DRUM TRANSLATION FOR TIMBRAL AND RHYTHMIC TRANSFORMATION

Maciek Tomczak, Jake Drysdale and Jason Hockman

Digital Media Technology Lab (DMT Lab)
Birmingham City University
Birmingham, United Kingdom
{maciek.tomczak@bcu.ac.uk, jake.drysdale@bcu.ac.uk, jason.hockman@bcu.ac.uk}

ABSTRACT

Many recent approaches to creative transformations of musical audio have been motivated by the success of raw audio generation models such as WaveNet, in which audio samples are modeled by generative neural networks. This paper describes a generative audio synthesis model for multi-drum translation based on a WaveNet denosing autoencoder architecture. The timbre of an arbitrary source audio input is transformed to sound as if it were played by various percussive instruments while preserving its rhythmic structure. Two evaluations of the transformations are conducted based on the capacity of the model to preserve the rhythmic patterns of the input and the audio quality as it relates to timbre of the target drum domain. The first evaluation measures the rhythmic similarities between the source audio and the corresponding drum translations, and the second provides a numerical analysis of the quality of the synthesised audio. Additionally, a semi- and fully-automatic audio effect has been proposed, in which the user may assist the system by manually labelling source audio segments or use a state-of-the-art automatic drum transcription system prior to drum translation.

1. INTRODUCTION

The creative transformation addressed in this paper is generative audio synthesis of percussive instruments, which involves mapping (or *translation*) of musical audio to drum sounds achieved with artificial neural networks. Flexible digital audio effects that utilise machine learning techniques would benefit musicians and music producers by generating audio controllable by a target rhythm, melody or style that may be used directly in the music production process. Among such tools, autoregressive (AR) models for raw audio generation such as WaveNet [1] have inspired several systems that utilise the power of generative neural networks for musical audio synthesis. Audio synthesis in these models is achieved by learning an AR distribution that predicts the next audio sample from the previous samples in its receptive field using a series of dilated convolutions. The majority of these systems have been developed to address pitched instruments [1, 2], while no such systems have been focused on the generation of percussive instruments and rhythmic aspects of such transformations.

1.1. Background

In the field of digital audio effects, rhythmic and timbral transformations have initially been addressed through signal processing architectures [3, 4, 5, 6]. At present there are only a handful of systems that approach this type of transformation using artificial neural networks in areas such as audio style transfer [7] and AR generative audio synthesis [1, 8, 9].

WaveNet is a generative audio synthesis model that was developed for tasks related to speech synthesis [10, 11]. To date, a relatively small number of systems have experimented with it for synthesis of musical audio. Engel et al. [2] proposed a WaveNet autoencoder that learns codes that meaningfully represent the space of musical instruments with the ability to model long temporal dependencies. This work led to the NSynth system [2], a neural synthesiser capable of generating new sound embeddings learned from a large dataset of musical notes. Dieleman et al. [12] adapted the WaveNet architecture for the unconditional generation of piano music that exhibits stylistic consistency at longer timescales across tens of seconds. In [13], the authors combined audio and symbolic models and use a long short-term memory recurrent neural network (RNN) to learn melodic structures of different styles of music, which are used as conditioning input to a WaveNet-based instrument melody generator. Other AR models include RNN-based architectures such as: VRNN [14], SampleRNN [15] and WaveRNN [16]. Alternatively, the WaveNet architecture has been used in the context of musical timbre transfer. Huang et al. [17] adapted an image-based style transfer method [18] for translation of an image from one domain to another using a conditional WaveNet synthesiser within the TimbreTron model. Kim et al. [8] proposed a music synthesis system with timbre control that learns to generate spectrograms from symbolic music representations and instrument embeddings, and generates raw audio with a WaveNet vocoder.

Mor et al. [9, 19] introduced a novel system for timbre and style translations that combined the WaveNet autoencoder with unsupervised adversarial training. This architecture differed from other generative audio synthesis models in that it could convert the timbre of one instrument to that of another while preserving the melody and rhythm of the input. The WaveNet autoencoder architecture of Mor et al. [19] has been adopted, with key differences made in training strategies and the use of a simplified architecture, specialised for rhythmic and timbral transformations of percussion instruments.

