Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)
Demonstrations Track

Pattern-Based Music Generation with Wasserstein Autoencoders and PR^{C}Descriptions

Valentijn Borghuis[1,2*], **Luca Angioloni**[3], **Lorenzo Brusci**[2] and **Paolo Frasconi**[3]
[1]Eindhoven University of Technology
[2]MUSI-CO
[3]DINFO, University of Florence
{valentijn.borghuis, lorenzo.brusci}@musi-co.com, {luca.angioloni, paolo.frasconi}@unifi.it

Abstract

We present a pattern-based MIDI music generation system with a generation strategy based on Wasserstein autoencoders and a novel variant of pianoroll descriptions of patterns which employs separate channels for note velocities and note durations and can be fed into classic DCGAN-style convolutional architectures. We trained the system on two new datasets composed by musicians in our team with music generation in mind. Moving smoothly in the latent space allows us to generate meaningful sequences of four-bars patterns.

1 Introduction

During the last decade, the availability of new and powerful methods based on deep learning has sparkled a flourishing line of research in which generative models trained on data have been applied to the generation of music in various modalities, styles, and genres [Briot *et al.*, 2019]. In spite of significant recent advancements in this area, results are still not always usable in realistic music scenarios such as studio production using standard Digital Audio Workstations (DAWs) and live performance of electronic music. A detailed review is impossible here due to space limitations, but at a high level we can identify two main streams of research for MIDI music generation. The first one describes MIDI partitures as temporal sequences of events, suitable to be handled by generative models based on various types of recurrent networks [Boulanger-Lewandowski *et al.*, 2012; Huang *et al.*, 2019; Roberts *et al.*, 2018]. Generation of multi-instrument polyphonic music in this setting is however difficult. The second one, used in this work, focuses on patterns that consist of a fixed amount of bars (e.g., two or four), and that are described as pianorolls (PR) [Yang *et al.*, 2017; Dong *et al.*, 2018; Dong and Yang, 2018].

We introduce some important variants compared to previous works, whose combination allows us to generate meaningful and professionally usable streams of music. These include: (1) the use of a novel PR-like pattern description, what we call PR^{C}[1], (2) the use of Wasserstein autoencoders (WAE) [Tolstikhin *et al.*, 2018] as generative learners; and (3) the definition of an optimization strategy for exploring the autoencoder latent space during generation.

2 Materials and Methods

2.1 Datasets

We assembled two datasets of 910 and 968 patterns respectively, especially composed by two professional musicians for this study. In both cases we instructed composers to create coherent 120bpm patterns of four bars that were quantized at 1/32 resolution and limiting the extension to four octaves, yielding tensors of size $T = 128 \times N = 48 \times 2$ for each track. Dataset A contains patterns in the genres of *acid jazz*, *soul* and *funk* with four instruments (drums, bass, Rhodes piano, and Hammond organ). Dataset B contains patterns in the *high-pop* and *progressive trance* genres and has ten instruments (guitar, two pads, strings, choir, and lead, in addition to the above four). All patterns were transposed to prevent tonality variations.

2.2 MIDI Patterns in PR^{C}

Each track in a MIDI stream consists of a sequence of time stamped events. We consider here only two types of note events: $\mathrm{ON}(t, n, v)$, and $\mathrm{OFF}(t, n)$ where t denotes the time at which note n begins or ends and the note identifier, n, takes values in general from 0 (C-1) to 127 (G9) but only 4 octaves were used in this work. The value, v, is the MIDI velocity in the integer range $[0, 127]$. Roughly, velocity is associated with the note intensity (allowing to represent dynamic expression elements, such as *pianissimo*, *forte*, or accents in percussive instruments) but depending on the synth attached to the track, it can also affect timbre. After time quantization is applied, a pattern consisting of a fixed number of bars can be described as a tensor, a strategy previously proposed in [Dong *et al.*, 2018] but in our PR^{C}description we employ a $T \times N \times 2$ tensor $\mathbf{x}$, where T is the number of legal time positions, N the number of notes. In the first channel, $x_{q(t),n,1} = v$ if there occurs a note-on event $\mathrm{ON}(t, n, v)$ and zero otherwise. In the second channel, $x_{q(t),n,2} = d$ if there occurs a note-on event $\mathrm{ON}(t, n, v)$ and zero otherwise, but in this case d is the duration (expressed in quantized steps) of the note. Note that

