ABCD: A Bach Chorale Discriminator

Jason d'Eon
Dalhousie University, Vector Institute

Sageev Oore
Dalhousie University, Vector Institute

Abstract

Many generative models for music revolve around the Bach Chorales: a set of pieces from the 1700s with highly constrained harmonic patterns. In this work we demonstrate a method of finishing incomplete parts of a Bach Chorale using a discriminating model, which can only identify whether or not a musical sample is like a Bach Chorale. The model consists of an ensemble of fully-connected neural networks and was trained on modest hardware.

1 Introduction

The Bach chorales [1] have constituted part of the canon of music datasets for machine learning since the earliest explorations in the field. Harmonet [2] found success in directly representing the straightforward harmonies found in these pieces. Even though modeling long-term structure was out of reach at the time, early music generation research in the 90's modeled after Bach's music sounded pleasing locally [3]. More recent work revolving around Bach chorales involves BachBot [4], which implements LSTMs, DeepBach [5], which combines RNNs and MCMC methods, and Coconet [6], which uses a CNN with a Gibbs sampling approach.

Music students typically learn to analyze and write in the tradition of Bach Chorales in introductory courses on harmonic analysis. The Bach Chorales are remarkably regular: they consist of 4-part harmony, they are generally 8 to 12 measures long, they consist of almost entirely eighth notes, quarter notes and half notes with almost no sixteenth notes and without triplets, subsequent analysis has shown that they follow certain rules almost rigorously, and many people find them deeply pleasing to listen to. In short, they exemplify a highly structured form of composition tightly following a small and simple set of rules.

As a dataset for machine learning, however, some other characteristics are worth mentioning. Notably, there are only a few hundred chorales, and most rules that govern them are not about (large-scale) musical form, but rather local rules about what note a voice might move to next given its previous note and the neighbouring notes of the other three voices. Therefore, much of this structure can be captured by simple models that do not support learning long-term correlations. In this work we show that a fully-connected feed-forward network is able to generate pleasing outputs by filling in partially complete Bach Chorales. Like previous methods, we re-sample parts we wish to rewrite and use a discriminating model to score the sample, similar to a generative adversarial network (GAN) [7, 8].

2 Dataset

We used a version of the Bach Chorales dataset found here[1], using an 8th-note quantization. The chorales were divided into a train/validation/test split of $229/76/77$. For each of the 4 voices, we restricted the available pitches to 3 octaves, resulting in 37 one-hot tokens (including silence). For pre-processing, we transformed the input into a shape of $(6, 4, 37)$, which corresponds to $(\texttt{seq\_len}, \texttt{num\_voices}, \texttt{num\_tokens})$.

[1] `https://github.com/czhuang/JSB-Chorales-dataset`

5th Workshop on Machine Learning for Creativity and Design at NeurIPS 2021, Online.

To generate training examples for the discriminator, we transposed by $\{-3, -2, \ldots, 3, 4\}$ semitones, chosen uniformly at random. There was a 0.5 probability that the sample was corrupted by the following method:

- Choose 1-8 notes in the (2×4) center of the window to be corrupted
- For each note, corrupt by the following mixture distribution:
  - $(p = 0.5)$ Sample a pitch from a Gaussian centered on the existing note, with $\sigma = 3$ semitones, forcing the new pitch to be distinct
  - $(p = 0.5)$ Sample uniformly from all 37 possible tokens

3 Model and Rejection Sampling

The discriminator model consists of an ensemble of fully-connected feed-forward networks. We trained an individual network for each voice, every combination of 2 voices, every combination of 3 voices, and the full 4 voices. The size of the input is $(6 \times \texttt{num\_voices} \times 37)$, followed by 1 hidden layer of size 256, and 2 output logits. All activation functions are ReLUs. We optimized log-likelihood using Adam with a learning rate of $1e{-3}$ and training for 150 epochs. Other Adam settings were set to PyTorch defaults.