1.2. Motivation

In this paper, a system that adapts the music translation approach for timbre transfer is proposed, with the aim of encoding the rhythmic structure of an arbitrary audio input as a combination of different percussion instruments from the common drum kit. In this

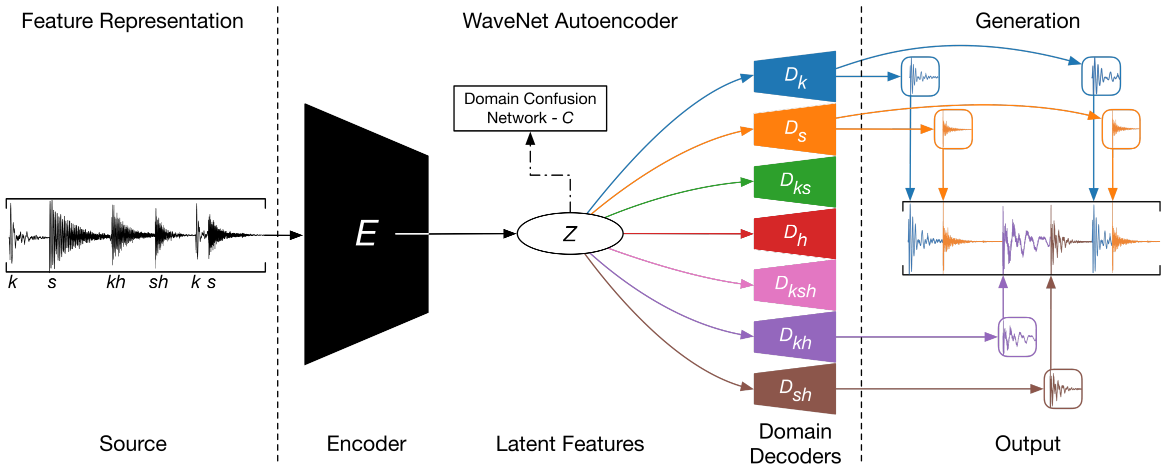

Figure 1: *Drum translation overview in three stages. Source audio is transformed to output through a single shared autoencoder of domain p specialised on domain decoders D_p, where p represents: kick (k), snare (s), kick and snare (ks), hi-hat (h), kick, snare and hi-hat (ksh), kick and hi-hat (kh) or snare and hi-hat (sh). Colours illustrate pathways between source and corresponding D_p trained to synthesise the target instrument (e.g., orange decoder D_s synthesises snare drums). Solid lines represent information flow during synthesis and dashed-dotted line represents information flow to a domain confusion network present only during training.*

transformation, the timbre of an input is transformed such that it sounds as if it were played on a different drum. To that end, a denoising WaveNet autoencoder architecture is modified and specialised for drum translation by utilising an unsupervised training strategy of a multi-domain latent space that is trained end-to-end on combinations of drum samples. In this architecture, a single encoder of [19] encodes a shared latent space for multiple decoders to use during training and audio generation. The size of the architecture is adjusted to learn short-term sounds of drum samples, while maintaining encodings for different drum instruments. The rhythmic accuracy of this model has been explored as well as a variety of creative percussive transformations in a simplified task of *drum-to-drum* translation akin to the task of redrumming or drum replacement [20]. The aims of the system are to facilitate the creation of new drum arrangements from arbitrary audio inputs provided by untrained musicians and to uncover the musical relationships between audio recordings that might otherwise have never been heard.

The remainder of this paper is structured as follows: Section 2 outlines our proposed method for drum translation. Section 3 presents experiments undertaken to assess the rhythmic accuracy and the quality of the translated audio. Section 4 provides experiment results with a discussion on audio degradation caused by the system for different drum domains. Conclusions and suggestions for future work are presented in Section 5.

