Drexel University
ExCITE Center
Expressive and Creative Interaction Technologies

Drexel.edu/excite
@excitecenter
Sophia’s Forest

music: Lembit Beecher, libretto: Hannah Moscovitch
This is DRUMHENGE
A New Musical Instrument
MET-LAB ALUMS IN MIR

Erik Schmidt
Sr. Scientist,
Pandora

Jeff Scott
Research Engineer,
Gracenote

Matt Prockup,
Scientist,
Pandora
Dr. David Rosen

PhD Thesis
The Neural Substrates of Expertise and Flow Among Jazz Guitarists

Anyone looking to hire a music neuroscientist?
Learned Timbral Controllers for Augmenting Parametric Synthesizer Interfaces
Jeff Gregorio and Youngmoo E. Kim • {jgregorio, ykim}@drexel.edu
Electrical and Computer Engineering, Drexel University

Background
Feature-based synthesis applies machine learning and signal processing methods to the development of alternative interfaces for controlling parametric synthesis algorithms. One approach geared toward real-time control, uses low dimensional gestural controllers and learned mappings from control spaces to parameter spaces, making use of an intermediate latent timbre distribution, such that the control space affords a spatially-intuitive arrangement of sonic possibilities. This work attempts to address questions regarding user experience in such systems, including the accuracy of user mental models, and how these techniques can be integrated in a way that simplifies interaction for novices while affording new abilities to experts without encumbering existing modes of interaction.

Proposed System
• Though others have explored learned many-to-many mappings from control spaces to parameter spaces based on timbre [1][2], such interfaces necessarily supplant traditional one-to-one interfaces due to the lack of integration between the two spaces.
• One system integrates the two, but uses no timbral arrangement of the control space. [3]
• The proposed system uses an invertible mapping layer allowing inference of parameter values from control space coordinates, but also ensures updates made in parameter space (using the one-to-one interface) can project into the control space, which provides strong visual intuition for the equivalence of the two spaces.

Deep Latent Gaussian Model

![Figure 1](image1.png)

Figure 1: (Left) Overview of system during training and runtime. Encoded training examples are used to predict known parameter values. Training yields a normally-distributed latent encoding with predictive dimensions. Post-training, principal component analysis (PCA) re-orient the latent space and parameters are exported. (Right) Invertible runtime mapping model, consisting of (top to bottom) a uniform to normal scaling layer (using the normal CDF), PCA projection, and a dense layer.

Future Work
• While unimodal latent space constraints suffice for low-dimensional synths, multimodal latent spaces modeled by Deep Latent Gaussian Mixture Models (DLGMMs) may be more appropriate for high-dimensional synths, whose latent spaces are over-regularized by the current system.
• A systematic evaluation is needed to address whether the visual equivalence of parameter and control spaces is necessary to accurately understand the system, and whether control and parameter spaces support different modes of creation and ranges of synthesis expertise. We plan on conducting a user study consisting of open-ended exploration, musical tasks, and user interviews.

References

![Figure 2](image2.png)

Figure 2: Example application using a 3D controller to control a 5D low-frequency oscillator (LFO).
Introduction

While autoregressive generative models for waveform audio trained on musical datasets achieve high fidelity results, there is still a question of how to design these models to make them more expressive from the user end? One approach to achieving this is the use of conditioning on both local (time-varying) variables like MIDI and global variables such as instrument or genre. However, this requires well-annotated musical datasets. We approach this problem by using information theory to learn informative latent variables.

Methods

Generating any length of audio

We propose using a recurrent layer as the first layer of the network to replace the dense layer in WaveGAN. This allows us to model the input to the network, $z$, as a $(N_{\text{frames}} \times N_z)$ matrix rather than a $1 \times N_z$ vector. In training, we use $N_{\text{frames}} = 16$ to generate 16384 samples, corresponding to 1.024 seconds at a sampling rate of 16 kHz. At test, however, we use any integer $N_{\text{frames}}$ to generate a $1024 \times N_{\text{frames}}$ long sample of audio. This extends the capabilities of the network after training by exploiting the statefulness of the recurrent layer.

