CONLON: A PSEUDO-SONG GENERATOR BASED ON A NEW PIANOROLL, WASSERSTEIN AUTOENCODERS, AND OPTIMAL INTERPOLATIONS

Luca Angioloni[1] **Tijn Borghuis**[2,3] **Lorenzo Brusci**[3] **Paolo Frasconi**[1]
[1] DINFO, Università di Firenze, Italy [2] Eindhoven University of Technology, The Netherlands
[3] Musi-co, Eindhoven, The Netherlands
first.last@unifi.it, first.last@musi-co.com

ABSTRACT

We introduce CONLON, a pattern-based MIDI generation method that employs a new lossless pianoroll-like data description in which velocities and durations are stored in separate channels. CONLON uses Wasserstein autoencoders as the underlying generative model. Its generation strategy is similar to interpolation, where MIDI pseudo-songs are obtained by concatenating patterns decoded from smooth trajectories in the embedding space, but aims to produce a smooth result in the pattern space by computing optimal trajectories as the solution of a widest-path problem. A set of surveys enrolling 69 professional musicians shows that our system, when trained on datasets of carefully selected and coherent patterns, is able to produce pseudo-songs that are musically consistent and potentially useful for professional musicians. Additional materials can be found at https://paolo-f.github.io/CONLON/.

1. INTRODUCTION

Algorithmic music generation has attracted the interest of musicians and practictioners for long time, starting from early works by Guttman, Hiller and Isaacson [1] and Xenakis [2] in the 1950's. Significant progress has recently resulted from the widespread application of new and powerful methods based on deep generative models, letting this class of data-driven approaches gradually take over more traditional rule-based or probabilistic techniques [3, 4]. This thriving line of research spans several dimensions of the generation process, including different types of data: audio signal [5, 6] vs. symbolic MIDI data; musical textures: monophonic [7] vs. polyphonic [8, 9]; ensembles: single-instrument [9] vs. multi-instrument [7]; goals: e.g., continuation [10], accompaniment [11], style transfer [12–14], or interpolation [7, 14].

We are particularly interested in developing a tool for a context where the automatic generation and proliferation of new music material (not necessarily finished pieces) is useful to assist musicians in music and media production. Usefulness in this context may have different facets: supporting and accelerating personal explorations of unknown or semi-known music areas, collecting intra-genre or cross-genre ideas of various complexity and abstraction, augmenting the composer's ability to explore the combinatorial space of rhythmic, melodic, and harmonic variations. In other words, what Boden calls "exploring conceptual spaces" [15]. In spite of the impressive amount of recent advancements in music generation with machine learning approaches, the musical quality of the results is still not always sufficient to enable a widespread adoption in realistic professional scenarios such as studio production using standard Digital Audio Workstations (DAWs) or live performance of electronic music.

In this paper, we focus on the autonomous generation of polyphonic and multi-instrument MIDI partitures, aiming at producing relatively long *pseudo-songs* (i.e. tracks that have the duration of a song but whose temporal structure is not controlled by a compositional intent) in mainstream genres such as Acid Jazz, Soul or High Pop, that are effectively *usable* in a professional music production context. We argue that achieving this goal requires not only the effective exploitation of algorithmic ideas but also a careful selection of coherent musical materials to be used as training data. Unlike the case of image data, where there exist large scale high quality coherent datasets (for example CelebA [16] focusing on human faces), existing symbolic music datasets for mainstream music contain large variations in genre, style, and track/instrument role, that make it more difficult to learn to generate musically coherent pseudo-songs. In the attempt to verify the impact of dataset quality on the results, we introduce in this paper (perhaps for the first time in this research area) two new datasets that were not extracted from existing collections but that have been especially composed and edited by two musicians made aware of creating training sets for generative models. One dataset, ASF-4, is in Acid Jazz, Soul and Funk; the other one, HP-10, in High Pop. Compared to datasets used in other experiments (e.g. LPD-5 [17]) they are small, i.e. of a size that individual musicians would be able to compose or curate by themselves. This opens up the possibility of personalized generators, assisting musicians in producing their own music.

