sqrt{\sum [10 \log_{10}(|S|/|\hat{S}|)]^2}. \tag{12}$$

[6] http://isophonics.net/ndrum

[7] https://breakfastquay.com/rubberband/

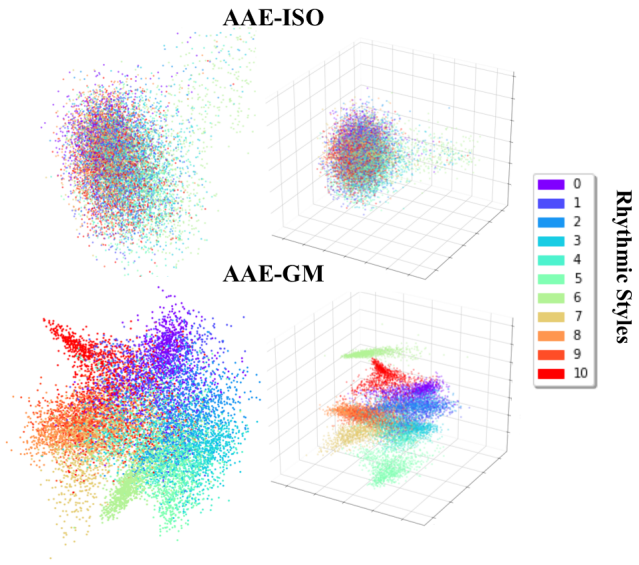

Figure 4: ***PCA visualisations of the baseline*** `AAE-ISO` ***(top) and the proposed*** `AAE-GM` ***(bottom) with 2 PCs (left) and 3 PCs (right) for 11 rhythmic styles.***

Following [11, 50], temporal reconstruction of the generations is evaluated with cosine similarity (CS) between rhythmic envelopes R (see Section 2.3) extracted from source χ and generated ν recording as follows:

$$CS_{\chi,\nu} = 1 - \frac{R_\chi \cdot R_\nu}{\|R_\chi\|\|R_\nu\|}. \tag{13}$$

CS will be close to unity for very similar patterns and nearer to zero for dissimilar patterns. All reported experiments in the following sections use 1000 patterns from each ξ rhythmic pattern style, resulting in a total of 11000 evaluation audio examples.

To evaluate the continuity of the transformations, latent space interpolations between rhythmic patterns are performed using Equation (11). Scores for each metric are calculated between the source recording and the resulting rhythmic transformation. Reconstruction scores for all examples are scaled to range [0,1] and averaged for different α.

3.3 Baseline Systems

In addition to the proposed Gaussian mixture AAE (`AAE-GM`) architecture, three additional models are implemented for comparisons: (1) AAE using isotropic Gaussian prior distribution (`AAE-ISO`), (2) variational autoencoder (`VAE`), (3) a Wasserstein autoencoder with maximum mean discrepancy (`WAE-MMD`) regularisation. All models share the same architecture implementations and are trained in a supervised manner. The proposed `AAE-GM` uses a regularisation based on a WGAN adversarial framework—including a gradient penalty with Guassian mixture prior for conditional disentanglement of rhythmic pattern styles—for the transformation of rhythmic and timbral qualities of drum recordings. As a comparison for the rhythmic transformation capabilities of the presented `AAE-GM` model, the audio synthesis framework using `WAE-MMD` [5] is here modified to act on longer timescales as present in bar-length patterns.

	VAE	WAE-MMD	AAE-ISO	AAE-GM
LSD	34.26	34.23	34.28	34.37
RMSE	0.39	0.38	0.38	0.38
CS	0.67	0.84	0.84	0.82

Table 1: ***Reconstruction scores shown for three baseline models and the proposed*** `AAE-GM`.

