ADVERSARIAL SYNTHESIS OF DRUM SOUNDS

Jake Drysdale, Maciek Tomczak, Jason Hockman

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

ABSTRACT

Recent advancements in generative audio synthesis have allowed for the development of creative tools for generation and manipulation of audio. In this paper, a strategy is proposed for the synthesis of drum sounds using generative adversarial networks (GANs). The system is based on a conditional Wasserstein GAN, which learns the underlying probability distribution of a dataset compiled of labeled drum sounds. Labels are used to condition the system on an integer value that can be used to generate audio with the desired characteristics. Synthesis is controlled by an input latent vector that enables continuous exploration and interpolation of generated waveforms. Additionally we experiment with a training method that progressively learns to generate audio at different temporal resolutions. We present our results and discuss the benefits of generating audio with GANs along with sound examples and demonstrations.

1. INTRODUCTION

Sample-based electronic music (EM) describes a variety of genres that emerged through advancements in audio production and digital sampling technologies. EM is mainly created through the use of digital audio workstation (DAW) software for arranging and manipulating short audio recordings, commonly referred to as samples. Early sampling technologies (e.g., Akai S950) were limited by a small amount of memory; however, this constraint stimulated creativity, artistic choices, and new genres of music. Considering the abundance of free and affordable audio sample libraries available at present, there is the potential for an EM producer's personal collection of samples to become unwieldy and therefore difficult to navigate and maintain.

Sample selection is an integral part of the EM production workflow and is one of the key skills harnessed by EM producers. The selection of samples in this context is a meticulous retrieval task involving careful listening for key subjective attributes (e.g., warmth, boominess) of particular timbral features. Online sample libraries such as Splice[1] and Loopmasters[2] have well-annotated databases with high quality sounds; however, when a producer is searching a collection for an exact sample or a sample with certain characteristics (e.g., bass-heavy kick), the sound selection process can be tedious and labor-intensive.

[1]`https://splice.com/`
[2]`https://www.loopmasters.com/`

In this paper, a system is presented that allows EM producers to interactively generate and fine tune novel audio sounds based on their own personal collections. The system is based on a generative adversarial network, which learns a mapping between a collection (i.e., dataset) of labelled drum sounds and a low-dimensional latent space that provides high-level control of the input data distribution.

1.1. Background

Advancements in generative modelling have allowed for the development of novel tools for the generation and manipulation of audio. Generative models learn the underlying probability distribution of a given dataset and produce new data based on example observations. Generative methodologies include generative adversarial networks (GANs) [1], autoencoders [2] and autoregressive networks [3]. Autoencoders map high-dimensional data distributions onto low-dimensional latent spaces and reconstruct the output from this representation using a decoder. Several generative models using autoencoders have been proposed for the task of generalised musical audio generation including autoregressive (AR) models (e.g., [4, 5]) and non-AR models (e.g. [6, 7, 8]). AR models for raw audio synthesis have the capacity to generate high fidelity audio, yet this comes at the cost of slow generation and the inability to learn compact latent space representations. An alternative solution is found in GANs, a subset of non-AR generative models, which map low-dimensional latent spaces to complex data distributions through an adversarial training strategy [1]. The generator learns to produce realistic synthesized data from a prior distribution, while the discriminator learns to correctly classify real and synthetic data. GANs can be conditioned on additional information (e.g., pitch, instrument class) enabling high-level control over data generation [9]. Unlike AR models, GANs are capable of parallelised training and generation. However, GANs require much larger models to generate longer audio recordings, becoming computationally expensive. Thus, GANs are well-suited for the synthesis of short audio recordings such as drum sounds.

Donahue et al. [10] were the first apply adversarial learning to musical audio using a modified deep convolutional GAN [11] that operates on raw audio data. Alternatively, Engel et al. [12] proposed GANSynth, an adversarial approach to audio synthesis that utilised recent improvements in the training stability of GANs [13, 14, 15]. Musical notes are conditioned with labels representing the pitch content and are modelled as log magnitude and instantaneous frequency spectrograms, which are used to approximate the time-domain signal. More recently, Engel et al. [16] achieved high resolution audio generation without the need for large AR models or adversarial losses through a modular approach to generative audio modeling that integrates digital signal processing elements into a neural network.

Specific to the generation of drum sounds, Aouameur et al.