*Contact Author

[1]The C after Conlon Nancarrow (1912–1997), a pioneer of piano roll compositions. Here Conlon also stands for Channeled Onset of Notes and Length Of Notes.

polyphony is handled naturally in this description and multi-instrument patterns with K tracks can be easily described by allocating two channels for each track as above, resulting in a $T \times N \times 2K$ tensor. Unlike the PR used in [Dong *et al.*, 2018], our PR$^{\text{C}}$description does not suffer the ambiguity between long notes and repeated occurrences of the same note and is completely lossless (a quantized MIDI pattern transformed into the corresponding PR$^{\text{C}}$tensor can be recovered exactly). Additionally, the PR in [Dong *et al.*, 2018] consists of Boolean tensors (after applying a hard threshold to velocity), which discard dynamic expressiveness.

2.3 Wasserstein Autoencoders

Generative models that have been applied to music include variational autoencoders (VAE) [Kingma and Welling, 2014] and generative adversarial networks (GAN) [Goodfellow *et al.*, 2014]. In both cases, a network (called either decoder or generator) is trained to map a latent or noise vector $\mathbf{z} \in \mathbb{R}^D$ into a pattern. Here we are interested in autoencoder based approaches (see also below for a motivation). The regularizer in VAEs is well known to originate a blurriness phenomenon in the case of images [Dosovitskiy and Brox, 2016]. In the case of music, we observed that "blurriness" also occurs, resulting in large clusters of notes being played together and in sometimes obsessive repetitions of short notes. WAEs [Tolstikhin *et al.*, 2018] avoid this problem by penalizing a measure of discrepancy between the *expected* $p(\mathbf{z}|\mathbf{x})$ and the prior $p(\mathbf{z})$, i.e., by pushing the expectation inside the distance. We employed the maximum mean discrepancy measure and a Gaussian prior. Encoder and decoder were structured as in the DCGAN [Radford *et al.*, 2016] architecture with large filter sizes (8×8) and 256 filters in the middle layers. A very small latent vector size ($D = 3$) was sufficient to ensure good reconstruction of training patterns. We used the square loss (with tanh output units) to measure the reconstruction error.

2.4 Generation of Pseudo-Songs

We define two modalities for producing pseudo-songs: interpolation and swirls. In the first case (only applicable to autoencoders), users have the choice of picking a start pattern $\mathbf{x}_s$ and a goal pattern $\mathbf{x}_g$ (both from the test set) and the system creates a smooth interpolation between the two. In particular, embeddings $\mathbf{z}_1 = E(\mathbf{x}_s)$ and $\mathbf{z}_T = E(\mathbf{x}_g)$ are first created using the encoder E, and then a spherical [White, 2016] trajectory in the embedding space is computed as $\mathbf{z}_t = (\sin(\frac{1-t}{\theta(1-T)})\mathbf{z}_1 + \sin(\theta\frac{t-T}{1-T})\mathbf{z}_T)/\sin(\theta)$ where $\theta = \arccos(\mathbf{z}_1/\|\mathbf{z}_1\|, \mathbf{z}_T/\|\mathbf{z}_T\|)$. In the case of swirls (also applicable to GANs), latent trajectories are produced deterministically by taking real and imaginary parts of periodic complex-valued parametric functions of the form

$$f(t; a_l, b_l, c_l, d_l) = e^{ja_l t} - e^{jb_l t}/2 + je^{jc_l t}/3 + e^{jd_l t}/4$$

i.e., using $z_{2l,t} = \Re(f(t; a_l, b_l, c_l, d_l))$ and $z_{2l+1,t} = \Im(f(t; a_l, b_l, c_l, d_l))$, for $l = 1, \ldots, D/2$.

Note that equally spaced points in the embedding space do not necessarily correspond to equally spaced reconstructions in the pattern space. In some cases, this may lead to abrupt transitions, and in some other cases to "flat" progressions that might be musically uninteresting. To address this issue we subsampled the trajectory by maximizing the minimum Euclidean distance between consecutive reconstructions, a problem that has a standard solution in terms of the bottleneck path problem [Kaibel and Peinhardt, 2006] in a Viterbi-like $T \times L$ trellis (arcs weighted by distance), being $L < T$ the desired duration of the pseudo-song. The problem is solvable by a simple modification of the Dijkstra algorithm.

