An Adaptive Model of Pulse in Jazz Percussion: Rhythmic Generation in Quasi-Periodic Musical Contexts using Sequence-to-Sequence Learning

Nolan Lem
Department of Computer Science
Stanford University
`nlem@ccrma.stanford.edu`

Abstract

This project attempts to formulate a novel approach to musical rhythm synthesis by incorporating research from beat entrainment and pulse estimation into an audio-to-midi neural network based pipeline. In light of the success of sequence-to-sequence language models [17], this project oversees a "beat-to-beat" translation model to determine to what extent a network can characterize the rhythmic behavior of a particular jazz drummer using feature embeddings and a non-quantized output grid. This "beat-to-beat" encoder-decoder recurrent neural network (RNN) is trained on data associated with pulse (local and spectral beat features) extracted from audio in order to reproduce rhythmic sequences conditioned on prior estimates both locally and globally. After training, we can generate (and predict) sequences of musical rhythms that have been conditioned on the past rhythmic material of the performer. The model was shown to be able to infer future rhythmic sequences with a maximum accuracy of 30% on the test set. A small survey was conducted to gauge the qualitative performance of the generated rhythmic sequences in comparison to a ground-truth baseline.

1 Introduction

The establishment of the pulse percept, the basic unit of perceived rhythm that constitutes a musical beat, is fundamental to the way in which we listen, engage, and perform with music [12]. Pulse, an isochronous grid most suitable to organize a given temporal pattern, is the way in which it orientates cognitive processes involved in anticipation, expectation, and arousal. Taking cues from connectionist theories related to musical structure, pulse can be thought of as an organizational unit from which we derive emergent orders of temporal structure [14]. Pulse extraction is the means through which we are able to manage time-dependent events, namely those that occur with some level of regularity. In fixed-tempo music, pulse often overlaps with the notion of the beat however in genres of music where time or tempo is more variable, pulse can vary with time.

Many neural network models for music generation focus on genres of music that are structured around fixed-beat time units (e.g. classical music, rock and pop) where musical material is constructed in reference to an isochronous pulse. However unlike many styles of popular music, a signature feature of jazz performance is the way in which time can expand and contract in the course of an improvisation. Jazz musicians will typically latch onto (or deviate from) a perceived pulse during a performance and this adaptive sense of pulse entrainment is particularly relevant in characterizing music of a quasi-periodic nature that includes contemporary classical music and free jazz. In this

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

paper, I attempt to tackle the problem of teaching a network how to infer pulse fluctuations in the course of a performance by introducing a generative RNN model that attempts to improvise around an explicitly non-isochronous pulse.

The input to the proposed neural network are beat descriptors (local pulse estimates and spectral onset envelopes) and drum set onsets (snare, hi-hat, and kick drum) which were extracted from raw audio of solo drum improvisations while the network outputs contain the same feature set with variable sequence lengths. The network presented here is a sequence-to-sequence LSTM encoder-decoder network that is trained to produce sequences of rhythms that were conditioned on prior estimates of the pulse. After training, the network can generate rhythmic sequences that have been seeded by inputs from a test set.

2 Related work

A common challenge in working with sequence-based RNNs (e.g. word generation, time series prediction, etc.) is teaching the network to capture temporally distant inputs. Because of a vanishing (or exploding) gradient, when RNNs are trained with stochastic gradient descent they can have difficulties learning long term dependencies in the input. Recently, there have been many approaches to neural-network based music generation for jazz many of which use Long-short Term Memory (LSTM) [11] based RNNs [4][7][9][16].

Several of these approaches use variational autoencoders to generate semantically meaningful latent representations that can be traversed to generate related melodies, chords, and musical rhythms [8][16]. By interpolating between points in the latent space, these networks have the advantage of being able to generate novel music materials by sampling from a continuous distribution. The ıMetallicaLSTM network [4] explicitly encoded a reference to the beat as an input feature to the network where the word ‘<bar>’ signals the end of each measure. This has the advantage of teaching the network the underlying metrical grid of the individual training samples even when a song is in a different time signature. Because many of these models focus on jazz harmony or melody generation, their inputs are MIDI representations that quantize time into subdivisions representing an eighth note or sixteenth note subdivision. While this approach simplifies the learning process and speeds up computation, it also forces the generated output to always land on an equal beat subdivision which is less suitable for the domain of free improvisation.

By allowing one feature to mark/designate time within a beat, using some subdivision of the beat, the network will be better able to understand how musical events are orientated with respect to underlying metrical structures. However in the case of music that does not exist within an isochronous temporal frame, such as free jazz or forms of contemporary classical music, this problem becomes intractable in the absence of any strict tempo information. Using beat estimates of audio derived from a particular neural network based model, this project attempts to teach a recurrent network how to impose rhythmic events on top of shifting estimates of perceptual pulse information to better encode temporal structure in specific improvisation contexts. More specifically, I employ a recurrent sequence-to-sequence encoder-decoder network that is parameterized by the beat structure of the input data. By using the perceived beat (pulse) as a way to structure sequence learning, this network can better adapt to changes in tempo which allows it to generalizes to the flexible nature of time in free improvisation.