[8] used a conditional Wasserstein autoencoder to generate audio spectrograms that are inverted to audio through a multi-head CNN. Ramires et al. [17] synthesized percussion sounds with high-level control over timbral characteristics using Wave-u-net [18]. Tomczak et al. [19] proposed a method for joint synthesis and rhythm transformation of drum sounds by combining the use of adversarial autoencoders with a Wasserstein GAN adversarial framework.

1.2. Motivation

In this paper, a system for synthesising drum samples is presented, which is suitable for generating novel drums sounds based on a producers personal sample collection. The system is designed to be lightweight, in that it can learn to generate high-quality audio when trained using a small amount of data. High-level conditioning organises drum sounds into specific categories, while a compact latent space with low dimensionality is used for intuitive synthesis control to output a variety of different drum sounds. In addition, interpolating the compact latent space of a learned generative model provides an intuitive way for EM producers to morph between generated drum samples when composing new grooves and rhythms.

The system is realised through a conditional Wasserstein generative adversarial network trained with a small dataset of labelled drums sounds. Conditioning is achieved with the three main percussion instruments from the common drum kit—that is, kick drum, snare drum, and cymbals—and it can generate a diverse range of sounds when trained on a relatively small dataset of short audio recordings.

By varying the input latent vector, large or subtle variations can be made to the timbral characteristics of the output. In order to reduce training time, a progressive growing training methodology similar to [13] is considered, in which audio is incrementally generated at increasingly higher temporal resolutions.

The remainder of this paper is structured as follows: Section 2 presents our proposed method for drum sound generation. Section 3 presents our training procedure and dataset processing, and Section 4 provides the results and discussion. Conclusions and suggestions for future work are presented in Section 5.

2. METHOD

The proposed approach to drum synthesis builds upon the architecture of WaveGAN [10] but is designed specifically for conditional audio generation of a variety of different drum sounds. Figure 1 presents a general overview of the proposed system. Generator G is trained to generate audio signals given a latent vector z and a conditional variable y, and discriminator D is trained to estimate the Wasserstein distance between the generated and observed distributions. Both networks are optimised simultaneously until G can produce drum samples that are indistinguishable from the observed training data.

The original GAN framework as proposed by [1] defines an adversarial game between generator network G and discriminator network D. G is used to learn mappings from a noise space Z to drum data space X. $Z = \mathbb{R}^{d_z}$, where d_z is a hyperparameter that controls the dimensionality of Z. Latent variables $z \in Z$ are sampled from a known prior $p(z)$, which is modelled with a simple distribution (e.g., Gaussian, Uniform). X is the drum data space that represents the input to D or output of G. As training data, drum samples D are drawn from a real distribution $p_D(x)$. By

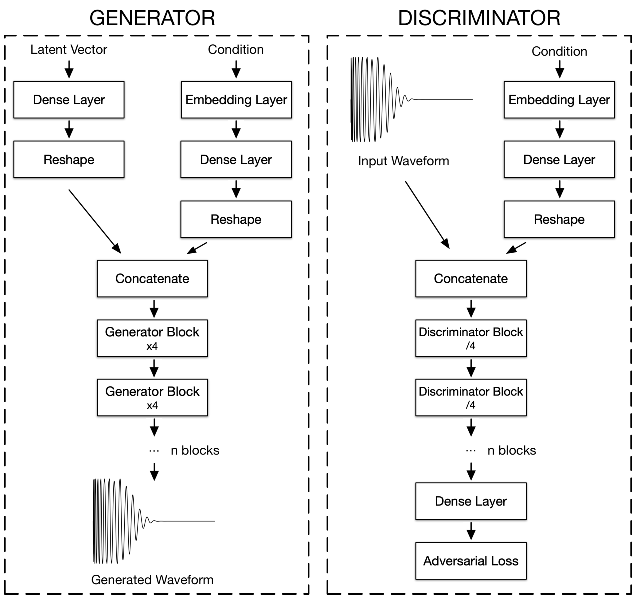

Figure 1: *Overview of proposed system for drum synthesis: Generator G (left) is trained to generate audio given a latent vector z and conditioning variable y. Discriminator D (right) is trained to minimise the Wasserstein distance between the generated distribution and the observed distribution.*

sampling from $p(z)$, G can be used to output drums that represent a synthetic distribution $q(x)$. Following the more general formulation introduced in [20], the GAN learning problem aims at finding a min-max optimisation of objective V between the pair of G and D (i.e., Nash equilibrium), of the value function defined as:

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim p_D(x)}[f(D(x))] + \mathbb{E}_{z\sim p(z)}[f(-D(G(z)))], \quad (1)$$

where $\mathbb{E}[\cdot]$ denotes expectation, and $f : \mathbb{R} \to \mathbb{R}$ is a concave function. G is trained to output $q(x)$ as close to $p_D(x)$ as possible. D is trained to distinguish between real data P_X and synthesised data $q(x)$. Convergence occurs when G can mislead D by generating synthesized samples that are indistinguishable from real samples.