From the algorithmic point of view, possible solutions to the music generation problem can be characterized across several separate dimensions (see [3] for a thorough taxonomy). The most important ones for the goals of this paper are (1) the type of data structures that are used to describe MIDI patterns (2) the nature of the generative learning models, and (3) the strategy used to produce a whole musical piece. The system presented in this paper introduces novelties across all these three dimensions, whose combination allows us to generate meaningful and professionally usable streams of music. In terms of data description, we introduce a novel pianoroll-like pattern description that stores velocities and durations in two separate channels. The description is lossless (i.e., it can be inverted to recover the original MIDI data exactly) and perceptually robust to reconstruction errors (i.e., it does not suffer the note shattering problem associated with binary pianorolls that store consecutive high bits to represent note durations). As a generative model, we experiment with Wasserstein autoencoders (WAE) [18], a type of autoencoder that is less subject to the "blurriness" problem typically associated with variational autoencoders (VAE) [19]. To the best of our knowledge, WAEs have not been applied to music generation before. Third, our generation strategy is based on interpolation as in previous works [7,14] but we formulate it as an optimization problem for exploring the autoencoder latent space in a way that prevents abrupt transitions between consecutively generated patterns, as well as regions with little variation.

We call our system CONLON, for Channeled Onset of Notes and Length Of Notes, and in honor of Conlon Nancarrow (1912–1997), a pioneer of piano roll compositions. A thorough evaluation with human experts suggest that CONLON is able to produce pseudo-songs that are truly exploitable by professional musicians in mainstream genres.

2. A NEW PIANOROLL WITH EXPLICIT DURATIONS

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 $\mathrm{OFF}(t, n)$. Here t denotes the time at which a note with pitch n begins or ends, measured in MIDI clock pulses. In general, n takes values between 0 (C-1) and 127 (G9). The MIDI velocity, v, takes values 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 instrument attached to the track, it can also affect timbre (for example, a "clicked" Hammond organ sound could be selected in the sound bank when the velocity exceed a given threshold). Note that although n and v take values in the same range, v should be regarded as a continuous variable and n as a categorical one.

Before this data can be fed into a learning algorithm, it needs to be arranged in a proper description format. Both variable-lenght and fixed-lenght descriptions have been studied in the literature. Variable-length descriptions are typically used in conjunction with various types of recurrent neural networks (RNNs) [20]. Fixed-length description include the pianorolls (PR) introduced in [21] for melodies and later extended to the multitrack polyphonic case [17, 22], and are suitable for modules based on convolutional neural networks (CNNs) [23]. Pianorolls, however, are a *lossy* description of MIDI data in at least two ways. First, they do not include note velocities, which are important for dynamic expression in many musical genres. Second, they make it impossible to distinguish between long notes and repeated occurrences of the same notes (see Fig. 1). The latter limitation can be mitigated by using a finer quantization step or fixed by adding a replay matrix [9]. Still, the PR description may suffer a fundamental problem when there are imperfections in the reconstructions generated by a trained model. In facts, false negatives in the reconstruction may shatter a long note into several shorter ones, which may produce a musically obsessive and unpleasant result.

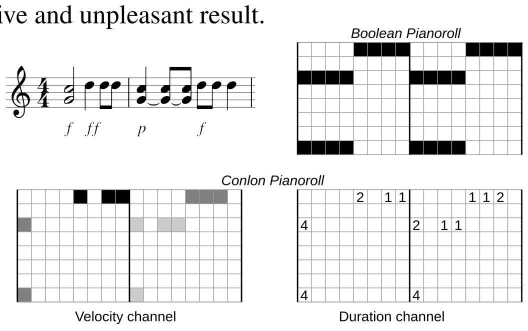

Figure 1. A short phrase described as PR (top right) and as PR^{C} (bottom). To construct a simple example, here we set quantization at 1/8.

The solution proposed in this paper uses a second channel as in [9] but explictly represents note durations as continuous variables. More precisely, in our PR^{C} description, the tensor associated with a fixed-length music pattern is constructed as follows. First, we introduce a time quantization function q that maps fine-grained timestamps into coarse-grained temporal positions in the range $1, \ldots, T$. For example $T = 128$ if we quantize four $\frac{4}{4}$ bars at 1/32th. Assuming for simplicity a single instrument and denoting by N the number of pitches in the used range, we create a tensor x of shape $T \times N \times 2$. In the first channel, $x_{q(t),n,1} = v$ if there occurs an event $\mathrm{ON}(t, n, v)$ and $x_{q(t),n,1} = 0$ otherwise. In the second channel, $x_{q(t),n,2} = d$ if there occurs an event $\mathrm{ON}(t, n, v)$ whose duration (expressed in quantized steps) is d, and $x_{q(t),n,2} = 0$ otherwise. The construction is illustrated in Fig. 1. 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. Our PR^{C} description does not suffer the ambiguity between long notes and repeated occurrences of the same note and, except for time quantization, is completely lossless (i.e., a quantized MIDI pattern transformed into the corresponding PR^{C} tensor can be recovered exactly). Additionally, it can be perceptually more robust to reconstruction errors. A further advantage is that all the information about a note is local, whereas in the case of PR, a convolutional network requires a wide receptive field to infer the note duration.