We follow the `WAE-MMD` implementation without the conditioning module proposed by the authors in [5]. In the case of `VAE`, the model minimises the evidence lower bound objective [33] with isotropic Gaussian latent distribution. The `WAE-MMD` uses *BCE* reconstruction loss where the regularisation from Equation (3) is replaced with maximum mean discrepancy (MMD). MMD represents a distance measure between the samples of the distributions $x \sim p(x)$ and $y \sim q(y)$ and was proposed as a more flexible regularisation to Kullback–Leibler divergence used in a vanilla `VAE` [5]. MMD defines a differentiable divergence and was developed as a distance between probabilistic moments $\phi_{p,q}$ that map to a general reproducing kernel Hilbert space as follows:

$$\begin{aligned} ||\phi_p - \phi_q||_H^2 &= \langle \phi_p - \phi_q, \phi_p - \phi_q \rangle \\ &= \mathbb{E}_{p,p}\kappa(x,x') + \mathbb{E}_{q,q}\kappa(y,y') \\ &\quad -2\mathbb{E}_{p,q}\kappa(x,y), \end{aligned} \tag{14}$$

where $\mathbb{E}_{p,q}$ is the expectation that is evaluated with a radial basis kernel function κ:

$$\kappa(x,y) = \exp\left(\frac{||x^2 + y^2||}{-2\Sigma^2}\right). \tag{15}$$

4 RESULTS AND DISCUSSION

4.1 Latent Space Structure

The 64-d latent spaces for `AAE-GM` and `VAE` are visualised in 2- and 3-d in Figure 4 using PCA. PCA ensures that the visualisation is a linear transform of the original space, and thus preserves the real distances inside the latent space. As can be seen, it is not possible to distinguish between the different rhythmic pattern styles in `AAE-ISO` without the Gaussian mixture prior. The effect of the proposed `AAE-GM` with Gaussian mixture prior can be clearly seen with more visibly organised clusters in both 2-d and 3-d PCA representations. When analysing mean rhythmic pattern representations as clustered by the X-means algorithm, pattern types 0 (purple) and 6 (green) represent disparate rhythmic styles—style 0 is typified by a clear 16th-note pattern and style 6 is an 8th-note pattern with an accent on the second beat of the musical measure.

4.2 Reconstruction Performance

The reconstruction performance scores of the proposed and baseline models are shown in Table 1. The mean LSD and RMSE scores describe the spectral reconstruction quality of generated audio spectrograms with regard to the original. The results for mean LSD and RMSE indicate that the proposed `AAE-GM` model achieves a similar level of reconstruction quality as the other approaches. The CS score quantifies how similar are the rhythmic envelopes of the newly synthesised audio in comparison to the original. Although the reconstructions from the `AAE-GM`, `WAE-MMD` and `AAE-ISO` all

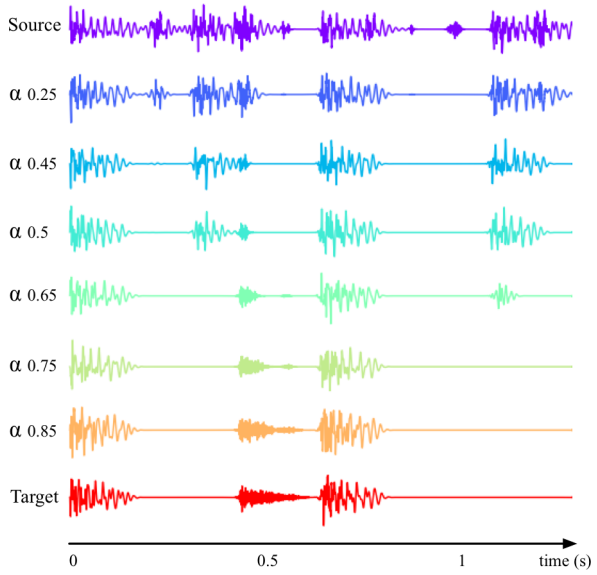

Figure 5: ***Example of interpolations between two rhythmic patterns.***

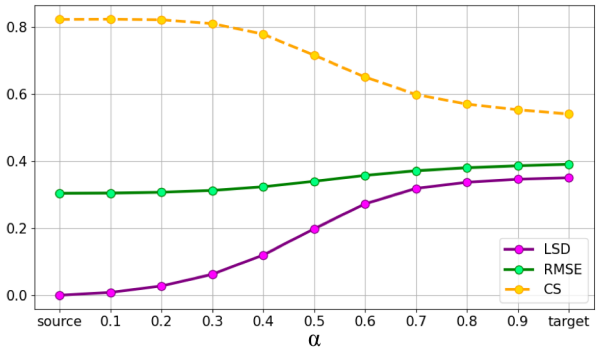

Figure 6: ***Reconstruction scores for interpolations between source and target rhythmic patterns. The results are calculated as mean of 11000 transformations per each interpolated value for α.***

contain a degree of noise, the results of these three systems achieve comparable CS (>0.8). The CS for the `VAE` is considerably lower, likely due to the reconstructions being generated with a more substantial amount of noise.

