STYLE-BASED DRUM SYNTHESIS WITH GAN INVERSION

Jake Drysdale, Maciek Tomczak, Jason Hockman
Sound and Music Analysis (SoMA) Group, Digital Media Technology Lab (DMT Lab)
School of Computing and Digital Technology
Birmingham City University
Birmingham, UK
jake.drysdale@bcu.ac.uk

ABSTRACT

Neural audio synthesizers exploit deep learning as an alternative to traditional synthesizers that generate audio from hand-designed components, such as oscillators and wavetables. For a neural audio synthesizer to be applicable to music creation, meaningful control over the output is essential. This paper provides an overview of an unsupervised approach to deriving useful feature controls learned by a generative model. A system for generation and transformation of drum samples using a style-based generative adversarial network (GAN) is proposed. The system provides functional control of audio style features, based on principal component analysis (PCA) applied to the intermediate latent space. Additionally, we propose the use of an encoder trained to invert input drums back to the latent space of the pre-trained GAN. We experiment with three modes of control and provide audio results on a supporting website.

1. INTRODUCTION

One of the chief skills harnessed by electronic music (EM) producers is the ability to select and arrange suitable drum sounds. The integration of drum sounds into an EM composition may be achieved either through the time-consuming task of browsing sound libraries for an appropriate drum recording or alternatively through the use of traditional drum synthesizers, which require mastery over a large number of parameters, and provide only limited control over sound generation. More recently, neural drum synthesis [1–3] has been proposed to allow EM producers to interactively generate and manipulate drum sounds based on personal sound collections.

As compared to traditional drum synthesis techniques (e.g., subtractive, FM), control parameters are learned through an unsupervised process. Neural audio synthesis enables intuitive exploration of a generation space through a compact latent representation. A crucial requirement of a well-trained latent representation is its ability to generate audio with musically-meaningful control over the output. In the case of neural drum synthesis, previous methods have been proposed to enable continuous semantic control over generations by supplying high-level conditional information based on a chosen set of audio features [2–4].

In this paper, an unsupervised approach for deriving useful synthesis parameters is used as an alternative to selecting and extracting a fixed set of audio features. The proposed system is based on [1], and is extended with a conditional style-based generator network [5] for drum style (i.e., timbre) transformations. A mapping between a distribution of labelled drum sounds and a low-dimensional latent space is learned, providing high-level control over generation. The system operates directly on waveforms and leads to an unsupervised separation of high-level features (e.g., pitch, envelope shape, loudness), and enables intuitive, layer-wise control of the synthesis. Additionally, we introduce a novel approach to audio reconstruction through GAN inversion—a set of techniques for inverting a given input back into the latent space of a pre-trained GAN [6]. Using the predicted latent code, an input can be reconstructed by the GAN and manipulated using directions found in its latent space.

2. SYSTEM OVERVIEW

Figure 1 provides an overview of the proposed system for neural drum synthesis, which is achieved using four networks: (1) mapping network, (2) generator, (3) discriminator, (4) encoder. In a style-based GAN formalisation [5, 7], latent space $\mathcal{Z}$ is transformed into an intermediate space $\mathcal{W}$ using mapping network $M : \mathcal{Z} \rightarrow \mathcal{W}$. To facilitate the functionality of the mapping network, G is modified to take a constant value as input, and an intermediate latent vector $\mathbf{w} \in \mathcal{W}$ is provided to each upsampling layer. The mapping network M is a conditional multilayer perceptron, that learns to create disentangled features that are integrated at each upsampling block of the generator network G through adaptive instance normalisation (AdaIN). The generator is trained to output audio waveforms given a constant input and latent vector $\mathbf{w}$ for each upsampling block with affine transform A. Gaussian noise is added to each upsampling block using per-layer scaling factors B to introduce stochastic variation at each layer. Discrimi-

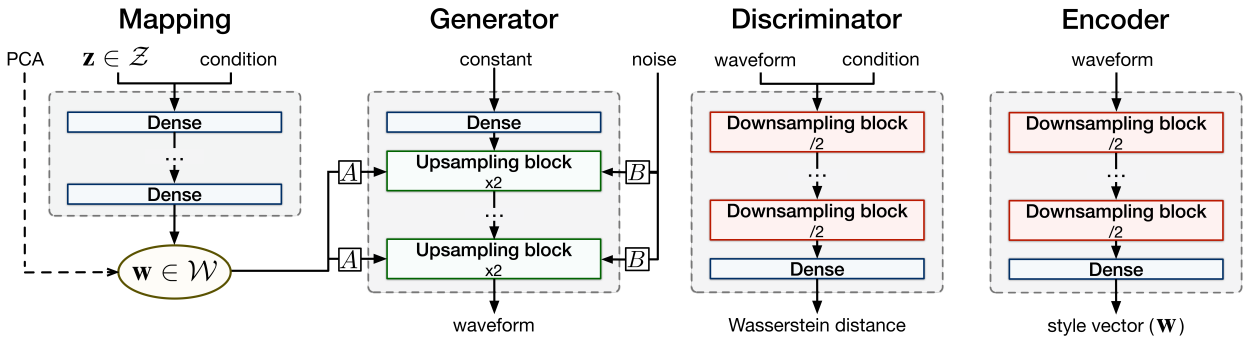

Figure 1. *Overview of proposed style-based drum synthesis system.*

nator D takes both a waveform and conditioning variable as an input and is trained to estimate the Wasserstein distance between the generated and observed distributions. All three networks are optimised simultaneously until G can produce waveforms that are indistinguishable from the observed training data.

