Sounds Made With Machine Learning

Catalina Vallejos
Julian LaPlace

Introduction

For this project, our intention is to use an AI-assisted generative method to understand complex sounds from a variety of sources that fall inside the range of human hearing.

We make use of SampleRNN, a neural networking algorithm that learns directly from raw audio files. After training, it outputs a generator which can produce new sounds with similar statistical properties.

SampleRNN is a layered statistical process that works somewhere between sampling and synthesis, with a systematic numerical output that falls both inside and outside the material that can be heard with human ears.

The generators manufactured using SampleRNN vary widely in output. Counterintuitively, simple sounds cause it to overfit quickly, whereas complex sounds can train well, producing unexpected output. Analysis of its output can lead the listener to understand the overall interest expressed by this algorithm.

Method

First, we take an original sound and perform data augmentation. What this means is we make several copies of the source sound, timestretch the copies, concatenate them, and further subdivide them to produce the dataset format required by the SampleRNN library.

The concatenated samples are subsequently trained in a a 2-tier SampleRNN consisting of two Gated Recurrent Unit (GRU) layers, which look across multiple data frames, followed by a multi-layer perceptron (MLP) which operates at the sample level.

We use frame sizes of 8 and 2 for each of the GRU layers, 1024 neurons in each layer, 256 embedding size, 1024 sequence length, batch size of 128, and a sample rate of 44,100 Hz., and the initial state h0 is learned.

We train each model on a desktop computer built with an NVIDIA GeForce GTX 1080 GPU with 8 GB RAM.

Typically, coherent results are heard after training between 4 & 6 epochs, and an arbitrary judgement can then be made as to whether more training will be necessary. Each epoch takes about half an hour, and rendering 30 seconds of audio takes just as long, so we can get results in 4-6 hours.

For this project we experimented on 200 different datasets including single speaker recordings, group speech, various genres of music, harsh noise, personal recordings, physical sounds, and high-quality ambient audio engineering samples.

Results

Given strict creative confidentiality only the results of three trials will be presented in this poster.

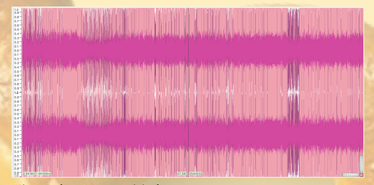

Fig 1.1 - clamstep.wav original waveform

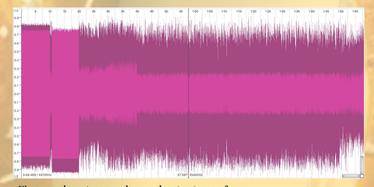

Fig 1.2 - clamstep.wav learned output waveform

The AI liked the noisier aspects of *clamstep.wav (figs 1.1-2.2)*. While it does not exhibit a consistent rhythm, it reproduces lots of midrange sounds and creates noise that sounds like it recycles pieces of the source material in interesting ways. While the rhythmic aspects were not well reproduced, it did reproduce some timbres and sounds from the source which are apparent when looking at the lower range of the spectrogram.

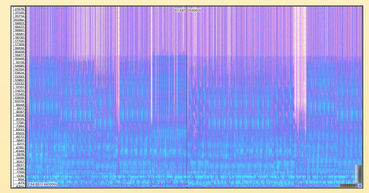

Fig 2.1 - clamstep.wav original spectrogram

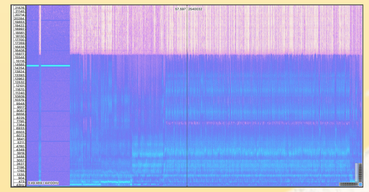

Fig 2.2 - clamstep.wav learned output spectrogram

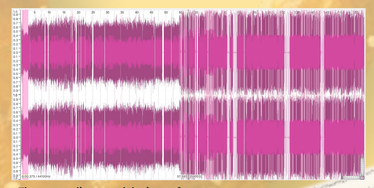

Fig 3.1 - textile.wav original waveform

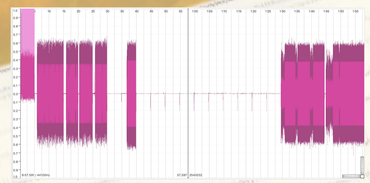

Fig 3.2 - textile.wav learned output waveform

The AI did not learn much about *textile.wav (figs 3.1-4.2)*. It quickly converged on a high pitched tone which does not occur in the source material. This is apparent when listening to the generated output. Retraining this sent would likely sustain the problem, given manifested in the first epoch and did not go away.