2. METHOD

This approach to drum translation concerns the task of synthesising source audio to corresponding drum sounds. The system is inspired by architecture of [19], in which music signals can be translated across instruments and styles. This paper contributes to this kind of transformation by simplifying the translation network and proposing a new training strategy specialised towards percussion instruments.

Figure 1 provides an overview of the proposed drum translation system, which is comprised of three stages: (1) Feature representation; (2) WaveNet autoencoder; and (3) Generation. At the core of the system is a WaveNet autoencoder network with a shared encoder and a disentangled latent space, distributed across each drum domain decoder D_p, where p represents a percussion domain for P total number of domains. In total, there are seven percussion domains ($P = 7$) defined as kick (`k`), snare (`s`), hi-hat (`h`) instruments as well as their combinations such as, kick and hi-hat (`kh`). During training, multiple source-target p pathways (one per drum domain illustrated by different colours in Figure 1) are encoded by a domain-independent encoder E. The input to the neural network is an audio segment X_p of length T samples ($T = 6000$) representing a waveform of one of the seven drum domains. Each segment is distorted by random pitch modulation to prevent the network from memorising the input signal and provide a semantic encoding.

To improve the generalisability of a single encoder during training and to increase the size of the training data, a pitch augmentation approach of [19] is implemented. In popular music production, pitch shifting of individual drum samples is a common processing technique that is used either on all drums or on a subset of drum samples that are layered underneath other sounds to create richer timbres. Instead of augmenting only parts of the input data as in [19], pitch is modulated across the whole length of each audio segment X_p by a random value between ± 3 semi-tones with LibROSA [21]. The final representation of each augmented percussion segment used for training is $X_p = \{x_{p,1}, ..., x_{p,T}\}$.

The input audio goes through the Wavenet encoder E, a fully convolutional network, and outputs latent space Z that is downsampled with 8-bit μ-law encoding [1]. The latent space is then used to condition a domain confusion network [22] responsible for providing an adversarial signal to the encoder during training. The latent signal is then temporally upsampled to the original audio rate and is used to condition a WaveNet decoder D_p. Each decoder uses a softmax activation to output the probability of the next time step. Once training is finished the embeddings of all drum domains in the shared latent space can be used to transform source audio from any arbitrary audio domain.

2.1. WaveNet Autoencoder

A similar dilated convolutional WaveNet encoder architecture as [2] is adopted and a WaveNet decoder from [19] to model percussion sounds in the time domain. Dilated convolutions are convolutions with holes that greatly increase the receptive field while significantly reducing the model parameters and computational cost. In addition to dilated convolutions, WaveNet incorporates a residual learning framework [23] that reduces training time and avoids the vanishing gradient problem linked with the training of very deep neural networks.

The shared encoder E has 18 layers with two blocks of nine residual-layers [23] and a maximum dilation of 512 samples. As in [19], a residual-layer structure with a ReLU non-linearity is used, a non-causal dilated convolution with an increasing kernel size, a second ReLU, and a 1x1 convolution (i.e., a time-distributed fully connected layer) followed by a residual summation of the activations before the first ReLU. Unless specified otherwise, 64 channels are used in all hidden layers of the autoencoder architecture. After two blocks, the encoding goes through a 1x1 layer and an average pooling with a kernel size of υ samples ($\upsilon = 400$).

The WaveNet domain decoder D_p is conditioned with a temporally upsampled version of the latent encoding Z^{up} obtained with nearest neighbour interpolation. The conditioning signal adds parameters to the probability distribution so that it depends on variables that describe the audio to be generated instead of only using the previously generated samples. Without conditioners, WaveNet has been shown to mix sequences of speech by repeated phoneme shifting between voices of all speakers used in training of the model. As in [19], the conditioning goes through a different 1x1 layer for each decoder D_p to ensure that the latent space is domain independent. This reduces source-target pathway memorisation, which is also aided by pitch augmentation. To ensure that only previous samples are used in the generation of the new ones, decoders D_p use dilated causal convolutions together with additional non-linear operations to enable them to learn input audio representations that cannot be captured with just linear operations. Each D_p has two blocks of nine residual-layers, where each layer contains a causal dilated convolution with an increasing kernel size, a gated hyperbolic tangent activation [1] (the main source of non-linearity), a 1x1 convolution followed by the residual summation of the layer input, and a 1x1 convolution layer for skip connections. Encoding Z^{up} is used to condition each residual-layer during training. The skip connections are summed with a ReLU non-linearity activation and passed through a 1x1 convolution layer before a softmax activation layer.

