Neural Loops

Vibert Thio
Music and AI Lab, Academia Sinica*
ITP, NYU†
vibertthio@gmail.com

Chris Donahue
Stanford University
cdonahue@cs.stanford.edu

Abstract

Neural Loops is a co-creation tool which allows users to explore and manipulate musical loops synthesized by neural network generative models. In search of the perfect loop, users can independently manipulate both the musical score (notes and rhythms) and the waveforms used to perform the score as audio. Neural Loops is a step towards music creation software which does not assume expertise on the part of the user, but still allows the user to customize the produced music to taste.

1 Introduction

Tools for creating electronic music, such as *music sequencers*, require substantial expertise for desirable operation. On the contrary, AI systems which automatically create music at the push of a button provide no mechanism for user interaction, and hence yield no sense of ownership over the results. In this work, we seek to build a new type of music sequencer which can be operated with little to no experience, and can be intuitively steered towards producing musical content which satisfy a user's intentions.

Our system, *Neural Loops*, is a web-based intelligent music sequencer which allows users to explore a space of four-bar looping musical *trios*.[3] In our case, a trio consists of a melody line, a bass line, and a drum kit. Contrary to typical music sequencers which present users with a blank canvas, Neural Loops automatically populates a symbolic score (discrete notes and rhythms) upon initialization. Furthermore, Neural Loops generates an ensemble of instrumental sounds, which are sent to a sample-based playback engine to produce an audio rendition of the trio in real time (Figure 1a).

Our user interface (Figure 1b) displays a familiar grid-based view of the musical score to users. Simple buttons on the left and right of the grid allow users to change both the sounds used for playback and the score of each instrument. The intended workflow is that a user will repeatedly reinitialize Neural Loops until they find an interesting loop, then further refine it to their taste by manipulating the score, the sounds used for playback, and traditional musical sequencer controls such as tempo and audio effects.

2 System overview

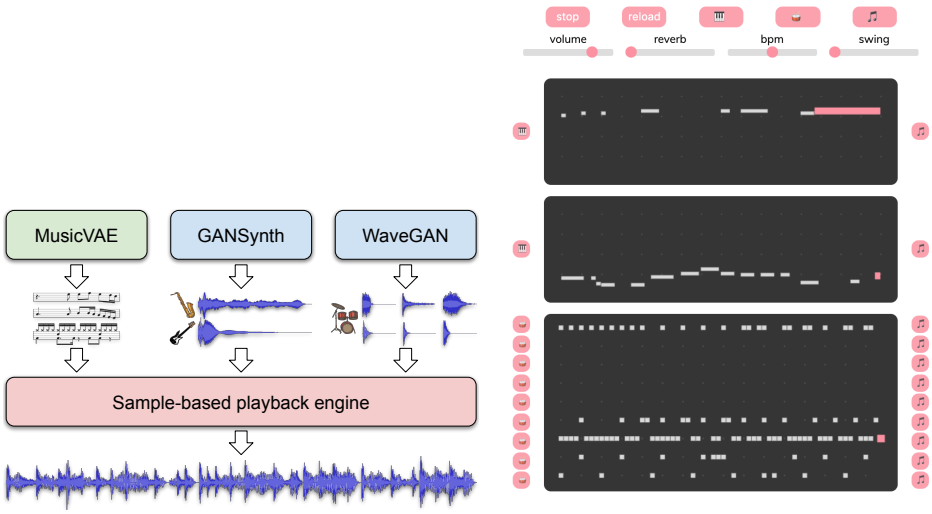
Producing a complete trio requires both *composition*, composing a musical score, and *sound design*, providing instrument sounds which will be used to perform the score. To perform these tasks, Neural Loops uses three distinct generative models: *MusicVAE* [Roberts et al., 2018] composes scores, and both *GANSynth* [Engel et al., 2019] and *WaveGAN* [Donahue et al., 2019] perform sound design.

*Music and AI Lab, Research Center for IT Innovation (CITI), Academia Sinica

†Interactive Telecommunications Program, New York University Tisch School of the Arts.

[3]Web demo: vibertthio.com/neural-loops (best viewed in Chrome on a GPU-enabled machine)
Code: github.com/vibertthio/neural-loops

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.



(a) System architecture (b) User interface (web)

Figure 1: Neural Loops is an intelligent musical sequencer which allows users to intuitively explore a space of four-bar musical trios. It merges three different symbolic and acoustic generative music models which were independently trained on different sub-tasks. For three instruments (melody, bass, and drum kit), users can independently manipulate both the symbolic score and the acoustic waveforms used by the sequencer to render the score as audio.

Specifically, MusicVAE models $P(\text{score})$, a distribution over scores, GANSynth models $P(\text{melody})$ and $P(\text{bass})$, distributions over melodic instrument waveforms, and WaveGAN models $P(\text{drum kit})$, a distribution over drum waveforms.[4] In typical musical production workflows, a user might first perform sound design (i.e., sample from $P(\text{sounds})$), and then compose a score which fits the sound palette (i.e., sample from $P(\text{score} \mid \text{sounds})$). For simplicity, Neural Loops makes an independence assumption upon initialization and samples a loop at random from the distribution

$$P(\text{loop}) \approx P(\text{score}) \cdot P(\text{melody}) \cdot P(\text{bass}) \cdot P(\text{drum kit}). \tag{1}$$

Subsequently, users can individually manipulate the components, e.g. by manipulating the drum kit sounds while keeping the score fixed. Hence, by fine tuning their loop in our interface, users can remove the independence assumption and produce better loops that suit their tastes.

3 User interface design

Our user interface (built using [Thio et al., 2019]) consists of both traditional musical sequencer controls, which control various aspects of the sample-based playback engine such as volume and tempo, and intelligent ones, which manipulate the components of our factorized generative model (1). The intelligent controls at the top of our interface are coarse-grain. Here, users can replace the melodic sounds (sample from $P(\text{melody}) \cdot P(\text{bass})$), replace the entire drum kit sounds (sample from $P(\text{drum kit})$), replace the score (sample from $P(\text{score})$), or all of the above my sampling from $P(\text{loop})$ as in (1).

Below, three grids display the score for the melody, bass, and drum kit instruments respectively. To the left of each grid is a button which will change the sounds for an individual instrument. To the right of each grid is a button which will change the score for that instrument while keeping others fixed. To replace the score for an individual instrument, we first send the complete current score through the encoder of MusicVAE to produce a latent vector. Then, we add a small amount of noise to the vector, and sample a new score from the decoder. Finally, we swap in the score for the instrument in question from the decoder output into the old score, keeping other instruments fixed. This is necessary to manipulate scores for individual instruments such that the new part agrees with the existing parts.

[4]For our application, we train a custom WaveGAN which takes drum types (e.g., kick drum) as conditioning info. We use both MusicVAE and GANSynth off the shelf through their Magenta.JS implementations.

4 Ethical implications

There are numerous ethical considerations with regards to AI generated music. Perhaps most concerning is the potential to displace human musicians. We believe that AI-powered *co-creation* tools, which pair humans with intelligent systems, mitigate this particular concern—these tools do not subvert humans but rather expand their creative capacities.

References

Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. In *ICLR*, 2019.

Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. GANSynth: Adversarial neural audio synthesis. In *ICLR*, 2019.

Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *ICML*, 2018.

Vibert Thio, Hao-Min Liu, Yin-Cheng Yeh, and Yi-Hsuan Yang. A minimal template for interactive web-based demonstrations of musical machine learning. In *Workshop on Intelligent Music Interfaces for Listening and Creation at ACM IUI*, 2019.