3 Dataset and Features

The input dataset consisted of approximately 1.2 hrs of uncompressed audio files (wav) sampled at 44.1kHz of solo jazz drum improvisations of the jazz drummer Paul Motion. 80% of the input was used as the training set, and the remainder 20% was split equally into the validation and test sets.

This audio went through a significant preprocessing stage to extract the relevant features for the network. This is shown in the top half of Figure 2. The audio is analyzed for global beat estimates and extracts a spectral onset envelope. The global tempo estimates used an offline dynamic programming model [6] to recursively calculate beat locations in the audio file; this function is capable of detecting tempo changes throughout the course of the audio being observed. The spectral onset envelope is another beat estimate feature but one that prioritizes local temporal events that incur a significant spectral flux (for more information, see appendix). These features operate at the frame level at the output of a short-time Fourier transform (hopsize=512) with each frame representing approximately 12 ms. The global beat estimate and spectral onset envelopes are combined to form a single vector which is segmented into 5 second (431 frames) clips.

To extract the other features, the audio is put through a pre-trained RNN (Automatic Drum Transcription (ADT) library[1]) to source separate the individual instruments in the drum set (high-hat, snare, and kick drum) and output a transcription of their rhythms. This notation was translated into a vectorized MIDI format and reformatted to be of the same length as the beat estimate vector with each time unit is 12ms. Both the beat features (local and global onsets) and percussion (MIDI) onset features per time frame were combined to create a feature vector of length 5.

The number of time steps per sample passed to the encoder and decoder for training and teacher forcing is parameterized by differences between preceding and future global beat estimates which I refer to as the beat gradients, ΔL_e and ΔL_h. The number of time steps was determined by difference between the current global beat estimate, $L[n]$, and the previous beat estimate $L[n-1]$ whereas the number of time steps for the target sequence is set by the different between $L[n]$ and $L[n+1]$. This is shown in Figure 1.

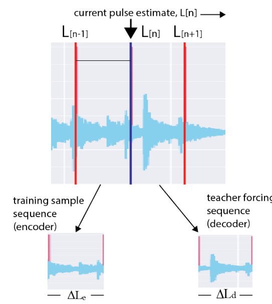


Figure 1: The beat gradients

4 Methods

The neural network proposed here is a sequence-to-sequence model with a single layer and 30 LSTM units in both the encoder RNN and decoder RNN. Unlike vanilla recurrent ANNs, the LSTM cells can allow the network to (ideally) handle long term dependencies in the input sequences by including input gates, forget gates, output gates and a new memory cell.

For sequence to sequence generation, the LSTM units can be thought of as trying to estimate the conditional probability of an output sequence, $y_{1...T^1}$ given an input sequence, $x_{1...T}$ where T_1 doesn't necessarily equal T. To determine this conditional probability, we can use an LSTM encoder network to obtain a fixed-dimensional representation, v, of the input $x_{0...T}$ given the last hidden state of the LSTM. Then we can compute the probability of $y_{1..T_1}$ using a LSTM decoder network whose hidden state is set to v. This is shown in equation .

$$p(y_1, ...y_{T^1}|x_1, ..., x_T) = \prod_{t=1}^{T^1} p(y_t|v, y_1, ..., y_{t-1})$$

An "end-of-sentence" (EOS) symbol is applied at the end of each generated sequence so that we can can define a distribution over sequences of all possible lengths.

The model proposed here differs from this one in several ways. Most notably, the location of the 'end-of-sentence' symbol in each sequence is predetermined during training using the beat gradient, $\Delta L_{e,d}$ where $x_T = \Delta L_e$ and $y_{T^1} = \Delta L_d$. Figure 2 illustrates the sequence to sequence network under consideration. Since the input is a multi-variate time series (multiple features), the loss function was categorical crossentropy.