Training GANs correctly utilising the original formulation is difficult and prone to mode collapse, resulting in reduced sample variability. To help stabilise training, Arjovsky et al. [14] suggest minimising the Wasserstein distance between the generated and observed distributions.

D is modified to emit an unconstrained real number rather than a probability value to recover the traditional GAN [1] formulation $f(x) = -log(1 + exp(-x))$, where f is the logistic loss. This convention slightly differs from the standard formulation in that the discriminator outputs the real-valued *logits* and the loss function would implicitly scale this to a probability. The Wasserstein GAN is achieved by taking $f(x) = x$. Within this formulation, f has to be a 1-Lipschitz function and D is trained to assist in computing the Wassertein distance, rather than to classify samples as real or fake. To enforce the Lipschitz constraint, Arjovsky et al. [14] suggest the application of a simple clipping function to restrict the maximum weight value in f. To avoid subsequent difficulties in optimisation (i.e., exploding or vanishing gradients), the authors in [15] utilised a gradient penalty parameterised by penalty coefficient λ to enforce the constraint.

In the conditional formulation of the GAN, the G and D networks use additional input layers with labels y. The updated objective function can be stated as:

$$\min_G \max_D V(G,D) = \mathbb{E}_{x,y\sim p_D(x,y)}[f(D(x))] + \mathbb{E}_{y\sim p(y),z\sim p(z)}[f(-D(G(z,y),y))], \quad (2)$$

where $p(y)$ is the prior conditioning distribution. Conditioning the system on labels allows for targeted generation of drum sounds from a specific category. Methods for categorical conditioning commonly involve encoding conditional labels as one-hot vectors and concatenating them with the latent code [9]; however, this can lead to undesirable behaviour such as cross-over between classes. Following [21], an embedding space Y is used to condition the model on external information, where $Y = \mathbb{R}^{d_Y}$, where d_Y is a hyperparameter used to control the dimensionality of Y.

2.1. Model Details

In order to learn an instrument-specific encoding, conditioning variable y is passed through an embedding layer with a dimensionality w, such that each of the three drum classes are mapped to a different w-element vector representation that is learned by G ($w = 50$). The embedding layer and latent vector are then scaled to the initial size of the network using a dense layer, then concatenated together and passed through a series of upsampling blocks to output a generated waveform. Each upsampling block consists of one-dimensional nearest neighbour upsampling with a stride of 4, a one-dimensional convolutional layer with a kernel length of 25, and a ReLU activation. Thus, at each block the number of audio samples is increased by a factor of 4 with the output layer passed through a $tanh$ activation.

Discriminator network D mirrors the architecture in G. D takes an audio signal and conditioning variable y. In D, y is passed to an identical embedding layer to that in G and is scaled to the size of the input waveform using a dense layer and reshaping. This representation is then concatenated with the input waveform and passed through a series of downsampling blocks. Each downsampling block consists of a convolutional layer with a stride of 4 and kernel length of 25, a leaky ReLU activation (α = 0.2). Thus, at each stage of the discriminator the input waveform is decreased by a factor of 4. The final layer of D is a dense layer with a linear activation function that outputs the authenticity of the input audio sample through the Wasserstein distance.

Upsampling in generator networks is known to cause periodic checkerboard artifacts when synthesising images [22]. When generating raw audio, checkerboard artifacts can be perceived as pitched noise that degrades the overall audio quality. An optimisation problem can occur when D learns to reject generated audio with artifact frequencies that always occur at a particular phase. Donahue et al. [10] introduced a phase shuffle module that randomly perturbs the phase at each layer of D. Phase shuffle forces D to become invariant to the phase of the input waveform and is controlled by hyperparameter s that perturbs the phase of a layer's activations by -s to s samples (s = 2).