2.2. Domain Confusion Network

In order to introduce an adversarial signal to the autoencoder and ensure that the encoding is not domain-specific, a domain confusion network C is implemented following [19]. The network predicts the percussion domain label of the input data based on the latent vectors Z. It uses a single gradient reversal layer defined in [22] and three 1D-convolution layers. The gradient reversal layer (GRL) reverses the gradient by multiplying it with a negative scalar λ ($\lambda = 0.01$). The GRL ensures that the feature distributions over the P drum domains are made similar (i.e., as difficult as possible to recognise for the domain classifier C), thus resulting in the domain-independent features. The three 1D-convolutional layers all include ELU non-linearities [24] with 128 channels in all hidden layers. After three layers the output is passed through a tanh and a 1x1 convolution layer to project the vectors to P total number of domains.

2.3. μ-law Quantisation

WaveNet predicts a non-normalised probability distribution from the residual-layers and transforms it into a proper probability distribution by using a softmax function. The authors of the original WaveNet [1] show that softmax distribution tends to be more flexible than other mixture models and can more easily model arbitrary distributions as it makes no assumptions about their shape.

All audio files processed by the model use a sampling rate of 22.05 kHz and are stored as 16-bit integers. To model all possible values per time step, a softmax layer would need to output 2^{16} probabilities. To moderate this high bit-depth resolution of the input audio, a μ-law algorithm [25] is implemented and quantises the data to 2^8 quantisation levels:

$$f(x_p) = sign(x_p) * \frac{ln(1 + \mu|x_p|)}{ln(1 + \mu)}, \tag{1}$$

where μ = 255 and $-1 < x_p < 1$. This quantises the high resolution input to 256 possible values causing a loss in audio quality [1], however it makes the model feasible to train.

2.4. Model Details

Two losses are minimised during training with regards to an input sample x_p at time step t from augmented segment X_p: (1) domain confusion loss $\mathcal{L}_{dc}$,

$$\mathcal{L}_{dc} = \sum_{p} \sum_{x_p} \ell_{ce}(C(E(x_p)), p), \tag{2}$$

which applies cross entropy loss ℓ_{ce} to each element of the output Z and the corresponding percussion label p, and (2) autoencoder loss $\mathcal{L}_{ac}$,

$$\mathcal{L}_{ac} = \sum_{p} \sum_{x_p} \ell_{ce}(D_p(E(x_p)), x_p). \tag{3}$$

The decoder D_p is an AR model conditioned on the output of the shared encoder E. Final loss $\mathcal{L}$ is defined as:

$$\mathcal{L} = \mathcal{L}_{ac} - \lambda \mathcal{L}_{dc}, \tag{4}$$

where λ is a scaling factor for $\mathcal{L}_{dc}$ described in Section 2.2.

An Adam optimiser with the initial learning rate of 0.001, and a decay factor of 0.98 is used. The model is trained for 10 epochs and 50,000 iterations in total, where each iteration takes a random mini-batch of 8 randomly pitch shifted X_p segments. All weights in the network are initialised using Xavier initialisation [26].

The system consists of a naïve WaveNet architecture [27], where $O(2^L)$ is the overall computation time for a single output with L total number of layers of the WaveNet autoencoder outlined in Section 2.1. The system is implemented using the Tensorflow Python library[1] and was trained on an NVIDIA Tesla M40 computing processor.

