Curating Generative Raw Audio Music with D.O.M.E.

CJ Carr
Dadabots
Boston, USA
emperorcj@gmail.com

Zack Zukowski
Dadabots
Los Angeles, USA
thedadabot@gmail.com

ABSTRACT

With the creation of neural synthesis systems which output raw audio, it has become possible to generate dozens of hours of music. While not a perfect imitation of the original training data, the quality of neural synthesis can provide an artist with many variations of musical ideas. However, it is tedious for an artist to explore the full musical range and select interesting material when searching through the output. We needed a faster human curation tool, and we built it. DOME is the Disproportionately-Oversized Music Explorer. A PCA-component k-means-clustered rasterfairy-quantized t-SNE grid is used to navigate clusters of similar audio clips. The color mapping of spectral and chroma data assist the user by enriching the visual representation with meaningful features. Care is taken in the visualizations to aid the user in quickly developing an intuition for the similarity and range of sound in the rendered audio. This turns the time consuming task of previewing hours of audio into something which can be done at a glance.

CCS CONCEPTS

• **Human-centered computing** → **Visualization systems and tools**; • **Applied computing** → **Sound and music computing**; • **Computing methodologies** → *Machine learning*.

KEYWORDS

audio clustering, t-SNE, PCA, k-means, generative music, visualization tool.

ACM Reference Format:
CJ Carr and Zack Zukowski. 2019. Curating Generative Raw Audio Music with D.O.M.E.. In *Joint Proceedings of the ACM IUI 2019 Workshops, Los Angeles, USA, March 20, 2019*. ACM, New York, NY, USA, 4 pages.

1 MOTIVATION

With the creation of neural synthesis systems which output raw audio, it has been getting easier to generate dozens of hours of music in a specific style. [1] describe a music production process where this generated audio is curated and arranged into albums. However, much of what is generated from these networks tends to fall into similar patterns. It can be tedious to discover sections of musical interest when searching through the output. We have felt a need for faster human curation tools.

Digital audio workstations such as Ableton Live only supply the user with an amplitude visualization, which makes searching difficult. Other digital audio workstations such as Audacity have a spectrogram visualization, but no method of clustering similar audio together.

Neural synthesizers include architectures based on convolutional neural networks (WaveNet [22]), recurrent neural networks (SampleRNN [11], WaveRNN [4]), and flow-based networks (WaveGlow [15]). This family of tools is often utilized as a vocoder component in end-to-end text-to-speech models [15] and can be appropriated for music synthesis [2, 25].

2 RELATED WORK

Self-organizing maps (SOMs) have been created for the purpose of organizing large libraries of audio using clustering techniques to build grids of audio [16]. These interfaces are useful for seeing the similarity between artist or genre. Some systems have used non-technical-looking designs to encourage a feeling of exploration by stylizing the clustered datapoints into island-like [6, 14] and galaxy-like [19] environments. Our prototype adds to this grid approach by introducing visualizations of the audio which allow the user to leverage visual cues resulting from spectral, chroma, and other metadata features.

Unlike [24] which uses text or thumbnails to represent audio, and unlike [6, 14] which use a histogram of critical bands, we choose to use mel-spectrograms to maintain local detail along the time axis.

3 METHOD

3.1 Audio Preparation

Our tool works well on audio datasets up to dozens of hours in total length, where individual audio files are typically 1 minute to 20 minutes long. Larger datasets of 100+ hours have not been tested.

In one example, we start with the entire wav output of a SampleRNN experiment, as described in [25], where the training data is the album 'Time Death' by earth metal band (((::ofthesun::))) and the output is generated music in the style of the training data. At each epoch during training, and after training, wav files are generated, each 1 to 5 minutes long, for a total of 10 hours of audio.

In another example, we worked with breakcore/electronic artist Drumcorps, to train a separate net on each of the stems (guitar, voice, drums, synth) on his newest unreleased album, as well as the combined master recording. For each net, at each epoch, and after training, wav files were generated, each 1 to 5 minutes long, totaling 50 hours.

3.2 Analysis

Three types of analysis are performed: a mel-spectrogram rolloff visualization, a source-separation-pre-processed chromagram visualization, and a PCA fingerprinting used for clustering and nearest neighbor sorting. Each audio file is loaded into our analyzer, which uses scikit-learn and the librosa library [9]. STFT is done with 44.1k

IUI Workshops'19, March 20, 2019, Los Angeles, USA

sample rate, 1024 fft size, 1024 window size, and 256 hop size. The power spectrogram is made by squaring the absolute value of the STFT, discarding the phase. Frequencies are weighted according to A-weighting, a perceptual weighting scheme. Power is converted to dB.