3. GENERATING PATTERNS WITH WASSERSTEIN AUTOENCODERS

Generative models that have been applied to music include various types of encoder/decoder architectures, different variants of generative adversarial networks (GAN) [24], as well as transformer based architectures [25]. Variational autoencoders (VAE) [19] have been used in systems like VRAE [26], GLSR-VAE [27] and MusicVAE [7] while GANs have been used in systems like C-RNN-GAN [28] MidiNet [21] and MuseGAN [17, 22].

Both with autoencoders and GANs, a network $G(z)$ (called either decoder or generator) is trained to map a latent or noise vector $z \in \mathbb{R}^{d_z}$ into a pattern. Here we are interested in autoencoder based approaches (see also Section 4.1 for a motivation). Among these approaches, VAEs are theoretically elegant and applicable to music generation. They are regularized by penalizing the expected KL divergence between the posterior $q(z|x)$ and a zero-mean Gaussian prior p_z, with the expectation being taken over training points. However, as nothing prevents different patterns being mapped to close latent codes, the decoder is sometimes asked to reconstruct different patterns from similar codes, resulting into a well known blurriness phenomenon in the case of images [29]. In the case of music patterns described as tensors, we observed that a form of "blurriness" also occurs, resulting in large clusters of notes being played together and sometimes in swarms of short notes that are never present in the training data. WAEs [18] avoid this problem by pushing the expectation inside the divergence, i.e., penalizing a divergence $\mathcal{D}$ between the prior q_z and the *aggregated* posterior $q_z(z) = \mathbb{E}_p\, q(z|x)$, where p is the data distribution. WAEs thus minimize, with respect to the parameters of the decoder, the quantity

$$\min_{q(z|x)} \mathbb{E}_p\, \mathbb{E}_{q(z|x)}\, c(x, G(z)) + \lambda \mathcal{D}(q_z, p_z) \qquad (1)$$

where c is a reconstruction loss and λ a hyperparameter to be fixed. In all our experiments we employed the Maximum Mean Discrepancy (MMD) [30] for $\mathcal{D}$ and a Gaussian prior for p_q, and we structured the encoder and the decoder as in the DCGAN [31] architecture. Note that unlike MuseGAN, which stacks bars over an additional tensor axis and uses 3D convolutional layers, tensors in PR$^{\text{C}}$ can be processed by 2D convolutional layers. The input tensors are normalized in the $(-1, 1)$ range. The final layer of all our decoders has hyperbolic tangent output units. The mean squared error between reconstructions and input patterns was used in the optimization criterion during training.

The latent vector size, d_z, was adjusted with a trial-and-error approach, trying to find the smallest possible value yielding good quality interpolations. If d_z is too small the validation error is large but if d_z is too large, interpolations tend to create many patterns that are too close to those in the training set, in spite of a small validation error (which is therefore not an useful metric). Additionally, nearly empty patterns tend to appear in the middle of interpolations. We found $d_z = 3$ for ASF-4 and $d_z = 5$ for HP-10 and LPD-5 to be a reasonable compromise. The remaining hyperparameters were tuned using random search [32] guided by the validation set reconstruction error. We focused in particular on the learning rate, η, and the number of epochs T used in conjunction with the Adam algorithm; the number of layers n_l; the number of filters n_f and the size of filters, k, used in the the DCGAN encoder and decoder (strides were fixed to 2). Interestingly, large filter sizes $k = 8$ were found to perform better than standard smaller sizes as compact filters are not able to capture musical patterns and distant relationships between notes.

3.1 Converting Generated Tensors to MIDI

In the case of PR$^{\text{C}}$ descriptions, "decoding" a MIDI pattern from the output tensor is almost straightforward (unlike the case of PR, where specialized GAN architectures have been introduced to avoid post-processing based on either hard thresholding or Bernoulli sampling [22]). First, predicted velocities and durations for each time t, pitch n, and instrument i, are rescaled in the range 0–127. Events whose rescaled predicted velocity $v(t, n, i) < 1$ or duration $d(t, n, i) < 1$ (i.e., non-audible notes) are discarded. Finally, we apply the following transformation (similar to a gamma correction) to obtain corrected velocities $v_c(t, n, i)$ as follows:

$$v_c(t, n, i) = \left\lfloor 127 \left(\frac{v(t, n, i)}{127} \right)^{\frac{1}{\gamma_i}} \right\rfloor \qquad (2)$$

where γ_i is an instrument specific correction factor ranging from 2.9 for drums to 4.0 for Rhodes. Durations are directly retrieved from the (rescaled) duration channel.