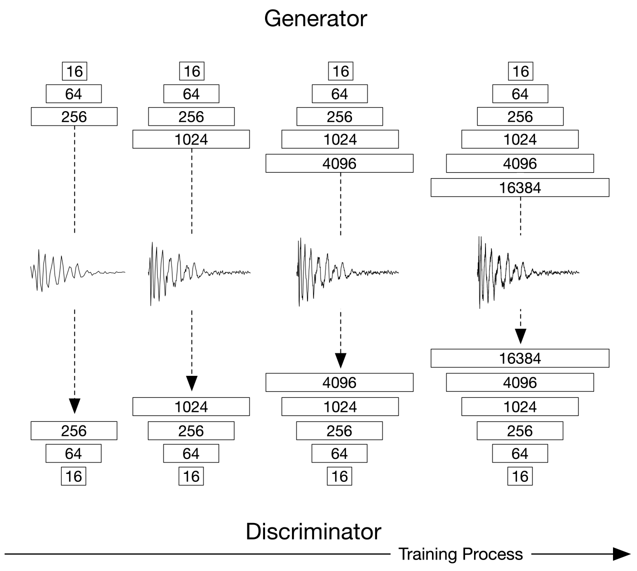

Figure 2: *Progressive growing procedure, in which D and G begin learning with low resolution audio resolution of 256 samples. As training advances new layers are added to the models to incrementally increase the number of samples by a multiple of 4 thus, learning higher frequencies as training progresses.*

3. TRAINING

3.1. Network training

In order to optimise Equation 2 we use alternating updates between networks G and D. At each training iteration, the parameters of network D are updated k times for each G parameter update ($k = 5$). The model is trained using the Adam optimiser [23] with a learning rate $2e$–4, $\beta_1 = 0.5$, $\beta_2 = 0.99$ for a 2000 epochs and 50000 iterations in total, where each iteration takes a mini-batch of 64 examples. The model is trained using a gradient penalty coefficient (λ = 10). n upsampling and downsampling blocks are used to allow for the generation of T samples of audio. Following [10], the latent dimensionality d_z was initially set to 100 and a second model is trained with a lower dimensionality ($d_z = 3$) to explore the tradeoff between dimensionality and audio quality.

3.2. Progressive Growing

To reduce the length of training time, a progressive growing procedure is adopted during training. Following [13], the model is initially trained with downsampled input audio data, then learns to generate output at samplerates of incrementally higher quality. Figure 2 depicts the progressive growing procedure for networks D and G, which are trained on low resolution audio until stable. Additional layers are then added to support more audio samples and thus higher samplerates can be used to sample the audio. Higher frequencies are learned in successive epochs as training progresses. As in [10], the output size of layers grows in increments of 4 until the desired samplerate of is met. When training completes at its current resolution, networks are incrementally grown by adding a new set of layers to increase the resolution each time by a multiple of 4. Skip connections are used to connect the new block to the input of D or output of G and the newly added

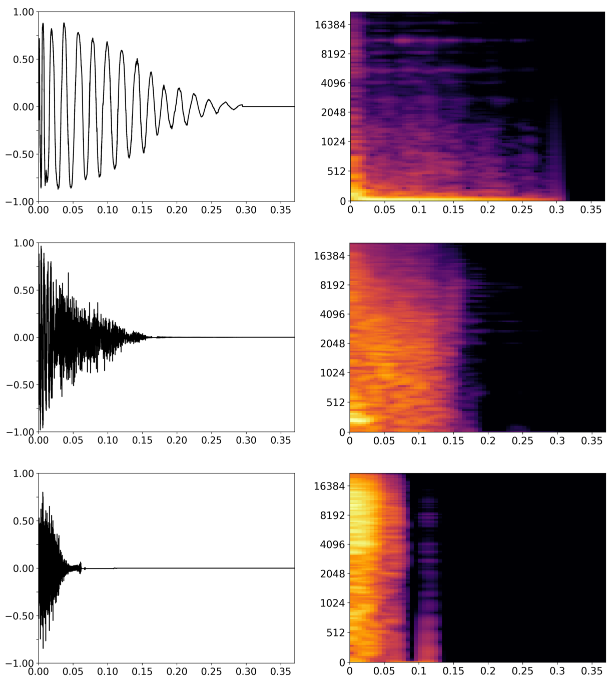

Figure 3: *Example waveform generations (left) and corresponding Mel-scaled log frequency spectrograms (right) for kick drum (top), snare drum (middle) and cymbal (bottom).*

layers are linearly faded in to prevent shocks from the sudden addition of a larger layer. Fading between layers is controlled by parameter r, which is linearly interpolated from 0 to 1. Learning the earlier simple models first helps to stabilise training and progressively learn finer high frequency details. G and D both start training with a short audio length of 256 samples. As training advances, we incrementally add layers to both G and D to increase the number of samples used in the generated audio.