[1] `https://www.tensorflow.org/`

2.5. Generation

During the transformation of an unaugmented audio sample y from any source domain, the autoencoder of domain p with its corresponding D_p is used to output the new sample $\hat{y}$ through:

$$\hat{y} = D_p(E(y)). \tag{5}$$

3. EXPERIMENTS

Two individual experiments are conducted to evaluate the rhythmic modification characteristics of the system as well as the translation quality through numerical analysis. In the first experiment (Section 3.2), the rhythmic similarity between source and drum translated audio pairs are evaluated by measuring the cosine similarity between rhythmic envelopes. For the second experiment (Section 3.3) a numerical analysis of audio degradation by [8] is presented measuring the Pearson correlation between the translated and source audio.

3.1. Training Data

For all experiments, the system is trained using raw audio waveforms as input features. Kick, snare and hi-hat samples used in the creation of different domains p for training are selected from a variety of different sample libraries included in the Ableton Live 10 Suite software. Drum samples are reduced to mono 16-bit WAV files and downsampled from 44.1 kHz to 22.05 kHz. To ensure each domain is accurately represented by the model, silence is removed from the start of each audio recording. The additional domains, which represent when two or more percussion instruments are played simultaneously (e.g., kick drum and hi-hat together), are artificially synthesised by overlaying randomly selected percussion samples.

In total, there are 1000 drum recordings for each domain resulting in a total dataset size of 7000. The mean duration of the drum recordings is 4862 samples (i.e., 0.22s). The resulting segments are normalised and zero-padded to a constant length T. Audio samples longer than T are trimmed, with a linear fade applied to the last 1000 samples. To mitigate the low resolution at ranges near ± 1 that results from the μ-law encoding stage, the amplitudes of all audio segments are randomly scaled between 0.5 and 0.6.

3.2. Experiment 1: Rhythmic Similarity

The purpose of the first experiment is to evaluate the capacity of the proposed system for preservation of rhythmic patterns during transformations. A variety of different drum loops are used as the source for this experiment. The events in each drum loop are manually labelled, then translated into an output drum loop where domains correspond to the source. The cosine similarity is measured between the rhythmic envelopes of source and output transformation pairs.

In order to extract the rhythmic envelope R from each file, the short-time Fourier transform of each audio file is computed using an n-length Hanning window ($n = 2048$) with a hop size of $\frac{n}{4}$. The standard spectral difference envelopes are then calculated as the sum of the first-order difference between each spectrogram (e.g., [28]). The resulting envelopes are then normalised between 0 and 1. Following the approach described in [7], the cosine similarity Φ between rhythmic envelopes of inputs and their transformations is calculated as follows:

$$\Phi_{\alpha,\beta} = \frac{R_\alpha \cdot R_\beta}{\|R_\alpha\|\|R_\beta\|}, \tag{6}$$

where α and β represent source input and drum translated output respectively.

This experiment is conducted using 20 drum loops selected from the Apple Logic Pro sample library, resulting in 20 transformations to be evaluated. The drum loops are chosen to reflect a variety of different drum patterns and styles, with multiple domains reflected in each loop. The drum loops have a mean duration of 3.5s and tempo ranges from 100 beats per minute (BPM) to 170 BPM. All loops are in the mono WAV format and are resampled to 22.05 kHz with 16-bit resolution.

3.3. Experiment 2: Translation Quality

In the second experiment, the timbral differences linked to the μ-law quantisation and the limited WaveNet model capacity are analysed. As an objective measure of audio quality, following [8], the Pearson correlations between the source and the drum translated audio are plotted. The correlations are visualised over a logarithmically scaled range of frequencies between 32–10548 Hz and calculated using 101-bin log-magnitude constant-Q transforms (CQT) with 12 bins per octave starting from C1 ($\approx$32.70Hz) and hop size of 512 using LibROSA [21]. For the purpose of this experiment, seven audio tracks are created using drum samples selected from the Logic Pro sample library. Each track contains ten drum samples with different timbres from the corresponding domains p (e.g., ten kick drum samples). All ten samples are translated into their corresponding domains (e.g., ten kick samples transformed into ten kick samples), resulting in a total of 70 separate drum translations.