4. PSEUDO-SONGS

The following approach assumes that a generative model $G : z \in \mathbb{R}^{d_z} \mapsto x \in \mathbb{R}^{m \times q \times 2}$ from embeddings to patterns is available. Function G can be either the decoder of an autoencoder or the generator of a GAN. A pseudo-song is then generated by creating a trajectory of length T, $z_1, \ldots, z_T$, and applying G to each latent vector to produce a corresponding sequence of patterns.

4.1 Interpolations

When using autoencoders, we have the choice of picking a start pattern x_s and a goal pattern x_g (both from the test set) and use the encoder E to obtain $z_1 = E(x_s)$, $z_T = E(x_g)$. This for example allows users to produce a pseudo-song that moves smoothly from one genre to another. Musicians could even create ex-novo start and goal patterns with a particular purpose in mind. When using GANs to generate, this option is not available but z_1 and z_T can be sampled from $p(z)$ (with a less intentional result) or, alternatively, users may be given a set of pre-generated patterns from which to pick the endpoints from the pre-images of the generator.

Two options are common for creating trajectories, linear interpolation: $z_t = \frac{t-T}{1-T} z_1 + \frac{1-t}{1-T} z_T$, $t = 2, \ldots, T-1$, and spherical interpolation:

$$z_t = \frac{\sin\left(\frac{1-t}{\theta(1-T)}\right) z_1 + \sin\left(\theta \frac{t-T}{1-T}\right) z_T}{\sin(\theta)} \tag{3}$$

where $\theta = \arccos(z_1/\|z_1\|, z_T/\|z_T\|)$. The latter is preferable when $p(z)$ is Gaussian and d_z is large [33].

4.2 Swirls

In this approach (also applicable to GANs), latent trajectories are produced by taking real and imaginary parts of periodic complex-valued parametric functions of the form

$$f(t; a_l, b_l, c_l, d_l) = e^{j a_l t} - e^{j b_l t}/2 + j e^{j c_l t}/3 + e^{j d_l t}/4$$

for random choices of (a_l, b_l, c_l, d_l), 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 $l = 1, \ldots, \lfloor d_z/2 \rfloor$, $t = 1, \ldots, T$.

4.3 Trajectory Smoothing

Equally spaced points in the embedding space do not necessarily correspond to equally spaced reconstructions in the pattern space. This essentially depends on how generative models allocate points in the pattern space to points in the embedding space. When creating pseudo-songs with either interpolations or swirls, this fact may lead in some cases to abrupt transitions and in some other cases to repetitive regions that might be musically uninteresting. To address this issue, we suggest to increase T beyond the desired length and then subsample the trajectory. Smoothness can be achieved by maximizing the minimum distance between consecutive reconstructions and constraining the final length to a desired integer L:

$$\max_{t_1, \ldots, t_L} \min_{i=1,\ldots,L-1} \delta(G(z_{t_i}), G(z_{t_{i+1}})) \tag{4}$$

$$\text{s.t.} \quad 1 \leq t_i < t_{i+1} \leq T \quad i = 1, \ldots L-1 \tag{5}$$

$$t_{i+1} - t_i \leq H \quad i = 1, \ldots L-1 \tag{6}$$

where δ is a distance function on patterns (e.g., the Euclidean distance on PR^{C} tensors) and H a lookahead horizon, i.e., the maximum allowed number of positions that may be skipped. Problem (4) can be reduced to an instance of the bottleck shortest path problem [34] on the $T \times L$ trellis with vertex set $\{(t, \tau), t = 1, \ldots, T; \tau = 1, \ldots, L\}$, edge set $\{((t, \tau), (t+1, \tau+s)), t = 1, \ldots, T; \tau = 1, \ldots, L, s = 1, \ldots, H\}$, and edge weights $w((t, \tau), (t+1, \tau+s)) = \delta(G(z_t), G(z_{t+s}))$ (see Fig. 2). The problem is solvable by a slightly modified version of the Dijkstra algorithm (where nodes in the frontier are labeled by their maximum step cost rather than the sum of the step costs) or by a faster algorithm based on bucketing [34].