5 Experiments - Results - Discussion

All the code for these experiments was written in Keras 2.0 using tensorflow as a backend. This model was trained using stochastic gradient descent (batch size=1), adam optimization and a learning rate of 1.0 that was decayed by a factor of 0.8 after epoch 20. The initial weights were randomly initialized within a range between -0.05 and 0.05. During training, I applied dropout to the units with a probability of 20% The input and target data sequences were processed 'statefully' (meaning that the prior states are used as the initial state for the next batch). Because the input sequences were arranged in order (according to the beat gradients), this was performed so that the network might

[1] https://github.com/CarlSouthall/ADTLib

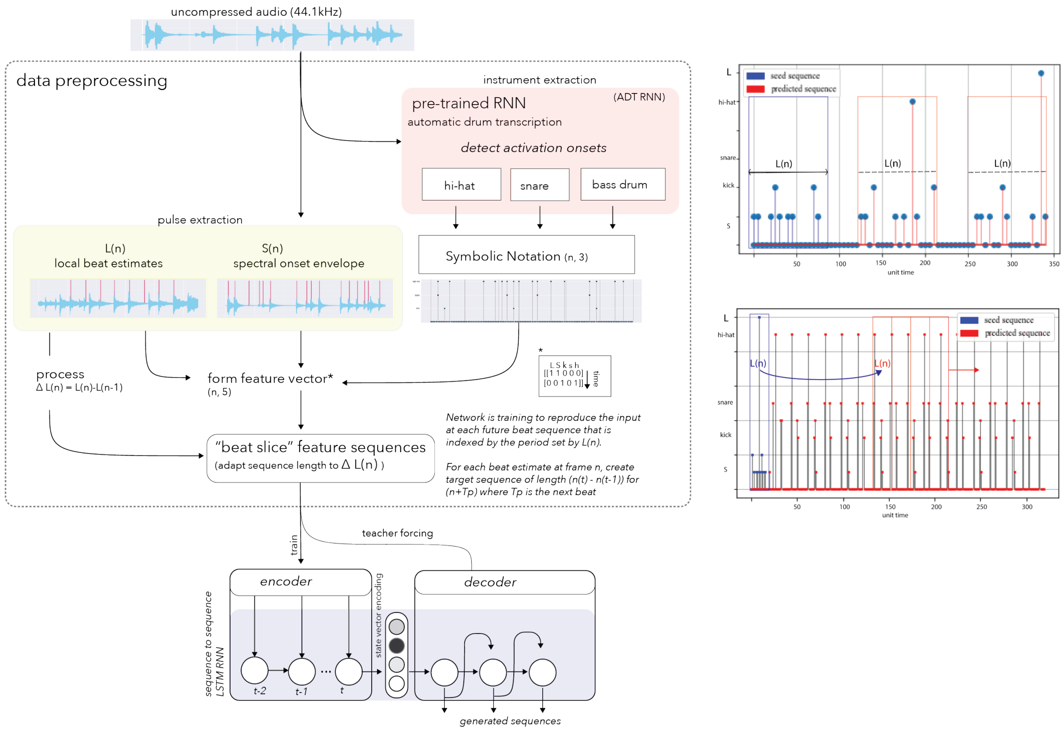


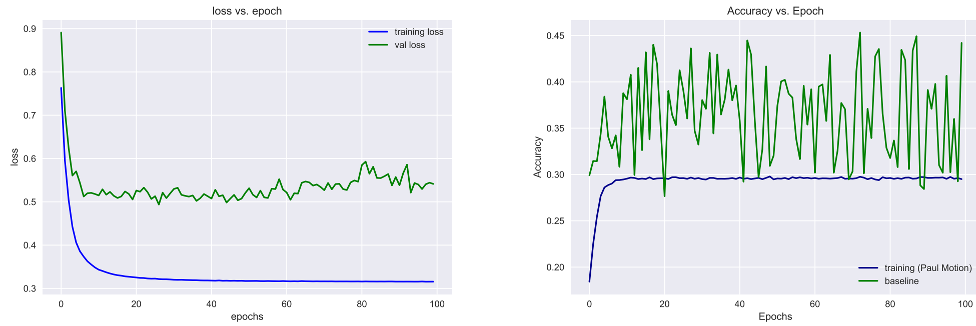
Figure 2: (left) Data Pre-processing pipeline and Beat-to-Beat encoder-decoder RNN (right) Two generated sequences using two different sequence lengths. The generated sequences retain the heirarchical periodicity of the seeded sequence

learn the to model dependencies across training examples[2]. I also trained the network on mini-batches of size 64, 128 which sped up computation but did not result in improved performance.

Observing Figure 3, in terms of loss and accuracy the model is overfitting the training data as it cannot generalize as well to the validation set. Normally this might call for performing batch normalization on the data, increasing dropout or reducing the number of hidden units. However, given that one of the goals of this project is to generate rhythmic sequences that are representative of the drummer, this is not necessarily an undesirable consequence. In fact, this overfitting has the general outcome of being a forcing function, one that constrains the output to conform to the sequence width set by the beat gradient. Observing the two example generated sequences shown in Figure 2 where the network generated output continuously (without an 'end of sentence' token), the pulse width of the seeded input sequence is clearly echoed (and repeated) in the output sequence with slight rhythmic variation between each pattern. The generated sequences also tended to conform to the general instrumental density of the seed. However, on the level of the beat onset features, the network was able to capture the local and global density of onsets depending on the intialization seed. This is shown in the two plots in Figure 2 where the lower pulses on the bottom of the horizontal axis serve as a sort of a low-level pulse 'tactus' and the more sparse pulses one feature above seem to indicate percussive event onsets.