4. RESULTS AND DISCUSSION

4.1. Rhythmic Similarity

The mean cosine similarity score across the 20 transformations in the rhythmic similarity experiment is 0.75. This indicates that in most cases, the proposed system is capable of preserving rhythmic structure when translating from source to target domain. The highest rhythmic similarity score is 0.96, which suggests that for this transformation the majority of the target domains were successfully translated resulting in a rhythmic envelope that is almost identical to the input. Transformations receiving low similarity scores failed to translate parts of the input resulting in dissimilar rhythmic envelopes. The lowest performing transformation has a rhythmic similarity score of 0.48 due to a frequent failure in properly translating domain `k`. The possible reasons for different artifacts and behaviours of the synthesised audio are discussed using two translation outputs presented in Figure 2.

To better understand the issues present in the translated outputs, two examples of source and output waveforms have been plotted in Figure 2. The cosine similarity of the translation A is 0.79 indicating some differences in their rhythmic similarities. In the first beat of the output example A, it can be seen that the waveform of the source kick drum was unnaturally smeared, whereas the source kick drum A from beat 3 was not translated at all. Both, the smearing and the missing event are examples of the system failing to correctly translate to domain `k` that is a cause of multiple failed transformations in Experiment 1. In example B, all drum

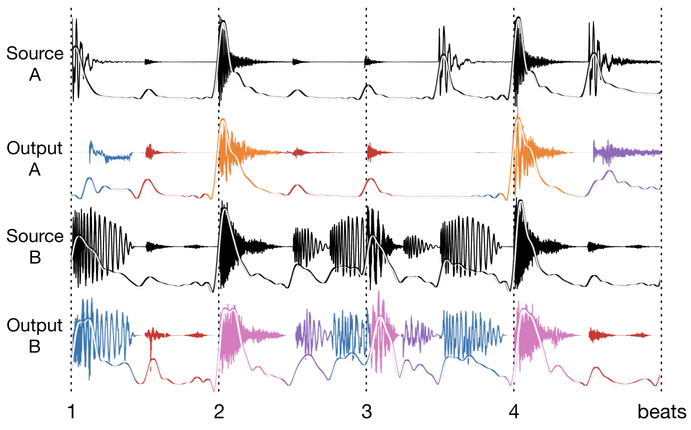

Figure 2: *Example translations generated from two sources (A and B), with spectral difference functions as solid lines over each waveform. Output colours correspond to target drum domains (e.g., blue represents kick drum translations).*

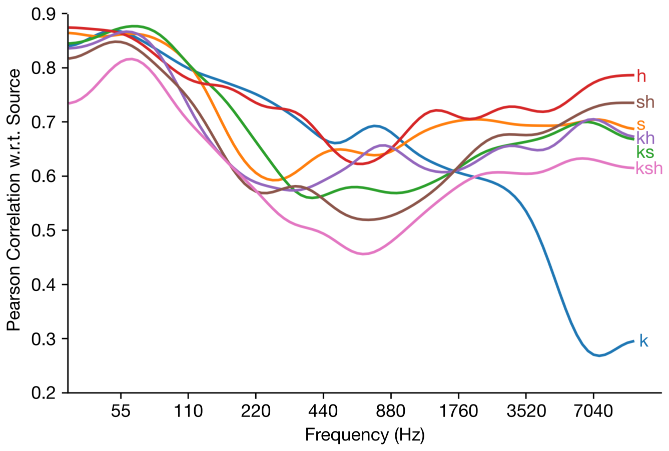

Figure 3: *Smoothed mean Pearson correlations between the translated and source audio for all drum domains.*

domains were effectively translated resulting in a high rhythmic similarity score of 0.96. Is is likely that the timbral characteristics of source B contributed to a more suitable translation output that shared, to a certain extent, some of the latent representations of different drum domains. For instance, the output kick drum translation from source B in beat 1 exhibits a noisy attack but then effectively transforms the expected low frequencies, while still notably changing the characteristics of the source thus creating a new kick sound.

