Generating Coherent Drum Accompaniments with Fills and Improvisations

Rishabh Dahale[1], Vaibhav Talwadker[1], Preeti Rao[1], Prateek Verma[2]
[1]Department of Electrical Engineering, Indian Institute of Technology Bombay, India
[2]Stanford University

Motivation

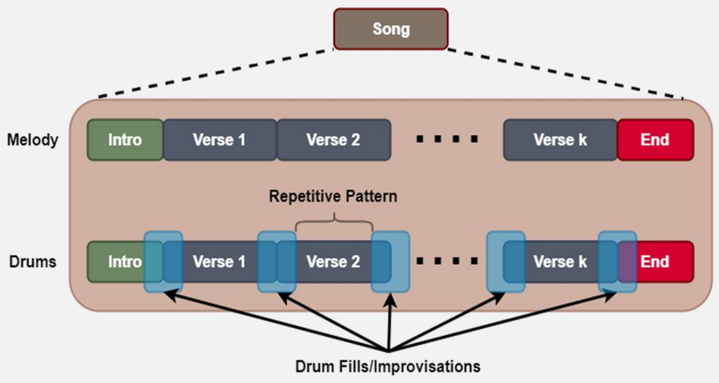


Figure 1: Temporal Structure of song

- Drums play a significant role in overall aesthetics of a song
- Fills/improvisations are short groups of notes played while transitioning from one section/verse to another
- They serve as an indicator, for audience and the band, of an upcoming transition in the song
- Duration of these fills/improvisations is typically a few beats long

Goal

Generating coherent accompanying percussion track for a given melody

- Generating a good basic drum pattern
- Identifying the locations of improvisations
- Generating fills/improvisations for these detected locations

Dataset & Preprocessing

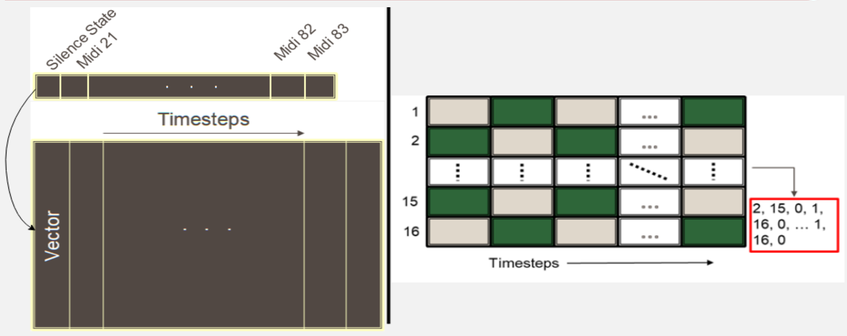
Dataset

- Lakh Pianoroll Dataset (LPD-5 Cleansed) [1]
- Consist of 5 instruments: Bass, Guitar, Strings, Piano and Drums
- All songs of 4/4 time signature i.e., 4 beats in a bar

Preprocessing

- Downsample all the beats from 24 parts per beat to 8 parts per beat
- Selecting only the active pitches for melodic instruments (MIDI 21-83)
- Combiing similar instruments in the percussion track. E.g., MIDI 38 and MIDI 40 correspond to electric and acoustic snare and combined
- Selecting only 16 percussion instruments: snare drum, open hi-hat, close hi-hat, kick drum, ride cymbal, crash cymbal, low-floor tom, high-floor tom, high tom, hi-mid tom, low tom, cowbell, pedal hi-hat, tambourine, cabasa, and maracas
- Binarize percussion track to decrease training parameters

Data Representation



(a) (b)

Figure 2: Data representation (a) Pianoroll (b) Serialized grid representation

- We use 2 different representations for our dataset
  1. Pianoroll representation: Vectors for each of the timestep are stacked together to generate a matrix like input.
  2. Serial Grid Input: Active instruments are unfolded into a sequence of tokens. This helps us take advantage of advances in the language modelling tasks
- Figure 4 shows the result of performing these operations on all possible bars
- Peaks of these curve represent the bars with fills
- These peaks were selected using a peak picking