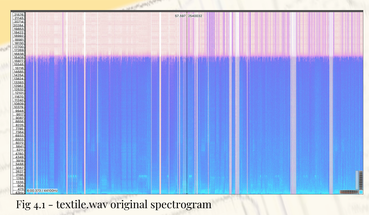

Fig 4.1 - textile.wav original spectrogram

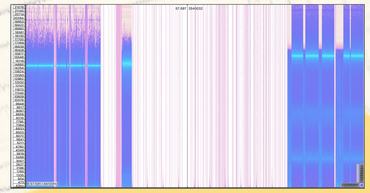

Fig 4.2 - textile.wav learned output spectrogram

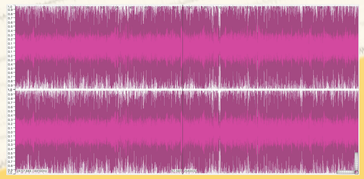

Fig 5.1 - glassbells.wav original waveform

Fig 5.2 - glassbells.wav learned output waveform

The AI likes the way *glassbells.wav (figs 5.1-6.2)* sounds especially. We see clear development of many overlapping tones which merge with and interrupt one another, producing complex and unpredictable output. The AI recognizes that harmonics belong to particular tones. The AI hears transients and uses these to switch from one sound to another.

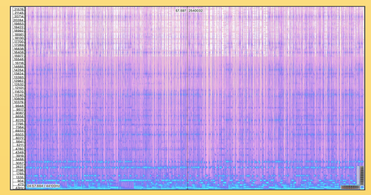

Fig 6.1 - glassbells.wav original spectrogram

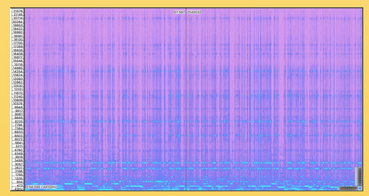

Fig 6.2 - glassbells.wav learned output spectrogram

Support & Funding

Kulturstiftung des Bundes / Podium Esslingen
Mr. Mat Dryhurst and Ms. Holly Herndon
Universität der Künste
The Hito Steyerl Lab

Special Thanks

Soroush Mehri for making SampleRNN, Kozakowski and Michalak at University of Warsaw who developed the PyTorch implementation we used.

Conclusion

We can tell if the AI learned anything by listening to the generated output. If problems of overfit occurred, it will be apparent as these fall into three clearly defined categories of silence, tones and noise. If the AI successfully reproduced any aspects of the original sound in a clear manner, then we consider this to be a successful training, and more training will likely continue to improve the sound.

In general, the AI likes sounds that contain a mix of tones and transients. To produce good results, the source sounds should have a mix of impulses and patterns, so the impulses cause it to switch from sound to sound, while also reproducing individual sounds coherently.

Works Cited

Cho, Kyunghyun, et al. "Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation." *Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation*, EMNLP 2014, 3 Sept. 2014, arxiv.org/abs/1406.1078v3.

Karpathy, Andrej, et al. "Visualizing and Understanding Recurrent Networks." *Visualizing and Understanding Recurrent Networks*, International Conference on Learning Representations, 17 Nov. 2015, arxiv.org/abs/1506.02078v2. Under review as a conference paper at ICLR 2016

Kozakowski, Piotr, and Bartosz Michalak. "SampleRNN in PyTorch." *DeepSound*, Deepsound.io, 29 June 2017, deepsound.io/samplernn_pytorch.html.

Kozakowski, Piotr, and Bartosz Michalak. "Samplernn-Pytorch." *Deepsound-Project*, GitHub, 20 Nov. 2017, github.com/deepsound-project/samplernn-pytorch. Source Code

Mehri, Soroush, et al. "SampleRNN: An Unconditional End-to-End Neural Audio Generation Model." *Arxiv.org*, International Conference on Learning Representations, 11 Feb. 2017, arxiv.org/abs/1612.07837v2. Published as a conference paper at ICLR 2017

Zukowski, Zack, and C J Carr. "Generating Black Metal and Math Rock: Beyond Bach, Beethoven, and Beatles." *NIPS 2017 Workshop*, 8 Dec. 2017, nips2017creativity.github.io/.