The reconstruction CS score shows high similarities of the generations in the temporal domain, whereas LSD and RMSE scores outline the challenging aspects of synthesising realistic audio with neural networks. It is anticipated that reconstruction quality would also be improved with audio that has not been manipulated through time-stretching, improved filtering of noisy patterns, and larger architectures with dilated convolutions with skip connections to help consolidate more information into the network. To mitigate the artifacts of the time-stretching effect, it would be useful to investigate the potential of variable-length features for training of neural networks. The transformation would also benefit from additional drum placement information provided by automatic drum transcription.

4.3 Latent Space Interpolation

One of the chief characteristics of a well-trained latent representation is its ability to generate meaningful samples based on embeddings created by performing linear interpolation (Section 2.3.3). Smooth transitions in the latent space are desired in a user-controlled sound transformation [5, 6]. Figure 5 demonstrates a transformation between two different types of rhythmic patterns and different instruments (e.g., purple kick drum transforms into a red snare drum at around 0.5s). Notably, the temporal positions of the last two events in the source audio (i.e., purple kicks after 1.0s) are gradually shifted in time as they are morphed into a single softer and higher pitched sound event at $\alpha = 0.5$ before it disappears completely at $\alpha = 0.75$.

To analyse the effect of the rhythmic transformation for the intermediate α values, all audio examples in the test set are interpolated to randomly chosen target patterns. Figure 6 depicts the reconstruction scores for all 11000 transformations. As expected, the CS decreases as the transformation moves output audio further away from the source. As the transformation operates on audio containing percussion only, the intention is not to adjust the spectral content by a large margin. The scores for RMSE reflect that characteristic by not varying considerably throughout the interpolation, indicating that the spectrogram reconstruction quality remains similar. On the other hand, LSD mirrors the behaviour of CS indicating change in the spectral contents moving towards a novel target spectrogram after transformation. This can be equivalent to moving and removing an event in one position or transforming it into another instrument.

Audio examples and additional experiments are available on a supplementary website.[8]

5 CONCLUSIONS AND FUTURE WORK

We propose a novel method for combined drum synthesis and rhythmic transformation akin to the popular task of redrumming. We provide user control to continuously navigate among complex rhythmic possibilities by interpolating through a low-dimensional latent space. This is achieved by integrating Gaussian mixture latent distributions for rhythmic pattern conditioning with state-of-the-art adversarial autoencoders. To train and evaluate the system, we collected and annotated a dataset of over 500000 bars from 5418 audio tracks from a variety of musical genres. Our experiments confirmed the importance of the structure of the disentangled latent distributions that relate to rhythm and timbre. In future work, we will investigate evaluation metrics for latent space organisation and rhythmic transformation, as well as the effects of additional musical conditioning techniques for different prior distributions.

[8] https://maciek-tomczak.github.io/maciek.github.io/Drum-Synthesis-and-Rhythmic-Transformation/

ACKNOWLEDGMENTS

This work was supported in part by JST ACCEL Grant Number JPMJAC1602.