3.3 Color

We follow the usability guidelines of memorability, informational delivery, distinguishability [24] through our use of color choices which make it easier to distinguish between different pieces of audio.

It is important to choose the color mappings with care. Rainbow palettes are prone to illusions and misrepresenting variance. However, in the case of chromagrams, because of the shared symmetry between pitch and hue, a rainbow palette is useful. Another consideration is the difficulty a color blind person might have with some palettes. Since 1 in 12 men, and a little over 4% of the whole population, are color blind [23], we have included a key command to rotate hues to accommodate incomplete color blindness.

3.4 Rolloff Spectrogram

For the rolloff visualization (Figure 1) the spectrograms are scaled to 23 mel bins. Spectral rolloff is calculated for each frame and colored accordingly. We find rolloff to be a useful measure of bassiness to trebliness. When looking at a spectrogram colored this way, at a distance, the eye clearly sees long-term structure and spots similar-sounding sections in a collection. We also include the option to vertically reflect the spectrogram along the time axis, which seems to assist the eye in seeing structure, perhaps because of the vertical symmetry, or perhaps because users are conditioned to see waveforms this way.

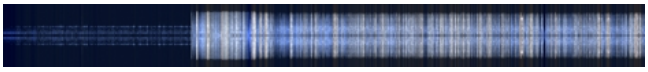

Figure 1: Rolloff visualization mode. Frequency is on the y-axis, time is on the x-axis. The spectrogram is vertically reflected, and colored reddish when bass frequencies are dominant and blue for treble. Notice a quiet section on the left, followed by a rhythmic section to the right.

Figure 2: The same wav from Figure 1 in chroma visualization mode. Pitches are on the y-axis, time is on the x-axis. Notice it starts with a drone which holds out a 5th dyad between G (cyan) and D (yellow). The drone is comparatively quiet, but normalization of the chromagram makes it stand out visually.

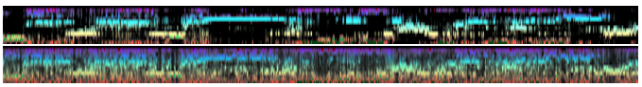

Figure 3: Chroma visualization with (above) and without (bottom) pre-processing with harmonic source separation using HPSS. Notice how removal of non-pitched audio denoises the chromagram.

3.5 Chromagram

For the chromagram visualization (Figure 2) the STFT is pre-processed with harmonic-percussive source separation (HPSS) [3]. A large margin of 10 is applied to single out the harmonic component, on which the chroma values are calculated. The percussion and residual components are discarded. The chromagram is colored according to the rainbow, such that the same pitches have the same color. To make true pitches to pop out, and to avoid the case where white noise looks colorful, a norm of 1 is used, any chroma value under 0.5 is desaturated, and values between 0.5 and 1.0 are progressively saturated.

3.6 Metadata Visualization

More advanced visualization is possible if the generative model supplements the audio with metadata.

In Figure 4 we display the current epoch and iteration of the model which generated the audio, along with parameters such as beam_width [10].

In Figure 5 we display the change in temperature [18] over time. Temperature is a parameter related to autoregressive sequence generation which regulates stochasticity when sampling the multinomial distribution during inference. A lower temperature often causes the model to get stuck in repetitions.

In Figure 6 we display local conditioning [8, 17] over time. If we condition the model during training using a one-hot vector of size n for each of the n songs in the training data, then during generation we can change the value of this conditioning vector over time, which influences the audio to sound more like those particular songs.

Figure 4: Visualizing epoch, iteration, and beam_width metadata

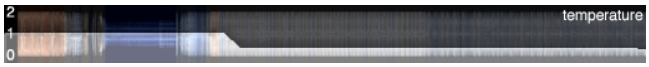

Figure 5: Visualizing change in temperature over time during inference

Figure 6: Visualizing change in local conditioning over time during inference

3.7 Fingerprinting

For fingerprinting, the entire dataset is iteratively loaded, segmented into ten-second chunks with five-second hops, and converted to dB mel-spectrograms. Dimensionality reduction is performed using Incremental PCA rendering 10 components per chunk. The learned PCA basis functions are shown in Figure 7.