3.3. Dataset

For all experiments, both networks are trained using raw audio waveforms. We compiled a dataset of drums sounds D selected from a wide variety of sample libraries (including Loopmasters and Splice). Sounds were categorised manually into p domains comprising kick, snare and cymbal samples. Each domain contains 3000 individual samples resulting in a total dataset size of 9000 samples. All samples are mono 16-bit PCM audio files sampled at 44.1kHz. Prior to preprocessing, the mean sample length of each audio file in the dataset was 18234 samples (i.e., 0.41s). In accordance with the network architecture in [10], we choose the training data input length to the nearest power of two ($T = 16384$) to satisfy the symmetric structure of networks G and D. Each training sample is trimmed or zero-padded to ensure a constant length of T samples. All waveforms are normalised and a short linear fade of samples is applied to the start and end of each waveform to ensure that they consistently begin and end at 0 amplitude.

4. RESULTS

A system for generative audio synthesis of drum sounds has been implemented as presented in Sections 2 and 3. We report on the system's capacity for generating coherent drum samples and provide an accompanying webpage[3] for examples of individual generated audio samples, interpolation experiments, and example usage within electronic music compositions. These examples allow for subjective evaluation of audio generation quality and waveform interpolation properties.

4.1. Generation Quality

Figure 3 presents examples of a kick drum, snare drum and cymbal generated through the system output. Informal listening tests were conducted to assess the generation quality of audio samples from each class. Conditioning the system with labels improves overall quality and omits overlap between classes. Generally, kick and snare drums can be more easily modelled by the system and are less prone to artifacts. As can be seen from the spectrograms in Figure 3, some checkerboard artifacts remain; however, this does not have a considerable effect on the overall perceived quality of the drum sounds and in most cases could be removed with simple post-processing (e.g., amplitude fading, equalisation). Inclusion of the progressive growing procedure results in both reduced training time and coherent audio generated at an earlier stage. Unfortunately, this results in an increase in artifacts present, degrading the perceived quality of the generations. Due to its fast training time, the progressive growing model could be used as a tool to preview drum samples from a large collection.

4.2. Latent Space Dimensionality

As the proposed system is intended to allow producers to interactively navigate a compact representation of audio sounds, experimentation was undertaken with a small latent dimensionality. The dimensionality of the latent space and its relationship to generation quality and diversity in GANs has yet to be thoroughly explored in literature.

For comparison, we provide a selection of randomly generated drum sounds from each domain using $d_z = 100$ and $d_z = 3$. Interestingly, the size of the latent space had little effect on output audio quality, following findings in other similar research [24]. Different values for d_z returned similar results, leading to our early conclusion that latent parameters up to rank three define the majority of parameter variance within the set of 100 dimensions; however, additional investigation is required to validate this.

4.3. Waveform Interpolation

The proposed system learns to map points in the latent space to the generated waveforms. The structure of the latent space can be explored by interpolating between two random points. Experiments with linear interpolation and spherical linear interpolation are provided on the accompanying webpage. The purpose of the spherical linear interpolation experiment is to ensure that the curving of the space is taken into account as linear interpolation assumes that the latent space is a uniformly distributed hypercube. When traversing the latent space, changes in audio quality are continuous

[3] `https://jake-drysdale.github.io/blog/adversarial-drum-synthesis/`

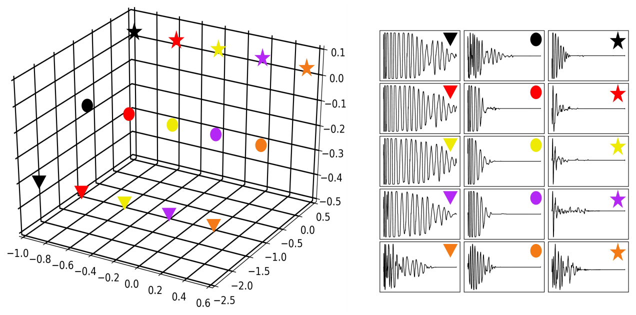

Figure 4: *Interpolation in the latent space for kick drum generation. Kick drums are generated for each point along linear paths through the latent space (left). Paths are colour coded and subsequent generated audio appears across rows (right).*

and without abrupt variation. Figure 4 demonstrates the transition between output kick drums when navigating linearly through the latent space. Timbral modifications can be made to a generation by making adjustments to latent variables Z. Larger steps in the latent space are perceptually equivalent to smoothly mixing amplitudes between distinct drum sounds whereas smaller adjustments result in subtle variations of timbral characteristics. Subtle variations in timbre could be a useful for humanizing programmed drum sequences to provide a more natural feel. While the effect each dimension in d has on the output can not be anticipated, many examples demonstrate consistent variations in pitch, envelope shape and the presence or omission of high and low frequencies. Spherical interpolation seem to result in a more abrupt change of timbral characteristics (e.g., alteration between different kick drum sounds) than linear interpolation.