REFERENCES

[1] Cyran Aouameur, Philippe Esling, and Gaëtan Hadjeres. 2019. Neural Drum Machine: An Interactive System for Real-time Synthesis of Drum Sounds. In *Proceedings of the International Conference on Computational Creativity (ICCC)*.
[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein GAN. In *Proceedings of the International Conference on Machine Learning (ICML)*. 214–223.
[3] David Arthur and Sergei Vassilvitskii. 2006. *k-means++: The Advantages of Careful Seeding*. Technical Report. Stanford.
[4] XLN Audio. 2019. XO. Available at: https://www.xlnaudio.com/products/xo.
[5] Adrien Bitton, Philippe Esling, Antoine Caillon, and Martin Fouilleul. 2019. Assisted Sound Sample Generation with Musical Conditioning in Adversarial Auto-encoders. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*.
[6] Adrien Bitton, Philippe Esling, and Axel Chemla-Romeu-Santos. 2018. Modulated Variational Auto-encoders for Many-to-many Musical Timbre Transfer. *arXiv:1810.00222* (2018).
[7] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. 2016. madmom: A New Python Audio and Music Signal Processing Library. In *Proceedings of the ACM International Conference on Multimedia*. 1174–1178.
[8] Sebastian Böck, Florian Krebs, and Markus Schedl. 2012. Evaluating the Online Capabilities of Onset Detection Methods. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 49–54.
[9] Sebastian Böck, Florian Krebs, and Gerhard Widmer. 2016. Joint Beat and Downbeat Tracking with Recurrent Neural Networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 255–261.
[10] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*. 2172–2180.
[11] Matthew E. P. Davies, Philippe Hamel, Kazuyoshi Yoshii, and Masataka Goto. 2014. AutoMashUpper: Automatic Creation of Multi-song Music Mashups. *In IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)* 22, 12, 1726–1737.
[12] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. 2020. Jukebox: A Generative Model for Music. *arXiv:2005.00341* (2020).
[13] Sander Dieleman, Aäron van den Oord, and Karen Simonyan. 2018. The Challenge of Realistic Music Generation: Modelling Raw Audio at Scale. In *Proceedings of the Neural Information Processing Systems (NIPS)*. 8000–8010.
[14] Simon Dixon, Fabien Gouyon, and Gerhard Widmer. 2004. Towards Characterisation of Music via Rhythmic Patterns. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*.
[15] Simon Dixon, Elias Pampalk, and Gerhard Widmer. 2003. Classification of Dance Music by Periodicity Patterns. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. 159–165.
[16] Chris Donahue, Julian McAuley, and Miller Puckette. 2018. Adversarial Audio Synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
[17] Jake Drysdale, Maciek Tomczak, and Jason Hockman. 2020. Adversarial Synthesis of Drum Sounds. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*.
[18] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. 2019. GANSynth: Adversarial Neural Audio Synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
[19] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan. 2017. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. In *Proceedings of the International Conference on Machine Learning (ICML)*. 1068–1077.
[20] Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style Transfer in Text: Exploration and Evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
[21] Xavier Glorot and Yoshua Bengio. 2010. Understanding the Difficulty of Training Deep Feedforward Neural Networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*. 249–256.
[22] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*. 2672–2680.
[23] Fabien Gouyon, Lars Fabig, and Jordi Bonada. 2003. Rhythmic Expressiveness Transformations of Audio Recordings: Swing Modifications. In *Digital Audio Effects (DAFx) Workshop*. 8–11.
[24] Daniel Griffin and Jae Lim. 1984. Signal Estimation from Modified Short-time Fourier Transform. *In IEEE Transactions on Acoustics, Speech, and Signal Processing (TASSP)* 32, 2 (1984), 236–243.
[25] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C. Courville. 2017. Improved Training of Wasserstein GANs. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*. 5767–5777.
[26] Romain Hennequin, Anis Khlif, Felix Voituret, and Manuel Moussallam. 2019. Spleeter: A Fast And State-of-the Art Music Source Separation Tool With Pre-trained Models. *Late-Breaking Demo Session Abstract in the International Society for Music Information Retrieval Conference* (2019).
[27] Jason Hockman, Matthew E. P. Davies, and Ichiro Fujinaga. 2012. One in the Jungle: Downbeat Detection in Hardcore, Jungle, and Drum and Bass. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 169–174.
[28] Jason A. Hockman, Juan P. Bello, Matthew E. P. Davies, and Mark D. Plumbley. 2008. Automated Rhythmic Transformation of Musical Audio. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. 177–180.
[29] Wei-Ning Hsu, Yu Zhang, Ron J. Weiss, Heiga Zen, Yonghui Wu, Yuxuan Wang, Yuan Cao, Ye Jia, Zhifeng Chen, Jonathan Shen, Patrick Nguyen, and Ruoming Pang. 2018. Hierarchical Generative Modeling for Controllable Speech Synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
[30] Apple Inc. 2017. Drummer Loops in Logic Pro X. Available at: https://support.apple.com/en-gb/HT207864.
[31] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the International Conference on Machine Learning (ICML)*. 448–456.
[32] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
[33] Diederik P. Kingma and Max Welling. 2013. Auto-encoding Variational Bayes. *arXiv:1312.6114* (2013).
[34] Florian Krebs, Sebastian Böck, and Gerhard Widmer. 2013. Rhythmic Pattern Modeling for Beat and Downbeat Tracking in Musical Audio. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 227–232.
[35] Patricio López-Serrano, Matthew E. P. Davies, and Jason Hockman. 2018. Break-informed Audio Decomposition for Interactive Redrumming. *Late-breaking Demo Session Abstract in the International Society for Music Information Retrieval Conference (ISMIR)* (2018).
[36] Yin-Jyun Luo, Kat Agres, and Dorien Herremans. 2019. Learning Disentangled Representations of Timbre and Pitch for Musical Instrument Sounds Using Gaussian Mixture Variational Autoencoders. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.
[37] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial Autoencoders. *arXiv:1511.05644* (2015).
[38] Matthias Mauch and Simon Dixon. 2012. A Corpus-based Study of Rhythm Patterns. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 163–168.
[39] Gabriel Meseguer-Brocal, Alice Cohen-Hadria, and Geoffroy Peeters. 2018. DALI: A Large Dataset Of Synchronized Audio, Lyrics And Notes, Automatically Created Using Teacher-student Machine Learning Paradigm. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.
[40] Vaishnavh Nagarajan and J. Zico Kolter. 2017. Gradient Descent GAN Optimization is Locally Stable. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*. 5585–5595.
[41] Oriol Nieto, Matthew McCallum, Matthew E. P. Davies, Andrew Robertson, Adam Stark, and Eran Egozy. 2019. The Harmonix Set: Beats, Downbeats, and Functional Segment Annotations of Western Popular Music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.
[42] Augustus Odena, Vincent Dumoulin, and Chris Olah. 2016. Deconvolution and Checkerboard Artifacts. *Available at: https://distill.pub/2016/deconv-checkerboard/, Distill* (2016).
[43] Geoffroy Peeters. 2005. Rhythm Classification Using Spectral Rhythm Patterns. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. 644–647.
[44] Dan Pelleg and Andrew W. Moore. 2000. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In *Proceedings of the International Conference on Machine Learning (ICML)*. Morgan Kaufmann, 727–734.
[45] Emmanuel Ravelli, Juan P. Bello, and Mark Sandler. 2007. Automatic Rhythm Modification of Drum Loops. *Signal Processing Letters, IEEE* 14, 4 (2007), 228–231.
[46] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. 2019. Generating Diverse High-fidelity Images with VQ-VAE-2. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*. 14837–14847.
[47] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. 2018. A Hierarchical Latent Vector Model for Learning Long-term Structure in Music. *arXiv:1803.05428* (2018).