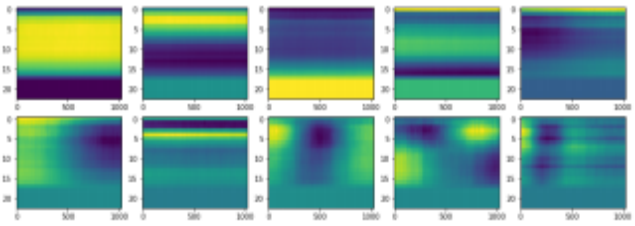

Figure 7: PCA basis functions for each of the 10 components. The y-axis represents the 23 mel-frequency bands and the x-axis represents STFT frames over time.

3.8 Grid building Process

The grid interface is 10x10. To fit thousands of chunks to this grid, k-means clustering is used on the PCA components with k=100. The cluster centroids are then spaced using t-SNE [7] , and quantized into the final 10x10 grid using Rasterfairy [5].

This process is intended to allow for the inspection of single audio chunks. With t-SNE visualizations there tends to be many overlapping points in the space which reduces the ease at which a single example can be previewed. t-SNE interfaces work well with short audio samples like those found in the infinite drum machine [20], but interface issues arise with larger sections of music. Rasterfairy, on the other hand, stretches these points out to cover an entire 2D grid.

3.9 Exploratory Interface Design

In grid view, each gridcell represents a cluster centroid and randomly displays one of the top 6 nearest neighbor chunks to the centroid. By clicking on a gridcell, the program sorts the chunks by distance to the centroid of that original audio chunk. In the list view, the full audio files which contain those chunks are listed. The user scrolls down in list view. The user seeks to any position in the audio by clicking. Highlighted sections can be exported as wav files. At the end of every audio file it finds a new file to play. This autoplay ability enables a continuous listening experience that we find useful while passively auditioning renderings.

Figures 8 and 9 show what the app looks like with the rolloff and chroma visualizations. The left side of the app is grid view. The right side of the app is list view.

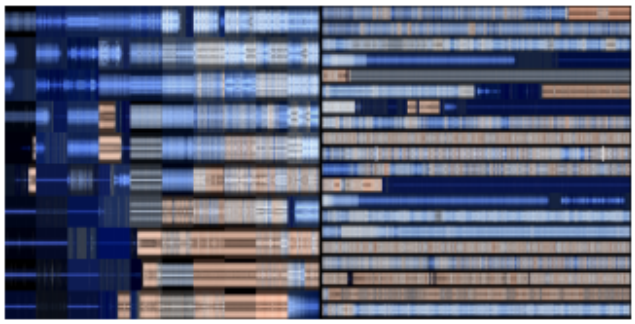

Figure 8: DOME in spectrogram rolloff mode. The left side is grid view, an overview of the diversity of sound which has been loaded. Clicking a gridcell changes the list view on the right, sorting the audio by closeness to that gridcell.

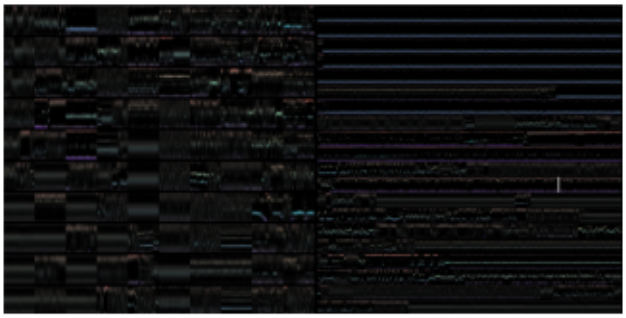

Figure 9: DOME in chromagram mode. At a glance, for example, the user can see melodic vs non-melodic sections, drones, melodies, and occurrences of the note G (cyan).

3.10 Crowd Curation

Our prototype works in all modern web browsers. This environment was chosen with a concern for cross-device compatibility. Our intention is to make curation more accessible for the purpose of crowd-sourcing. Export functionality allows the end user to rapidly collect variations of sonic elements and facilitates crowd-sourced collaboration.

4 INFORMAL EVALUATION

Informally, we gave three producers, experienced in sampling and curating music, the task of curating "interesting" pieces of music from 10 hours of SampleRNN generated audio. They first used their preferred method of curation (which included "load all 10 hours into Ableton Live and look at the raw waveform" and "hunt around with the MacOS Finder and hope for the best"), then later used DOME. They self-reported the curation process was between 5x and 20x faster with DOME.