The model was trained using a dataset of 9000 single drum hits selected from multiple commercially available sample libraries [1]. To increase the size of the dataset, individual drum sounds were augmented by pitch shifting between ± 3 semitones, resulting in a total of 63000 samples. The system uses WGAN-GP [8] training strategy to minimise the Wasserstein distance [9] between the training data distribution and generated data distribution. The model is trained using Adam optimiser [10] with a learning rate of 0.002 for the D and G networks and a minibatch size of 64 on an NVIDIA 2080ti GPU for 200k iterations.

2.1 Audio Inversion Network

Based on recent advances in the image domain [6], an encoder network E is trained separately to embed a given waveform into the intermediate latent space of the pre-trained generator. The predicted latent vectors are fed into the generator to synthesise drum sounds with similar characteristics to the input waveform. E replicates the architecture of the discriminator network D; however, the model is unconditional and its final dense layer has been modified to have has 128 output units to match the dimensionality of $\mathcal{W}$. Using a dataset of 10000 drum sounds generated with pre-trained network G, E is trained to minimise the MSE between the ground truth latent vectors and the predicted latent vectors.

2.2 Principal Feature Directions

PCA identifies patterns within the intermediate latent space $\mathcal{W}$, deriving a set of coordinates that emphasise variation in timbre. G feature controls are achieved by layer-wise perturbation along the principal directions. Following [11], principal axes of $p(\mathbf{w})$ are identified with PCA. N random vectors are sampled from Z and the corresponding $\mathbf{w}_l$ values are computed with M. At inference, PCA can be computed on $\mathbf{w}_l$ to obtain a basis V for $\mathcal{W}$. The principal components of $\mathbf{w}_l$ can then be used to control features of the generator by varying PCA coordinates scaled by control parameter g such that $\mathbf{w}' = \mathbf{w} + Vg$. Each entry g_i is initialised with zeros until modified by a user.

2.3 Synthesis Control Parameters

Style-based drum synthesis with GAN inversion allows the user to interact with the system parameters in three ways. In the first approach, drum synthesis can be controlled by sampling from the intermediate latent space and exploring the timbral characteristics with a preset number of style faders (i.e., PCA coordinates at each layer). In the second approach, the user can input a single drum sample to the encoder and modify its characteristics with style faders. In the third approach, the encoder can be used to reconstruct two arbitrary drum sounds, and various interpolation techniques may be incorporated for style transformation. Additionally, in each control method, Gaussian noise can be introduced into individual generator layers to modify the amount of stochastic variation within network layers. This is useful for shaping the high-frequency content of the generated drum sounds—especially for cymbals and snares.

3. EXPERIMENTAL RESULTS

For a demonstration of the generated drum sounds and synthesis parameters, we invite the reader to listen to the results and experiment with the code available on the accompanying website. [1] A python script is provided, which loads the networks pre-trained weights and enables utilisation of the synthesis control parameters described in Section 2.3. By traversing the latent space using the controls provided, a user can explore a space of different drum sounds. The audio examples demonstrate the systems capacity to generate a variety of different drum sounds, manipulate timbral characteristics and perform transformations between different inputs such as beatboxing and breakbeats. Although it is currently difficult to anticipate the exact effect that each principal direction has on the generated drum sounds, the directions correlate to changes in timbre, pitch and amplitude envelope.

[1] `https://jake-drysdale.github.io/blog/stylegan-drumsynth/`

4. REFERENCES

[1] J. Drysdale, M. Tomczak, and J. Hockman, "Adversarial synthesis of drum sounds," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, 2020.

[2] A. Ramires, P. Chandna, X. Favory, E. Gómez, and X. Serra, "Neural percussive synthesis parameterised by high-level timbral features," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 786–790, 2020.

[3] J. Nistal, S. Lattner, and G. Richard, "DrumGAN: Synthesis of drum sounds with timbral feature conditioning using generative adversarial networks," 2020.

[4] M. Tomczak, M. Goto, and J. Hockman, "Drum synthesis and rhythmic transformation with adversarial autoencoders," in *Proceedings of the ACM International Conference on Multimedia*, pp. 2427–2435, 2020.

[5] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

[6] W. Xia, Y. Zhang, Y. Yang, J.-H. Xue, B. Zhou, and M.-H. Yang, "GAN inversion: A survey," *arXiv preprint arXiv:2101.05278*, 2021.

[7] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp.8110–8119, 2020.

[8] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein GANs," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pp. 5767–5777, 2017.

[9] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," in *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 214–223, 2017.

[10] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[11] E. Härkönen, A. Hertzman, J. Lehtinen, and S. Paris, "GANspace: Discovering interpretable GAN controls," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pp. 9841–9850, 2020.