[48] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. 2017. Wasserstein Auto-encoders. *Proceedings of the International Conference on Learning Representations (ICLR)*.

[49] Maciek Tomczak, Jake Drysdale, and Jason Hockman. 2019. Drum Translation for Timbral and Rhythmic Transformation. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*.

[50] Maciek Tomczak, Carl Southall, and Jason Hockman. 2018. Audio Style Transfer with Rhythmic Constraints. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*. 45–50.

[51] Andrea Valenti, Antonio Carta, and Davide Bacciu. 2020. Learning a Latent Space of Style-Aware Symbolic Music Representations by Adversarial Autoencoders. *arXiv:2001.05494* (2020).

[52] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. WaveNet: A Generative Model for Raw Audio. *arXiv:1609.03499* (2016).

[53] Jian Wu, Changran Hu, Yulong Wang, Xiaolin Hu, and Jun Zhu. 2019. A Hierarchical Recurrent Neural Network for Symbolic Melody Generation. *In IEEE Transactions on Cybernetics* 50, 6 (2019), 2749–2757.

[54] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. 2017. MidiNet: A Convolutional Generative Adversarial Network for Symbolic-domain Music Generation. *arXiv:1703.10847* (2017).

[55] Kazuyoshi Yoshii, Masataka Goto, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. 2007. Drumix: An Audio Player with Real-time Drum-part Rearrangement Functions for Active Music Listening. *IPSJ (Information Processing Society of Japan) Journal* 48, 3 (2007), 1229–1239.