Producer Drumcorps reported, "DOME was quite helpful in this project. Being able to scan through the audio content visually made it much easier to pick out useful and interesting sounds. After a short time browsing, I can get a sense of what a specific type of sound might look like, and I can start to find what I'm looking

for much faster. I found it good to build a little directory of my favorite clips from DOME, and work from there, rather than the usual 'hunt around with the MacOS Finder and hope for the best' method. The two methods are incomparable, the differences are night and day, and the results end up different as well. With the Finder, I start out listening to files, but then often get frustrated and just select a few files at random, then choose the best one from those. With DOME I end up finding a wide variety quickly - and then can choose further work from a more informed position - it gives me a larger sample size. There's only so much listening one can do in a day, and if you need to listen to samples in real-time to determine which you're going to use, that's less time available for getting down to the actual composition work. Some samples contain wild shifts and interesting artefacts within them - you'll see this in DOME right away and be able to listen to that piece immediately. With the Finder, it's listening and hoping. There's a place for randomness and hidden surprises too - but I find that when I'm trying to get something done quickly, DOME is most helpful."

5 FUTURE WORK

The use of PCA for fingerprinting is limiting. While it is helpful at clustering similar audio textures together, it is not powerful and precise enough to distinguish nuances in sound. In the future, for fingerprints, we could use embeddings from a trained deep net audio classifier.

The use of Librosa for analysis was slow. A 10-hour dataset takes a few hours to analyze on a MacBookPro. A C-compiled analyzer, or a distributed process (using cloud compute or AWS Lambda) would be more efficient.

Additional audio visualizations could include: the fingerprint embeddings over time, and annotating audio used when priming the sequence before generation.

With the addition of a composing feature, the end user could arrange music by sticking curated sections together. With the addition of an upvoting feature, the crowd could further curate their favorite sections and arrangements.

6 CONCLUSION

Steady progress has been made on fast generative raw audio with neural synthesis. With the advent of audio style transfer [12, 13, 21], one could render all possible permutations of style transfers, yet would still need a good way to explore the output. Digital audio workstations such as Ableton Live are not fit for this task. We designed an interface to minimize the time and effort required by listening to hours of similar audio clips. Care was taken in the visualizations to aid the user. This turned the time-consuming task of previewing hours of audio into something which can be done at a glance. We believe self-organizing interfaces like ours will be more important as large directories of generated audio can be rendered with faster inference speeds and greater parallelization.

A demo of this tool will be available online at dadabots.com/dome

REFERENCES

[1] CJ Carr and Zack Zukowski. 2018. Generating Albums with SampleRNN to Imitate Metal, Rock, and Punk Bands. *MUME* (2018). arXiv:1811.06633 http://arxiv.org/abs/1811.06633

[2] Sander Dieleman, Aäron van den Oord, and Karen Simonyan. 2018. The challenge of realistic music generation: modelling raw audio at scale. *CoRR* abs/1806.10474 (2018). arXiv:1806.10474 http://arxiv.org/abs/1806.10474

[3] Jonathan Driedger, Meinard Müller, and Sascha Disch. 2014. Extending Harmonic-Percussive Separation of Audio Signals. *ISMIR* (2014).

[4] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aäron van den Oord, Sander Dieleman, and Koray Kavukcuoglu. 2018. Efficient Neural Audio Synthesis. *CoRR* abs/1802.08435 (2018). arXiv:1802.08435 http://arxiv.org/abs/1802.08435

[5] Mario Klingemann. 2015. RasterFairy. https://github.com/Quasimondo/RasterFairy

[6] Peter Knees, Markus Schedl, Tim Pohle, and Gerhard Widmer. 2006. An Innovative Three-dimensional User Interface for Exploring Music Collections Enriched. In *Proceedings of the 14th ACM International Conference on Multimedia (MM '06)*. ACM, New York, NY, USA, 17–24. https://doi.org/10.1145/1180639.1180652

[7] L.J.P.V.D. Maaten and GE Hinton. 2008. Visualizing High-Dimensional Data using t-SNE. *Journal of Machine Learning Research* 9 (01 2008), 2579–2605.