We use rejection sampling to leverage the discriminator for generation: we select a window in the chorale and feed it as input to the model before and after making a change (see Fig 1). We compute a score either by taking the mean or the minimum of all of our models' softmax probabilities, and keep modifications if the score is increased. Performing this process iteratively allows us to converge on a sample resembling a true Bach chorale.

We tried various methods of generating/modifying notes to test for rejection. We found that enforcing the windows to always include frozen notes from true samples (i.e. yellow notes in Figure 1) had a noticeable effect on the quality of generation, whereas allowing windows to overwrite each other resulted in thrashing. To start generation we sampled mid-range pitches uniformly, and then we perturbed pitches by adding Gaussian noise. We typically ran this process for 100,000 re-samplings, which is feasible on a modest CPU.

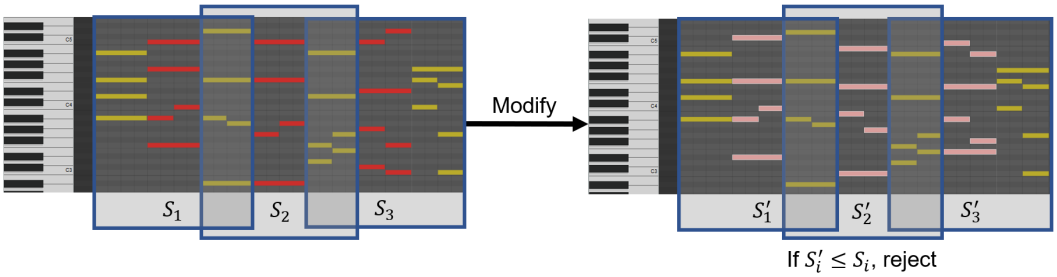

Figure 1: An example of using the discriminator for rejection sampling. We modify notes (red) within selected time-windows and individually prompt the discriminator as to whether or not the change is more Bach-like. The scores, S_i, are averaged softmax probabilities of the ensemble for each window.

4 Examples

To demonstrate the result of our process we have some examples before and after generation. (True Chorale) refers to the ground-truth sequence, (Input) shows which parts of the sequence were fixed, and (Output) is the generated result.

1. (True Chorale, BWV 55.5) $\longrightarrow$ (Input) $\longrightarrow$ (Output)
2. (True Chorale, BWV 122.6) $\longrightarrow$ (Input) $\longrightarrow$ (Output)
3. (True Chorale, BWV 393) $\longrightarrow$ (Input) $\longrightarrow$ (Output)

Ethical Implications

It is very unlikely that this work will have a negative impact from an ethical standpoint. It involves a generative model based on a music dataset which is in the public domain.

Acknowledgments and Disclosure of Funding

Resources used in preparing this research were provided, in part, by NSERC, the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute `www.vectorinstitute.ai/#partners`.

References

[1] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*, 2012.

[2] Hermann Hild, Johannes Feulner, and Wolfram Menzel. Harmonet: A neural net for harmonizing chorales in the style of JS Bach. In *Advances in neural information processing systems*, pages 267–274, 1992.

[3] Michael C Mozer. *Connectionist music composition based on melodic, stylistic, and psychophysical constraints*. University of Colorado, Boulder, Department of Computer Science, 1990.

[4] Feynman Liang. Bachbot: Automatic composition in the style of Bach chorales. *University of Cambridge*, 8:19–48, 2016.

[5] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. Deepbach: a steerable model for bach chorales generation, 2017.

[6] Cheng-Zhi Anna Huang, Tim Cooijmans, Adam Roberts, Aaron Courville, and Douglas Eck. Counterpoint by convolution. *arXiv preprint arXiv:1903.07227*, 2019.

[7] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. Midinet: A convolutional generative adversarial network for symbolic-domain music generation. *arXiv preprint arXiv:1703.10847*, 2017.

[8] Olof Mogren. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:1611.09904*, 2016.