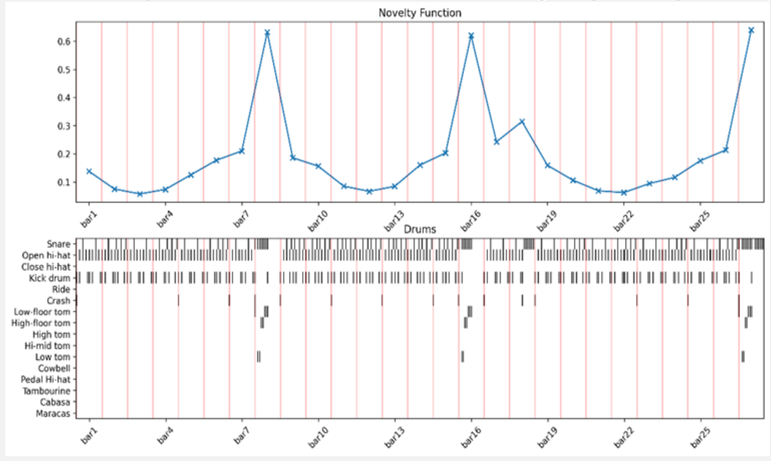


Figure 4: Novelty Function plot for a given drum track

Extracting Fills/Improvisations

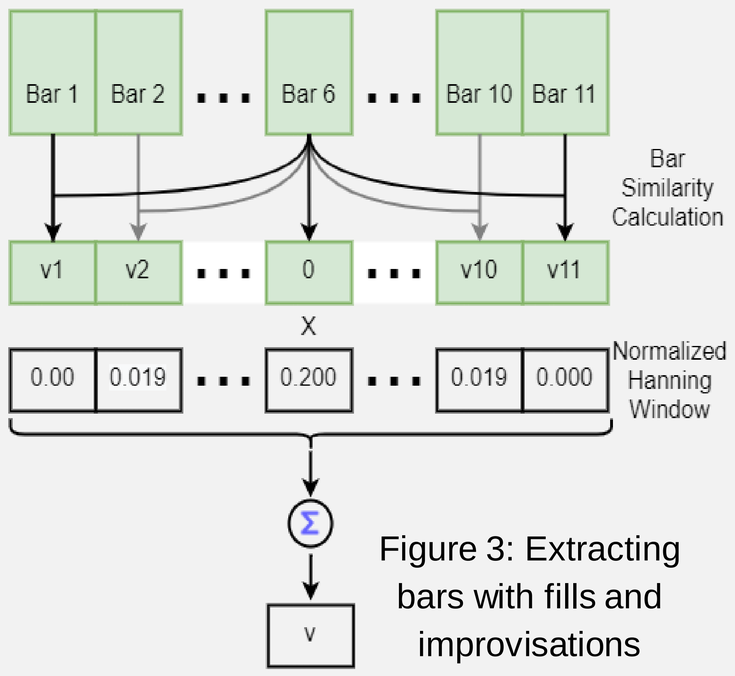


Figure 3: Extracting bars with fills and improvisations

- Bars with fills/improvisations are bars which have different pattern from their neighbours
- To extract these bars, we calculate the average dissimilarity in a bar compared to its neighbours using the formula

$$bar\ similarity = \frac{||bar_i - bar_j||_1}{||bar_i||_1 + ||bar_j||_1}$$

- A weighted mean of these values is taken, weight are calculated from hanning window parameter

Proposed System

Basic Drum Pattern Generation

- Generated basic drum pattern from melody using the sequence to sequence Transformer network
- Basic drum pattern mean the pattern without any fills and improvisations
- Train NLL: 0.108, Val NLL: 112

Improvisation Location Detection

- BERT [2] inspired model to detect locations of improvisation from melody
- Train F1 Score: 92.3, Val F1 Score: 79.2%

