BachBot: Deep generative modeling of Bach chorales

Feynman Liang[1], Marcin Tomczak[1], Matt Johnson[2], Mark Gotham[3], Jamie Shotten[2], Bill Byrne[1]

[1]Cambridge University Engineering Department, [2]Microsoft Research Cambridge,

[3]Faculty of Music, University of Cambridge

Objectives

The goal of BachBot is to generate 4-part Baroque chorales in the stype of Johann Sebastian Bach. We interpret this as sampling random chorales from a generative probabilistic model of Bach chorales and identify four discrete objectives:

- **Melody modeling**: Marginal distribution over univariate Soprano melody sequences
- **Melody harmonization**: Conditional distribution over multivariate (Alto, Tenor, and Bass) harmony parts given fixed Sporano melody
- **One-pass polyphonic generation**: Sequentially modeling of all parts jointly
- **Applications in music analysis**: What would Bach do? Enhance "Bach-ness" of inputs.

Background

Music is quantized to the nearest 16th beat (the smallest metric interval occuring in Bach chorales) and quantized to the twelve note chromatic scale used in Western music. We define two equivalent representations for music:

- The **per-parts tuples** representation consists of four collections (one for each voice) of note/rest and duration tuples. Note that durations between adjacent tuples are non-uniform.
- The **per-parts roll** consists of a categorical array $X_{p,t} \in V$ denoting the note played by part $p \in \{S, A, T, B\}$ at time $t \in \{1, 2, \cdots, T\}$ To distinguish notes held from previous times from notes articulated at the current time, $Y_{p,t} \in \{0, 1\}$ indicates if the note is articulated at time t.
- The **piano roll** is like the per-parts roll except $X_{n,t} \in \{0, 1\}$ denotes if note n is played at time t. Note that this definition precludes two parts playing the same note.

Melody Modeling

- Given initial seed, generate melody sequence
- Baseline: N-gram language model perplexity
- Experiments:
  - RNN, LSTM, and GRU architectures
  - Word level vs note level features, augmentation with expert crafted features
  - Constant (roll representation) or varying timestep (tuple representation) per input

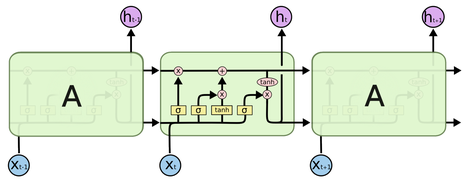

Figure 1: Inside of an LSTM cell[1]

One-pass polyphonic generation

- Given initial seed, generate entire chorale
- Baseline: n/a, subjective evaluation
- Experiments:
  - Bi-axial and grid architectures
  - Convolutional vs recurrent dimensions

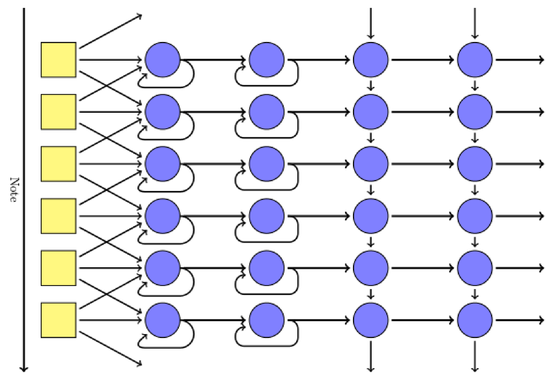

Figure 3: Biaxial RNN Architecture[3]

Melody Harmonization

- Given melody, generate the harmony parts
- Baseline: HMM-based system[2] accuracy
- Experiments:
  - Single multivariate vs 4 independent LSTMs with MRF refinement
  - Bi-directional LSTM

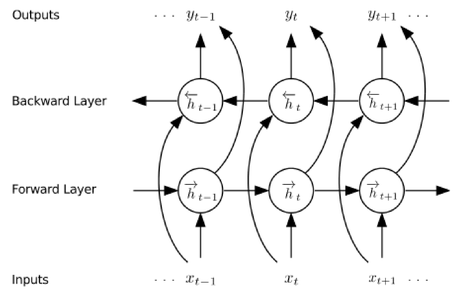

Figure 2: Bidirectional RNN hidden states

Applications in Music Analysis

- MAP estimation: how would Bach complete this note/melody/chorale?
- Interpreting hidden state activations
- Adversarial net: enhance "Bach-ness" of given input

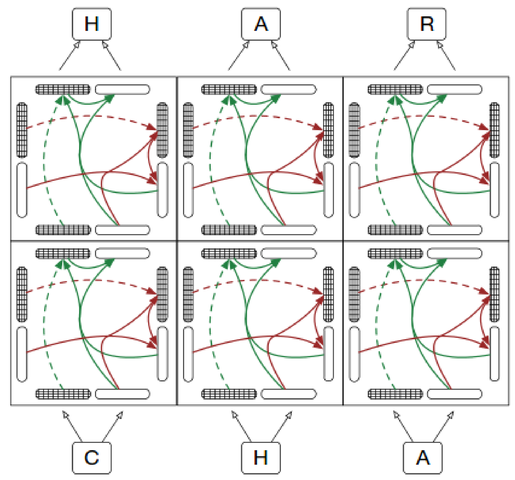

Figure 4: 2D grid RNN Architecture[4]

Project status

- Completed
  - Preprocessing pipeline (strip markup, extract 4 parts, transpose C major/A minor) complete
  - `torch` 2-layer LSTM melody model
  - `keras/tensorflow` bi-axial LSTM modely model
- Upcoming
  - Get baselines for N-gram melody model and HMM-based harmonization [2]
  - Augment feature representation with expert crafted features
  - Investigate GRUs and vanilla RNNs for melody modeling and harmonization
  - Implement and compare biaxial vs grid RNNs for harmonization and single pass generation
  - Sample outputs and perform subjective evaluation using MTurk

References

[1] *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013*. IEEE, 2013.

[2] Moray Allan and Christopher KI Williams. Harmonising chorales by probabilistic inference. *Advances in neural information processing systems*, 17:25–32, 2005.

[3] Christopher Olah. Understanding lstm networks, 2015.

[4] Nal Kalchbrenner, Ivo Danihelka, and Alex Graves. Grid long short-term memory. *CoRR*, abs/1507.01526, 2015.

Acknowledgements

We gratefully acknowledge the support of Microsoft Research Cambridge with the donation of GPUs used for this research.

Contact Information

- https://github.com/feynmanliang/bachbot
- fl350@cam.ac.uk

UNIVERSITY OF CAMBRIDGE
Department of Engineering