[8] Rachel Manzelli, Vijay Thakkar, Ali Siahkamari, and Brian Kulis. 2018. Conditioning Deep Generative Raw Audio Models for Structured Automatic Music. *CoRR* abs/1806.09905 (2018). arXiv:1806.09905 http://arxiv.org/abs/1806.09905

[9] Brian McFee, Matt McVicar, Stefan Balke, Carl Thomé, Vincent Lostanlen, Colin Raffel, Dana Lee, Oriol Nieto, Eric Battenberg, Dan Ellis, Ryuichi Yamamoto, Josh Moore, WZY, Rachel Bittner, Keunwoo Choi, Pius Friesch, Fabian-Robert Stöter, Matt Vollrath, Siddhartha Kumar, nehz, Simon Waloschek, Seth, Rimvydas Naktinis, Douglas Repetto, Curtis "Fjord" Hawthorne, CJ Carr, João Felipe Santos, Jackie Wu, Erik, and Adrian Holovaty. 2018. librosa/librosa: 0.6.2. https://doi.org/10.5281/zenodo.1342708

[10] M.F. Medress, F.S. Cooper, J.W. Forgie, C.C. Green, D.H. Klatt, M.H. O'Malley, E.P. Neuburg, A. Newell, D.R. Reddy, B. Ritea, J.E. Shoup-Hummel, D.E. Walker, and W.A. Woods. 1977. Speech understanding systems: Report of a steering committee. *Artificial Intelligence* 9, 3 (1977), 307 – 316. https://doi.org/10.1016/0004-3702(77)90026-1

[11] Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron C. Courville, and Yoshua Bengio. 2016. SampleRNN: An Unconditional End-to-End Neural Audio Generation Model. *CoRR* abs/1612.07837 (2016). arXiv:1612.07837 http://arxiv.org/abs/1612.07837

[12] Parag K. Mital. 2017. Time Domain Neural Audio Style Transfer. *CoRR* abs/1711.11160 (2017). arXiv:1711.11160 http://arxiv.org/abs/1711.11160

[13] Noam Mor, Lior Wolf, Adam Polyak, and Yaniv Taigman. 2018. A Universal Music Translation Network. *CoRR* abs/1805.07848 (2018). arXiv:1805.07848 http://arxiv.org/abs/1805.07848

[14] Elias Pampalk, Simon Dixon, and Gerhard Widmer. 2004. Exploring Music Collections by Browsing Different Views. *Comput. Music J.* 28, 2 (June 2004), 49–62. https://doi.org/10.1162/014892604323112248

[15] Ryan Prenger, Rafael Valle, and Bryan Catanzaro. 2018. WaveGlow: A Flow-based Generative Network for Speech Synthesis. *CoRR* abs/1811.00002 (2018). arXiv:1811.00002 http://arxiv.org/abs/1811.00002

[16] Andreas Rauber, Elias Pampalk, and Dieter Merkl. 2002. Using Psycho-Acoustic Models and Self-Organizing Maps To Create Hierarchical Structuring of Music by Sound Similarity. (2002).

[17] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, R. J. Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, and Yonghui Wu. 2017. Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions. *CoRR* abs/1712.05884 (2017). arXiv:1712.05884 http://arxiv.org/abs/1712.05884

[18] Robin Sloan. 2018. Expressive Temperature. https://www.robinsloan.com/expressive-temperature/

[19] Sebastian Stober. 2010. MusicGalaxy - An Adaptive User-Interface for Exploratory Music Retrieval.

[20] Manny Tan and Kyle McDonald. 2017. Infinite Drum Machine. https://experiments.withgoogle.com/ai/drum-machine

[21] Dmitry Ulyanov. 2016. Audio Texture Synthesis and Style Transfer. https://dmitryulyanov.github.io/audio-texture-synthesis-and-style-transfer/

[22] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. 2016. WaveNet: A Generative Model for Raw Audio. *CoRR* abs/1609.03499 (2016). arXiv:1609.03499 http://arxiv.org/abs/1609.03499

[23] Fernanda Viégas and Martin Wattenberg. 2018. Visualization for Machine Learning (NeurIPS 2018 Tutorial). https://www.youtube.com/watch?v=ze08gwVPaXk

[24] Kazuyoshi Yoshii and Masataka Goto. 2008. Music Thumbnailer: Visualizing Musical Pieces in Thumbnail Images Based on Acoustic Features. In *ISMIR*.

[25] Zack Zukowski and CJ Carr. 2017. Generating Black Metal and Math Rock: Beyond Bach, Beethoven, and Beatles. *NIPS Workshop on Machine Learning for Creativity and Design* (2017). arXiv:1811.06639 http://arxiv.org/abs/1811.06639