3 Results and Discussion

Generated pseudo-songs with PR$^{\text{C}}$are consistent over time in terms of harmony, rhythmic and melodic flow and style, with accurate interplay between the instruments. Discontinuities and choppy results often occur when using the PR description. Results with PR$^{\text{C}}$are consistently better than those obtained with PR also quantitatively, as measured by considering test set reconstructions. In Table 1 we report precision and recall (considering as predictions notes in the reconstruction whose predicted velocity is above the minimum velocity in the training data) and, in the true positives set, the mean absolute error for the velocity (in the range $[0 - 127]$) and for the duration (in units of 1/32ths of bar). For PR the predicted note-on event was the first element in the merged row of consecutive predictions.

To allow musicians to use pseudo-song generation in production and performance settings, an API for accessing trained networks over the internet was developed. It takes a specification in the form of the start and goal pattern and the required length, L, and returns a MIDI file for the interpolation. Specification by the user and delivery is arranged through a purpose-built web client or a plug-in for DAWs. LiveAI RP (https://www.musi-co.com/listen/live/research) is a web client that lets the user generate interpolations from WAEs trained on datasets A and B. The resulting MIDI files can be downloaded for production use in any DAW.

Earlier experiments have shown that musicians appreciate interpolation as an understandable and useful form of music generation [Borghuis *et al.*, 2018]. Interpolations can sound like directed musical flows, as the music "leaves the start pattern behind" and the "attraction" exerted by the goal manifests itself by elements of the goal pattern entering the music. However, the coherence of pseudo-songs is entirely due to the properties of the embedding space: points in close proximity have reconstructions in pattern space that are musically close. Thus, a natural direction for future development is to incorporate mechanisms of conditioning by labeling of dataset patterns with structural categories (e.g. intro, verse, chorus).

	Dataset A				Dataset B			
	P	R	V	D	P	R	V	D
PR	5.5	42.2	31.9	1.34	4.1	57.9	29.8	3.11
PR$^{\text{C}}$	32.8	53.8	24.2	0.98	36.9	58.0	23.1	1.83

Table 1: Test set precision (P), recall (R), mean absolute errors on velocity (V) and duration (D) for PR and PR$^{\text{C}}$.

References

[Borghuis *et al.*, 2018] Tijn Borghuis, Alessandro Tibo, Simone Conforti, Luca Canciello, Lorenzo Brusci, and Paolo Frasconi. Off the Beaten Track: Using Deep Learning to Interpolate Between Music Genres. *arXiv:1804.09808*, April 2018.

[Boulanger-Lewandowski *et al.*, 2012] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012.

[Briot *et al.*, 2019] Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet. *Deep Learning Techniques for Music Generation, Computational Synthesis and Creative Systems*. Springer Nature, 2019.

[Dong and Yang, 2018] Hao-Wen Dong and Yi-Hsuan Yang. Convolutional generative adversarial networks with binary neurons for polyphonic music generation. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018*, pages 190–196, 2018.

[Dong *et al.*, 2018] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 34–41, 2018.

[Dosovitskiy and Brox, 2016] Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks. In *Advances in Neural Information Processing Systems 29*, pages 658–666, 2016.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680. 2014.

[Huang *et al.*, 2019] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer: Generating music with long-term structure. In *7th International Conference on Learning Representations*, 2019.

[Kaibel and Peinhardt, 2006] V. Kaibel and M. Peinhardt. On the bottleneck shortest path problem. Technical report, Otto-von-Guericke-Univ. Magdeburg, Magdeburg, Germany, 2006.

[Kingma and Welling, 2014] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations*, 2014.

[Radford *et al.*, 2016] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *4th International Conference on Learning Representations*, 2016.

[Roberts *et al.*, 2018] Adam Roberts, Jesse H. Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80, pages 4361–4370. PMLR, 2018.

[Tolstikhin *et al.*, 2018] Ilya O. Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein auto-encoders. In *6th International Conference on Learning Representations*, 2018.

[White, 2016] Tom White. Sampling generative networks: Notes on a few effective techniques. *CoRR*, abs/1609.04468, 2016.

[Yang *et al.*, 2017] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. Midinet: A convolutional generative adversarial network for symbolic-domain music generation. In *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, pages 324–331, 2017.