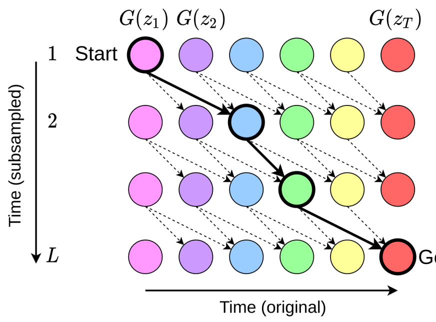

Figure 2. Trellis for trajectory smoothing. The horizon H is 2 in this example. Among all paths from Start to Goal, the highlighted path is the one whose smallest edge weight is maximum.

5. DATASETS

We tested CONLON on three dataset. ASF-4 is a set of 910 patterns of four bars in three genres: *acid jazz*, *soul* and *funk*. Each pattern has $K = 4$ tracks associated with a simple electro-acoustic quartet: drums, bass, Rhodes piano, and Hammond organ. HP-10 is a set of 968 patterns of four bars in two genres: *high-pop* and *progressive trance*. Each pattern has $K = 10$ tracks associated with the following instrument set: drums, bass, Rhodes, brass-synth, choir, dark-pad, guitar, lead, pad, and strings. Both ASF-4 and HP-10 have been especially composed by two professional musicians for this study. In both cases, composers were instructed to create coherent 120bpm patterns of four bars. All patterns in these datasets were subsequently quantized at 1/32th resolution, manually curated for mistakes (including fixing errors due to quantization), and finally transposed to either Cmaj or Amin to prevent tonality variations. The resulting PR^{C} tensors have size $128 \times N \times 2K$, where $N = 55$ for ASF-4 and $N = 60$ for HP-10. LPD-5 (cleansed version) was derived from the Lakh MIDI dataset [35] by Dong *et al.* [17] by retaining only songs with high matching scores to the Million Song Dataset. It contains 21,425 multitrack MIDI songs with $K = 5$ tracks and $N = 108$ pitches, where original tracks/instruments were merged into instrument families and cut by the authors of [17] into patterns consisting of two 4/4 bars. Automatic quantization at 1/48 was applied yielding tensors of size $192 \times 108 \times 10$.

Being manually curated, ASF-4 and HP-10 are much more *coherent* than LPD-5 in at least three ways. First, music style and genre is highly constrained, whereas LPD-5 contains a wide assortment of different genres. Second, instrument-role is well defined, i.e., the instrument that plays in a certain track always maintains its role across the whole dataset. In LPD-5, some heuristics have been applied by the authors of [17] to map instruments to the five tracks but in some cases very different instruments may collide on the same track. Third, the endpoints that demarcate patterns always cover homogeneous phrases, i.e., the phrase always begins on the first bar. In LPD-5, patterns are extracted by an automatic segmentation technique that cannot be equally reliable.

6. LISTENING EXPERIMENTS

To validate the CONLON approach, we conducted three listening experiments with a group of 69 musicians, recruited by the authors with the help of colleagues teaching in several music educational institutes. Subjects mainly identified themselves as composers and producers (often in combination with instrumentalist/performer), working mostly in three (non mutually exclusive) genres: Classical, Contemporary and Electronic Dance Music. Over half the participants had more than 10 years of professional experience in music, only one less than 3 years.

Participants were told that the survey was part of a research project on the generation of music with machine learning techniques, with the long-term aim of developing new tools for music production and performance. They participated through an online survey service [1] that allows for questions with audio materials. The experiments consisted of two types of tasks.

Comparison Subjects listen to a pair of short music tracks (64 bars, 130s), and indicate which of the two tracks is most usable in the context of mainstream music production (forced choice). The scenario they are asked to keep in mind is that they would receive the tracks as a midi file for inclusion and editing in the production of a mainstream song in their own DAW.

Analysis Subjects listen to a single track, an excerpt (64 bars) from a longer piece and assess the musical development of the composition over time, with regards to four aspects: Harmony, Rhythm, Melody, and Interplay of instruments, each judged on a 5-point Likert scale (from “very incoherent" to “very coherent"). To obtain further feedback, we asked the subjects to comment on good points of the composition and points for improvement.

Answers in the comparison task were converted to ranks for the tracks in a pair (i.e. rank is 1 for the preferred track and 2 for the other). We then computed the mean rank of the tracks across subjects. Following [36], we employed Kendall's Coefficient of Concordance (W) [37], to analyse the level of agreement among subjects, along with a commonly used significance test against the null hypothesis of no agreement [38].