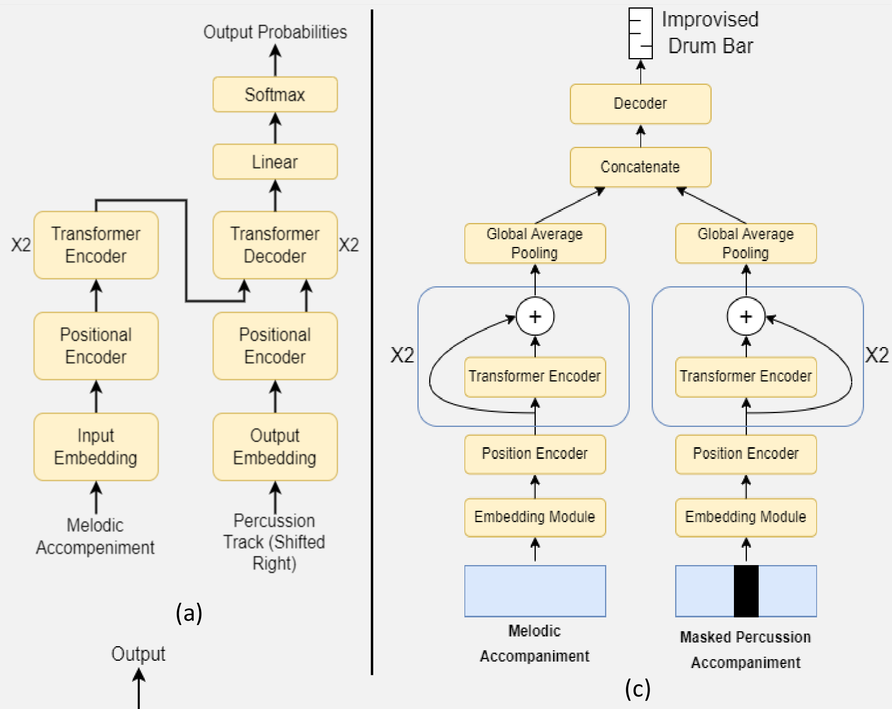


Figure 5: (a) Basic Drum Pattern Generation model (b) Improvisation Location Detection model (c) Improvisation Generation model

Improvisation Generation

- Generate fills for previously detected locations
- Generated fills are in coherence with the basic drum pattern
- Train F1 Score: 95.9%, Val F1 Score: 76.0%

Evaluations

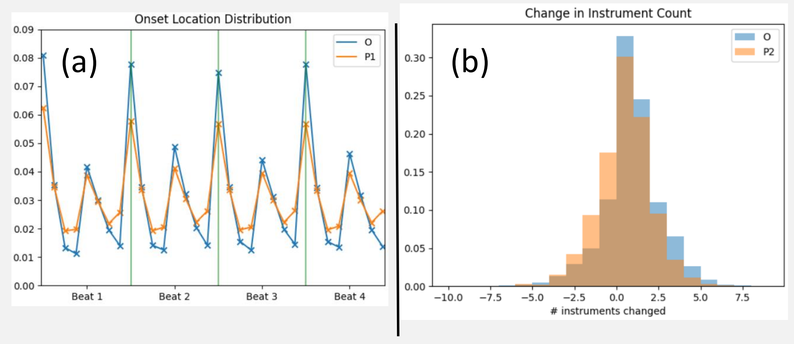


Figure 6: (a) Onset Location distribution in a bar (b) Change in instrument count

Onset Location

- As we do not provide any bar and beat boundary information, we evaluate if our model was able to learn this from the data
- From the plot we can see that our model was able to learn the locations of onsets

Instrument Count Change

- To understand if our system was able to introduce new instruments during a fill, we calcuate the change in instrument count of fill bar compared to previous bar
- When we compare this distribution with that of the dataset, we were able to have an overlapping area of 87.9%, indicating that the model was able to learn this behaviour

Conclusion

- This work highlights serious drawbacks of the traditional language-based generators
- By learning where and how to improvise, our evaluation shows improved generation quality
- With the step approach, we show that we can mitigate the biases intrinsic with the data imbalance issues

References

1. Dong, Hao-Wen, et al. "Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 32. No. 1. 2018.
2. Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).