5. CONCLUSIONS AND FUTURE WORK

A method for generative audio synthesis of drum sounds using a generative adversarial network has been presented. This system provides a music production tool that encourages creative sound experimentation. The results demonstrate the capacity of the conditional model to generate a wide variety of different class-specific drums sounds. High-level conditioning organises drum sounds into specific categories, while a compact latent space allows for intuitive synthesis control over output generations. The model is lightweight and can be trained using a reasonably small dataset to generate high-quality audio, further demonstrating the potential of GAN-based systems for creative audio generation. The experimental dataset could be replaced with an EM producers personal collection of samples and custom tags could be defined for conditioning. Future work will involve embedding the system into an audio plug-in that can be evaluated by EM producers in efforts to inform and improve the breadth of the design goals. The plug-in will be designed to have various parameters that enable navigation of the latent space.

6. REFERENCES

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pp. 2672–2680, 2014.

[2] Diederik P. Kingma and Max Welling, "Auto-encoding variational bayes.," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

[3] 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," in *Proceedings of the ISCA Speech Synthesis Workshop*, 2016.

[4] 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 (ICML)*, pp. 1068–1077, 2017.

[5] Maciek Tomczak, Jake Drysdale, and Jason Hockman, "Drum translation for timbral and rhythmic transformation," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, 2019.

[6] Philippe Esling, Axel Chemla-Romeu-Santos, and Adrien Bitton, "Bridging audio analysis, perception and synthesis with perceptually-regularized variational timbre spaces," in *Proceedings of the International Society of Music Information Retrieval Conference (ISMIR)*, pp. 175–181, 2018.

[7] Adrien Bitton, Philippe Esling, Antoine Caillon, and Martin Fouilleul, "Assisted sound sample generation with musical conditioning in adversarial auto-encoders," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, 2019.

[8] Cyran Aouameur, Philippe Esling, and Gaëtan Hadjeres, "Neural drum machine: An interactive system for real-time synthesis of drum sounds," in *Proceedings of the International Conference on Computational Creativity (ICCC)*, 2019.

[9] Mehdi Mirza and Simon Osindero, "Conditional generative adversarial nets," *arXiv:1411.1784*, 2014.

[10] Chris Donahue, Julian J. McAuley, and Miller S. Puckette, "Adversarial audio synthesis," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[11] Alec Radford, Luke Metz, and Soumith Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[12] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts, "GANSynth: Adversarial neural audio synthesis," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[13] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[14] Martin Arjovsky, Soumith Chintala, and Léon Bottou, "Wasserstein gan," in *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 214–223, 2017.

[15] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C. Courville, "Improved training of wasserstein gans," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pp. 5767–5777, 2017.

[16] Jesse Engel, Lamtharn (Hanoi) Hantrakul, Chenjie Gu, and Adam Roberts, "Ddsp: Differentiable digital signal processing," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[17] António Ramires, Pritish Chandna, Xavier Favory, Emilia Gómez, and Xavier 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.

[18] Daniel Stoller, Sebastian Ewert, and Simon Dixon, "Wave-u-net: A multi-scale neural network for end-to-end audio source separation," in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, pp. 334–340, 2018.

[19] Maciek Tomczak, Masataka Goto, and Jason Hockman, "Drum synthesis and rhythmic transformation with adversarial autoencoders," in *Proceedings of the ACM International Conference on Multimedia*, 2020.

[20] Vaishnavh Nagarajan and J. Zico Kolter, "Gradient descent gan optimization is locally stable," in *Proceedings of the Advances in neural information processing systems (NIPS)*, pp. 5585–5595, 2017.

[21] Emily L. Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus, "Deep generative image models using a Laplacian pyramid of adversarial networks," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pp. 1486–1494, 2015.

[22] Augustus Odena, Vincent Dumoulin, and Chris Olah, "Deconvolution and checkerboard artifacts," in *Distill*, 2016.

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

[24] Thomas Pinetz, Daniel Soukup, and Thomas Pock, "Impact of the latent space on the ability of GANs to fit the distribution," `https://openreview.net/forum?id=Hygy01StvH`, 2019.