4.2. Translation Quality

Figure 3 presents the results using evaluation methodology described in Section 3.3. For each domain, the mean Pearson correlation is taken for ten different source and output log-magnitude CQT spectrograms. The correlation values across all frequencies are smoothed using a median filter. High Pearson correlations indicate that translation quality for a particular domain is well preserved, whereas lower correlations indicate larger quality degradation from the source.

For all domains, the results indicate that frequency information below 100 Hz is well preserved. Across all domains there is a significant drop in correlation for low-mid frequencies between 220–1760 Hz. In comparison to other instruments, domain `k` (blue curve) maintains higher correlation for these low-mid frequencies however, there is a significant roll-off for high frequencies above 1760 Hz due to noise introduced by the model. Domain `ksh` (pink curve) represents when a kick, snare and hi-hat are all played simultaneously and has the lowest Pearson correlation across all frequency bands demonstrating that this domain was most difficult to reconstruct accurately due to complexity introduced by the three instruments.

Drum translations used in Experiment 1 as well as other examples can be found on the supporting website for this project.[2] As can be heard from many of the drum translations, the proposed system is capable of generating samples indicative of the intended target domains; however, a considerable amount of noise is produced in the transformations. It is assumed that the audio quality of the system is currently restricted by the limited amount of data on which it has been trained; by using more varied training data that has not been synthesised artificially (e.g., audio samples corresponding to real life audio loops or drum recordings), it is expected that the system would be capable of producing more accurate transformations. In addition to this, the limited size of the model restricts the number of possible transformations.

4.3. Automatic and Semi-automatic Drum Translation

An automatic and semi-automatic extension to drum translation has been proposed, in which a user can choose to automatically label different drums in the audio input with ADTLib [29] or manually annotate the input prior to translating it into various target drum domains. In this transformation, the length of segments attributed to the labelled domain corresponds to detected inter-onset-intervals. A novice or expert user can use such transformation in a music composition scenario. Once all onsets are processed the system translates annotated source audio segments into specified drum domains.

5. CONCLUSIONS

A drum translation technique that explores the rhythmic and timbral capabilities of generative audio synthesis with WaveNet autoencoders has been presented. In this transformation, an input file is transformed so that it sounds as if it were performed by different drum instruments. Two experiments were conducted to assess the rhythmic accuracy and overall audio quality of the transformations with respect to different drum instruments. The experimental results demonstrate that the system produces rhythmically accurate transformations, while there exist significant aspects of the proposed transformation that contribute to the creation of audio artifacts and added noisiness that could be mitigated with a network architecture of larger capacity. The memory requirements of the current model limit the number of available residual channels and the number of layers which can be overcome by architectures such as Parallel WaveNet [30]. Another possible avenue of future

[2] `https://maciek-tomczak.github.io/maciek.github.io/Drum-Translation-for-Timbral-and-Rhythmic-Transformation`

work would explore a broader range of signals outside the tested datasets as well as other applications of diverse adversarial losses to audio to improve translation audio quality.

6. REFERENCES

[1] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu, "WaveNet: A generative model for raw audio," *Computing Research Repository, abs/1609.03499*, 2016.

[2] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan, "Neural audio synthesis of musical notes with WaveNet autoencoders," in *Proceedings of the International Conference on Machine Learning*, pp. 1068–1077, 2017.

[3] Emmanuel Ravelli, Juan P. Bello, and Mark Sandler, "Automatic rhythm modification of drum loops," *IEEE Signal Processing Letters*, vol. 14, no. 4, pp. 228–231, 2007.

[4] Jason Hockman, Juan P. Bello, Matthew E. P. Davies, and Mark D. Plumbley, "Automated rhythmic transformation of musical audio," in *Proceedings of the International Conference on Digital Audio Effects*, pp. 177–180, 2008.