Because this is a generative model, the conventional measures of evaluation–loss and accuracy metrics–are less useful for assessing the quality of the network output. Nevertheless, I reproduced several of the 'ground truth' sequences (outputs of the ADT transcriptions) alongside their generated sequences using MIDI drum set instruments to conduct a small qualitative survey. I had a sample size of 12 participants listen to 10 examples of the ground truth sequence and the generated sequences and determine which one was 'computer-generated'. On the survey, the participants were able to on average detect the fake sequence 60 % of the time.

[2] This might have the added benefit of informing certain compositional decisions in the drummer's performance



(a) training and validation loss over epoch

(b) accuracy over epoch

Figure 3: Model Statistics

The main limitation of this model is its reliance on the pre-processing pipeline that infers beat estimates and the resulting percussion transcriptions. The ADT transcription algorithm is simply not well-suited for high fidelity transcriptions of non-isochronous types of audio. Similarly, the beat estimates are in no way an indication of how the drummer himself is constructing time during the course of a performance. One way to retrieve more reliable local beat estimates would be to use tap responses from drummers themselves.

6 Conclusion and Future Work

The beat-to-beat RNN was shown to be capable of learning pulse sequences of the drummer's performances in the training set but could not generalize as well to other performances. The teacher forcing, when training the decoder, was shown to be an effective way to teach the network how to reproduce the sequence length and pulse density of the training set. Although the network did tend to overfit the training data, this did not necessarily detract from the aesthetic quality of its output as it was able to maintain a balance between novel rhythmic pattern generation within different levels of temporal regularity. Along with trying out different network architectures, hyperparameter tunings, and regularization methods, such as weight decay, attention mechanisms might be helpful to control for overfitting and future work on this area would be beneficial for optimizing this equilibrium. However, as a first shot attempt at using pulse and onset estimates to synthesize stylistic rhythms from non-isochronous genres of music, this network was shown to be effective in infering pulse from specific rhythmic material from a drummer's performance arsenal. In the future, it could be potentially utilized in live musical applications where it could be seeded in real-time to produce rhythmic data that reflects the the performer's adaptive performance.

References

[1] Abadi, Martín et al. "TensorFlow: A System for Large-Scale Machine Learning." OSDI(2016).

[2] Bello, J.P. Daudet, L. Abdallah, A. Duxbury, C. Davies, C. & Sandler, M. *A tutorial on onset detection in music signals*, Speech and Audio Processing, IEEE Transactions on, vol. 13, no. 5, pp. 1035–1047, (2005).

[3] Boulanger-Lewandowski, Nicolas, 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).

[4] Choi, Keunwoo, George Fazekas, and Mark Sandler. *Text-based LSTM networks for automatic music composition.* arXiv preprint arXiv:1604.05358 (2016).

[5] Chollet, F. keras, GitHub. https://github.com/fchollet/keras (2015)

[6] D.P.W. Ellis, *Beat Tracking by Dynamic Programming,* Journal of New Music Research, Vol.36(1), 51–60, (2007).

[7] Eck, Douglas, and Juergen Schmidhuber. *A first look at music composition using lstm recurrent neural networks.* Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale 103 (2002).

[8] Engel, Jesse, et al. *Neural audio synthesis of musical notes with wavenet autoencoders.* arXiv preprint arXiv:1704.01279(2017).

[9] Franklin, Judy A. *Jazz Melody Generation from Recurrent Network Learning of Several Human Melodies.* FLAIRS Conference. 2005.

[10] Graves, Alex. *Generating sequences with recurrent neural networks.* arXiv preprint arXiv:1308.0850 (2013).

[11] Hochreiter, S. Schmidhuber, J. *Long Short-Term Memory.* Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. (1997).

[12] Large, E. W., Herrera J. A. and Velasco M. J. *Neural networks for beat perception in musical rhythm. Frontiers in Systems Neuroscience*. 9 (159). doi: 10.3389/fnsys.2015.00159 (2015).

[13] Large, E.W, Almonte, F. V, & Velasco, M. J. *A canonical model for gradient frequency neural networks.* Physica D, 239. 905-911. (2010)

[14] Lerdahl, Fred, and Ray S. Jackendoff. *A generative theory of tonal music*. MIT press, 1985.

[15] Raffel, Colin, and Daniel PW Ellis. *Extracting Ground-Truth Information from MIDI Files: A MIDIfesto.* ISMIR. 2016.

[16] Roberts, Adam, et al. *A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music.* arXiv preprint arXiv:1803.05428 (2018).

[17] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. *Sequence to sequence learning with neural networks.* Advances in neural information processing systems. 2014.