In the following we describe three experiments aiming at testing specific hypotheses relating CONLON to MuseGAN, PR to PR^C, and the usability of pseudo-songs. For all models we set the interpolation length $T = 64$, the desired length $L = 16$, and the widest-path horizon $H = 20$.

[1] `http://www.surveygizmo.com`

Method	HP-10	LPD-5
CONLON	1.17	1.45
MuseGAN	1.83	1.55
Concordance	0.64	0.01
Significance	$p < 0.0005$	ns

Table 1. Mean ranks assigned by subjects to the usability of interpolations generated with our system (CONLON) and MuseGAN [22]. $m = 75$ pairs were ranked.

Description	ASF-4	HP-10	LPD-5
PR^C	1.08	1.31	1.5
PR	1.92	1.69	1.5
Concordance	0.72	0.15	0
Significance	$p < 0.0005$	$p < 0.001$	ns

Table 2. Mean ranks assigned by subjects to the usability of pseudo-songs generated with PR^C and PR descriptions. $m = 78$ pairs were ranked.

	Aspect			
	Harmony	Rhythm	Melody	Interplay
Coherent	49%	67%	42%	51%
Neutral	35%	20%	29%	20%
Incoherent	16%	13%	29%	29%
Significance	p<.005	p<.0005	p<.005	p<.0005

Table 3. Coherence of CONLON pseudo-songs as judged by subjects, with respect to harmony, rhythm, melody, interplay of instruments. $m = 69$ judgements were collected.

6.1 Comparing CONLON and MuseGAN

To substantiate that CONLON is more usable in music production than previous approaches, we tested *Hypothesis 1: Musicians find pseudo-songs generated with WAEs and PR^C descriptions more useable in music production than pseudo-songs generated with the MuseGAN model and PR (other factors being equal).* A WAE-model was trained on PR^C representations of the datasets HP-10 and LPD-5, and a MuseGAN model on PR representations of datasets HP-10 and LPD-5 to generate interpolations. On the same data, we trained a MuseGAN model with binary neurons [22] using the implementation at `https://github.com/salu133445/musegan`. Subjects were given pairs of matching interpolations to compare, differing in the approach used, but with the same start, goal and length. Six pairs (three for each dataset) were presented for comparison to 25 subjects, producing a total of 75 observations per dataset.

Subjects generally judged pseudo-songs generated by CONLON to be more usable than pseudo-songs generated with the MuseGAN approach (for 5 out of the 6 pairs). But whereas the difference in ranking is clear and concordance among participants is significant for the three pairs on the HP-10 dataset, differences were small and concordance not significant among subjects for pseudo-songs generated from LPD-5. Table 1 shows the aggregated mean ranking per dataset.

6.2 Comparing PR and PR^C

To investigate whether part of the improvement over previous approaches is due to the representation, we tested *Hypothesis 2: Musicians find pseudo-songs generated with PR^C descriptions more useable in music production than pseudo-songs generated with PR descriptions (other factors being equal).* A WAE-model was trained on PR

	ASF-4				HP-10				LPD-5			
	P	R	V	D	P	R	V	D	P	R	V	D
PR	6.1	51.1	32.2	2.5	4.1	58.9	28.8	3.1	1.4	89.7	41.0	5.5
PR^C	32.1	53.9	24.3	1.0	37.0	58.0	23.4	1.8	35.5	60.2	19.5	2.6

Table 4. Test set precision (P), recall (R), mean absolute errors on velocity (V) and duration (D) for PR and PR^C .

and PR^C representations of the datasets ASF-4, HP-10 and LPD-5 to generate interpolations. 26 subjects were given nine pairs of matching interpolations to compare (differing only in representation used), three for each dataset, resulting in 78 observations per dataset.

Subjects generally judged pseudo-songs generated with PR^C representations to be more usable than pseudo-songs generated with PR representations (for 8 out of the 9 pairs). The difference in ranking is clear and concordance among participants is significant for the 6 pairs on the ASF-4 and HP-10 datasets, but differences were small and concordance not significant for the 3 pairs generated on different representations of the LPD-5 dataset. Table 2 shows the aggregated mean ranking and cross-subject concordance per dataset.

6.3 Analyzing Development over Time

To validate the way patterns are chained together by CONLON, we tested *Hypothesis 3: Musicians find the development over time of pseudo-songs generated with WAEs and PR^C description coherent rather than incoherent (in terms of harmony, rhythm, melody and interplay between instruments).* A WAE-model was trained on a PR^C representation of datasets ASF-4 and HP-10 to generate swirls. All subjects were given the analysis task for a swirl, resulting in 69 observations per aspect.