InfoGAN

InfoGAN [2] extends the standard GAN model by proposing a method for learning latent codes, $c$, that follow any distribution (Gaussian, Uniform, Categorical, etc.) that are appended to the standard, non-informative latent variable, $z$. They argue that these latent codes should have an impact on the generated images by maximizing mutual information between the learned codes and generated images, $I(c, G(z,c))$. Since this is intractable, however, they use variational inference to estimate the lower bound of mutual information.

Results

We trained a network that had multiple continuous variables together and one categorical variable on the MAESTRO dataset. This dataset features over 80 hours of professional piano playing. From early results, the model is able to separate piano playing on the lower register from the higher register and levels of reverb, sustain pedal usage, and volume as features. Further experiments are required to confirm this algorithm generalizes well to other types of datasets.

Future Work

Better Quality Generation

We hope to attempt other GAN training algorithms like Progressive growing GANs to see if it is possible to get competitive results with autoregressive models. Alternatively, it may be possible to use an auto-regressive model that learns latent codes using mutual information.

Local Conditioning

We would like to explore using both ground truth labels and learned variables together. Ideally, figuring out a way to extend this method while conditioning on MIDI data, letting the learned codes focus on non-labels features such as reverb and sustain.

References


Automatic Guitar Tablature Transcription with Convolutional Neural Networks
Andy Wiggins and Youngmoo E. Kim • {awiggins, ykim}@drexel.edu
Electrical and Computer Engineering, Drexel University

Motivation
Guitarists commonly use tablature notation to learn and share music. As it stands, most tablature is created by an experienced guitarist taking the time and effort to annotate a song. As the process is time consuming and requires expertise, we are interested in automating this task. Previous approaches to automatic tablature transcription [1, 3] break the problem into two discrete steps: 1) polyphonic pitch detection followed by 2) tablature fingering estimation. Using a convolutional neural network (CNN) model, we can learn a mapping directly from audio data to tablature. The model can simultaneously leverage physical playability constraints and differences in string timbres to determine the actual fingerings being used by the guitarist. We propose TabCNN, a convolutional neural network for transcribing guitar tablature from audio of a solo acoustic guitar performance.

TabCNN
Model: TabCNN is a convolutional neural network that takes as input an image representing a short window of isolated guitar audio and outputs a probability mass function for each string’s fret classification. (See model architecture in Figure 1 below.)

Dataset: We use the GuitarSet dataset [2], which contains acoustic guitar performances in a variety of musical keys and playing styles. The dataset’s string-wise pitch annotations are sampled to produce ground truth tablature labels.

Preprocessing: The audio is segmented into 200ms clips. Each clip is downsampled to 22050Hz, and then the magnitude Constant-Q Transform (CQT) is computed, with 24 frequency bins per octave, spanning 8 octaves. Using a CQT reduces dimensionality and offers linearity in time and pitch, which can be exploited by the model’s convolutional layers.

Training: We train the model for 30 epochs using a 6-dimensional categorical cross-entropy loss function. Dropout regularization is used to reduce overfitting.

Results

<table>
<thead>
<tr>
<th>Guitar String</th>
<th>Fret Classification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st (e)</td>
<td>0.898</td>
</tr>
<tr>
<td>2nd (B)</td>
<td>0.760</td>
</tr>
<tr>
<td>3rd (G)</td>
<td>0.801</td>
</tr>
<tr>
<td>4th (D)</td>
<td>0.808</td>
</tr>
<tr>
<td>5th (A)</td>
<td>0.884</td>
</tr>
<tr>
<td>6th (E)</td>
<td>0.921</td>
</tr>
</tbody>
</table>

Average: 0.845

Future Work
• The current system determines tablature window by window, and does not take into account the sequence over the course of the performance. The addition of a recurrent layer to model the progression of labels over time will help smooth the output tablature sequence.
• Data augmentation may help reduce any overfitting in the model. Additional training data can be constructed by pitch shifting the training audio and adjusting the tablature labels accordingly.

References

Figure 1: The TabCNN model architecture.

Figure 2: (Left) Example input-audio/output-tablature pairs predicted by TabCNN during testing. (Right) Table of string-wise and average accuracy metrics calculated during testing.
Summer Music Technology
Class of 2018
Report on Integrating Higher Education in the Arts & Humanities with Science, Engineering, and Medicine

Free download from National Academies Press
"Masters in Making" starting Fall 2019