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

[6] Shun Sawada, Satoru Fukayama, Masataka Goto, and Keiji Hirata, "TransDrums: A drum pattern transfer system preserving global pattern structure," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 391–395, 2019.

[7] Maciek Tomczak, Carl Southall, and Jason Hockman, "Audio style transfer with rhythmic constraints," *in Proceedings of the International Conference on Digital Audio Effects*, pp. 45–50, 2018.

[8] Jong W. Kim, Rachel Bittner, Aparna Kumar, and Juan P. Bello, "Neural music synthesis for flexible timbre control," *Computing Research Repository, abs/1811.00223*, 2018.

[9] Noam Mor, Lior Wolf, Adam Polyak, and Yaniv Taigman, "A universal music translation network," *Computing Research Repository, abs/1805.07848*, 2018.

[10] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, R. J. Skerrv-Ryan, et al., "Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 4779–4783, 2018.

[11] Andrew Gibiansky, Sercan Arik, Gregory Diamos, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou, "Deep voice 2: Multi-speaker neural text-to-speech," in *Proceedings of the Neural Information Processing Systems*, pp. 2962–2970, 2017.

[12] Sander Dieleman, Aäron van den Oord, and Karen Simonyan, "The challenge of realistic music generation: Modelling raw audio at scale," in *Proceedings of the Neural Information Processing Systems*, pp. 8000–8010, 2018.

[13] Rachel Manzelli, Vijay Thakkar, Ali Siahkamari, and Brian Kulis, "Conditioning deep generative raw audio models for structured automatic music," *Computing Research Repository, abs/1806.09905*, 2018.

[14] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio, "A recurrent latent variable model for sequential data," in *Proceedings of the Neural Information Processing Systems*, pp. 2980–2988, 2015.

[15] Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," *Computing Research Repository, abs/1612.07837*, 2016.

[16] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aäron van den Oord, Sander Dieleman, and Koray Kavukcuoglu, "Efficient neural audio synthesis," *Computing Research Repository, abs/1802.08435*, 2018.

[17] Sicong Huang, Qiyang Li, Cem Anil, Xuchan Bao, Sageev Oore, and Roger B. Grosse, "TimbreTron: A WaveNet (CycleGAN (CQT (Audio))) pipeline for musical timbre transfer," *Computing Research Repository, abs/1811.09620*, 2018.

[18] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the International Conference on Computer Vision*, pp. 2223–2232, 2017.

[19] Noam Mor, Lior Wolf, Adam Polyak, and Yaniv Taigman, "A universal music translation network," in *Proceedings of the International Conference on Learning Representations*, 2019.

[20] Patricio López-Serrano, Matthew E. P. Davies, and Jason Hockman, "Break-informed audio decomposition for interactive redrumming," *Late-Breaking Demo Session Abstract in the International Society for Music Information Retrieval*, 2018.

[21] Brian McFee, Colin Raffel, Dawen Liang, Daniel P. W. Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the Python in Science Conference*, pp. 18–25, 2015.

[22] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[24] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," *Computing Research Repository, abs/1511.07289*, 2015.

[25] ITU-T. Recommendation G. 711, "Pulse code modulation (PCM) of voice frequencies," 1988.

[26] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

[27] Tom Le Paine, Pooya Khorrami, Shiyu Chang, Yang Zhang, Prajit Ramachandran, Mark A. Hasegawa-Johnson, and Thomas S. Huang, "Fast WaveNet generation algorithm," *Computing Research Repository, abs/1611.09482*, 2016.

[28] Simon Dixon, "Onset detection revisited," in *Proceedings of the International Conference on Digital Audio Effects*, pp. 133–137, 2006.

[29] Carl Southall, Ryan Stables, and Jason Hockman, "Automatic drum transcription using bi-directional recurrent neural networks.," in *Proceedings of the International Society of Music Information Retrieval Conference*, pp. 591–597, 2016.

[30] Aäron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis Cobo, Florian Stimberg, et al., "Parallel WaveNet: Fast high-fidelity speech synthesis," *Computing Research Repository, abs/1711.10433*, 2017.