Subjects generally judged the coherence of pseudo-songs on the positive side of the scale. For the three swirls presented in the experiment, each with four aspects, the median for all aspects lies at "somewhat coherent" (8 out of 12) or "neutral" (4 out of 12). The rating "very coherent" is reached for all aspects, the rating "very incoherent" in 10 out of 12. For all swirls, rhythm is the aspect with the highest coherence rating. Table 3 shows the aggregated answers for the three swirls, recoded to a 3-point scale.

7. QUANTITATIVE EVALUATIONS

When only a limited amount of human expert time is available for surveys, it becomes difficult to cover all different dimensions on which alternative methods can be compared. Rather that allowing non experts in our surveys, it may be preferable to complement human evaluation with a number of automatically computed metrics [39]. Here we consider reconstruction error and note shattering.

7.1 Reconstruction error

Here we complement human evaluations with some automatically computed metrics that are derived from test set reconstructions and are applicable to autoencoder-based methods. In particular, we aim to compare WAEs fed by PR vs WAEs fed by PR^C . Precision and recall are defined on the binary classification problem where the ground truth consists of Bernoulli variables $y(t,n,i) = 1$ if there is a note-on event at time t for note n and instrument i. For these metrics we considered as predictions the binary quantities $\widehat{y}(t,n,i) = 1$ if the reconstructed value of the velocity at position (t,n,i) is above the smallest velocity encountered in the training set. In the case of PR description, the predicted note-on event was the first element in the merged row of consecutive predictions. We further considered the mean absolute errors in predicting velocities (in the range $[0-127]$) and durations (in units of 1/32ths of bar). Test set results comparing PR and PR^C (everything else being equal) are reported in Table 4.

7.2 Note Shattering

Results in Table 4 indicate that PR yields good recall but very low precision, and has a higher error on both velocity and duration. This can be partially explained by the presence of a high number of shattered notes. To verify this hypothesis we computed the note number growth due to shattering as follows. For each note in the ground truth, identified by the triplet (n,i,T), being n the pitch, i the instrument, and $T = [t_{\mathrm{ON}}, t_{\mathrm{OFF}}]$ the temporal interval, we counted the number of notes in the reconstruction that have the same pitch n and instrument i, and whose note-ON time falls within T. We then summed these counts over all notes in the test set. In the absence of shattering, the total count equals the original number of notes. We found that WAE-PR increased the number of notes by 19%, 12%, 38% on ASF-4, HP-10, and LPD-5, respectively. By comparison, the increase factors were only 5%, 3%, and 10% in the case of WAE-PR^C .

8. CONCLUSIONS

CONLON combines the new PR^C data description with Wasserstein autoencoders and generation strategies based on optimized interpolation and swirling to produce pattern-based pseudo-songs. When trained on coherent datasets, the generated material is musically coherent and potentially useful in music production by professional musicians.

Pseudo-songs can sound like directed musical flows, but this is entirely due to the properties of the embedding space, longer-term structure is not considered. In that sense, interpolating and swirling are closer to improvisation than to composition. A natural next step is to label dataset patterns with structural categories (e.g. verse, chorus) and introduce form via mechanisms of conditioning.

9. REFERENCES

[1] L. A. Hiller and L. M. Isaacson, *Experimental Music: Composition with an Electronic Computer*. McGraw-Hill, 1959.

[2] I. Xenakis, "Musiques formelles: nouveaux principes formels de composition musicale," *La Revue musicale*, no. 253–254, 1963, paris: Editions Richard-Masse.

[3] J.-P. Briot, G. Hadjeres, and F. Pachet, *Deep Learning Techniques for Music Generation, Computational Synthesis and Creative Systems*. Springer Nature, 2019.

[4] F. Carnovalini and A. Rodà, "Computational creativity and music generation systems: An introduction to the state of the art," *Frontiers in Artificial Intelligence*, vol. 3, Apr 2020.

[5] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *CoRR*, vol. abs/1609.03499, 2016.

[6] S. Dieleman, A. van den Oord, and K. Simonyan, "The challenge of realistic music generation: modelling raw audio at scale," in *Advances in Neural Information Processing Systems 31*, 2018, pp. 8000–8010.

[7] A. Roberts, J. H. Engel, C. Raffel, C. Hawthorne, and D. Eck, "A hierarchical latent vector model for learning long-term structure in music," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80. PMLR, 2018, pp. 4361–4370.

[8] N. Boulanger-Lewandowski, Y. Bengio, and P. 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*, 2012.

[9] H. H. Mao, T. Shin, and G. W. Cottrell, "Deepj: Style-specific music generation," in *12th IEEE International Conference on Semantic Computing*, 2018, pp. 377–382.

[10] C. Donahue, H. H. Mao, Y. E. Li, G. W. Cottrell, and J. J. McAuley, "Lakhnes: Improving multi-instrumental music generation with cross-domain pre-training," in *Proceedings of the 20th International Society for Music Information Retrieval Conference*, 2019, pp. 685–692.

[11] H. Lim, S. Rhyu, and K. Lee, "Chord generation from symbolic melody using BLSTM networks," in *Proceedings of the 18th International Society for Music Information Retrieval Conference*, 2017, pp. 621–627.

[12] I. Malik and C. H. Ek, "Neural translation of musical style," *CoRR*, vol. abs/1708.03535, 2017.

[13] S. Dai, Z. Zhang, and G. Xia, "Music style transfer issues: A position paper," *CoRR*, vol. abs/1803.06841, 2018.

[14] G. Brunner, A. Konrad, Y. Wang, and R. Wattenhofer, "MIDI-VAE: modeling dynamics and instrumentation of music with applications to style transfer," in *Proceedings of the 19th International Society for Music Information Retrieval Conference*, 2018, pp. 747–754.

[15] M. A. Boden, *The Creative Mind: Myths and Mechanisms*. Routledge, 2004.

[16] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[17] H. Dong, W. Hsiao, L. Yang, and Y. 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*, 2018, pp. 34–41.

[18] I. O. Tolstikhin, O. Bousquet, S. Gelly, and B. Schölkopf, "Wasserstein auto-encoders," in *6th International Conference on Learning Representations*, 2018.

[19] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations*, 2014.

[20] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, 2017.

[21] L. Yang, S. Chou, and Y. Yang, "Midinet: A convolutional generative adversarial network for symbolic-domain music generation," in *Proceedings of the 18th International Society for Music Information Retrieval Conference*, 2017, pp. 324–331.

[22] H. Dong and Y. Yang, "Convolutional generative adversarial networks with binary neurons for polyphonic music generation," in *Proceedings of the 19th International Society for Music Information Retrieval Conference*, 2018, pp. 190–196.

[23] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[24] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2672–2680.

[25] C. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music transformer: Generating music with long-term structure," in *7th International Conference on Learning Representations*, 2019.

[26] O. Fabius, J. R. van Amersfoort, and D. P. Kingma, "Variational recurrent auto-encoders," in *3rd International Conference on Learning Representations, ICLR*, 2015.

[27] G. Hadjeres, F. Nielsen, and F. Pachet, "GLSR-VAE: geodesic latent space regularization for variational autoencoder architectures," in *IEEE Symposium Series on Computational Intelligence*, 2017, pp. 1–7.

[28] O. Mogren, "C-RNN-GAN: continuous recurrent neural networks with adversarial training," *CoRR*, vol. abs/1611.09904, 2016, constructive Machine Learning Workshop at NIPS 2016, Barcelona.

[29] A. Dosovitskiy and T. Brox, "Generating images with perceptual similarity metrics based on deep networks," in *Advances in Neural Information Processing Systems 29*, 2016, pp. 658–666.

[30] M. Binkowski, D. J. Sutherland, M. Arbel, and A. Gretton, "Demystifying MMD gans," in *6th International Conference on Learning Representations*, 2018.

[31] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *4th International Conference on Learning Representations*, 2016.

[32] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

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

[34] V. Kaibel and M. Peinhardt, "On the bottleneck shortest path problem," Otto-von-Guericke-Univ. Magdeburg, Magdeburg, Germany, Tech. Rep., 2006.

[35] C. Raffel, "Learning-based methods for comparing sequences, with applications to audio-to-Midi alignment and matching," Ph.D. dissertation, Columbia University, 2016.

[36] A. Novello, M. F. McKinney, and A. Kohlrausch, "Perceptual evaluation of music similarity," in *Proceedings of the 7th International Conference on Music Information Retrieval*, 2006, pp. 246–249.

[37] M. Kendall, *Rank Correlation Methods (5th ed.)*. London: Charles Griffin, 1975.

[38] M. Kendall and J. D. Gibbons, *Rank Correlation Methods (5th ed.)*. New York, NY: Oxford University Press, 1990.

[39] L. Yang and A. Lerch, "On the evaluation of generative models in music," *Neural Computing and Applications*, vol. 32, no. 9, pp. 4773–4784, 2020.