Beatbox-to-Drum Conversion

Devansh Zurale, Jonathan Michelson
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
dzurale@andrew.cmu.edu, jmichels@andrew.cmu.edu

Abstract

MIDI drum tracking is prohibitively tedious. Musicians who desire specific realistic percussion patterns in their recordings must funnel their creative ideas through the physical restrictions of a MIDI controller. Beatboxing, contrastingly – where one imitates the different elements of a drum kit with his voice – is highly intuitive and widely used to convey or perform rhythmic ideas, suggesting its appropriateness as an input to a performance capture system. This potential is explored through creation of a preliminary translation system that employs signal processing and machine learning techniques to classify each unit of an input beatboxing sequence and to synthesize the corresponding drum pattern. The system focuses on the speaker-dependent case and is trained on three drum elements (kick, snare, hihat) to scale the problem to a tractable size. Accurate results were obtained with this arrangement, warranting future development of a robust implementation.

1. Introduction

The prevailing means by which amateur musicians with little access to, or expertise with, percussive instruments introduce rhythm into their recordings is MIDI programming. This task consists of translating conceived rhythmic patterns into MIDI data via unintuitive hand motions; MIDI controllers are typically electronic keyboards, so one must perform the desired percussive pattern on its key-shaped transducers, as one would a piano piece.

A more natural rhythm communication mode is beatboxing. By vocally imitating the various noises of a drum set, one can effectively communicate the basic structure of a rhythmic pattern with nearly as little effort as humming the melody played on an instrument. We suggest that this mode of percussive communication is highly preferable to most musicians, so we introduce and assess a preliminary automated beatbox-to-drum conversion system. Noting that MIDI conversion reduces to a programmatic hurdle if this system is reliable, we focus our efforts on the conception of this system and developing its preliminary success.

The restated technical goal of the project is then to detect when a beat in an unknown beatboxing input pattern is onset, and to classify each of those onsets in one of the three classes: kick, snare and hihat. These are then played back with substituted actual drum sounds with an aim to achieve the same pattern as was input using beatboxing. This goal can be sorted into two main issues: onset detection and classification.

2. Implementation

2.1. Training and Testing Data Collection

We first train our system on a set of thirty isolated beatboxing utterances each of the three types of drums: kick, snare and hihat. These are widely considered the fundamental elements of any modern percussive pattern. Our data is recorded at 24kHz (originally 48kHz, decimated by two), 16-bit depth, recorded in Logic Pro. We organize them into three classes for the computer and put them in appropriate folders and label them.

Our testing data consisted of eleven two-bar beatboxing patterns at a tempo of approximately 60-80 BPM, all performed by the same beatboxer. The patterns were restricted to only quarter and eighth-notes. These recordings were also obtained at 24kHz, 16-bit depths (again, originally 48kHz, decimated by a factor of 2). We must mention that these are large simplifications. Noise-free, speaker-dependent (only works on the speaker who submitted the training - in this case, one of the authors), three very spectrally-discriminable drum hits, etc.

2.2. Typical Signal Computation

We then compute a "typical" average drum sound for each of the elements. We define this typical kick as the inverse DFT of the mean of the DFTs of each class‘ training data. Rationale for this definition of the typical sound was that frequency domain averaging would be more audibly subtle due to lower phase sensitivity.

$$x_{TYP}[n] = \frac{1}{L}\sum_{k=0}^{L-1}\left(\overline{\sum_{n=0}^{L-1} x[n]e^{-jk\omega n}}\right)e^{jk\omega n}$$

Figure 1. Typical signal definition: inverse DFT of the mean of the DFTs of each training instance

2.3. Onset Detection

Once the training is done, our next step should be to classify an unknown beatbox hit. But, before we can classify a beatbox hit, we must detect when a hit is occurring, as the input to the system is going to be an entire sequence of beatbox hits. For this we need to run an onset detection algorithm over our input sequence. Plainly using the magnitude of the signal as a threshold, we may be able to detect the 1st hit, but in order to avoid false detections, we need to set a time window within which we have to bypass the magnitude-checking condition after a beat has been detected. This will put some constraints over the tempo with which you can input a signal.

So alternatively, we can take small 10ms windows and compute the total energy present in each of the successive windows. When there is a sudden energy change and also when the absolute value of the energy passes above a certain threshold – both the conditions are important – then we can say that it was a hit. Intuitively this method does seem to be pretty effective, but in practicality, the peaks of the energy plot arent́ very revealing and a little difficult to detect.

Our ultimate approach was to take 10ms windows, compute their DFTs, and plot the average energy present in each of the framed windowed signals and we see that the graph of the energy vs. the time frame produces peaks that are extremely consistent. Some post processing like smoothing is then required to get a very good looking graph, and the location of the peaks are the locations of the onsets. One important thing to note is the location we find for the onsets will be in terms of the window number, so we will have to translate that to the actual sample number which should be easy as we know the sampling frequency and the window length for the signal. The formula for obtaining the sample position of the onset given N being the Frame number is

$$M = 10Fs/1000 * N + 1$$

where Map M is the translation of N onto the original scale. Once we have the exact location of the hit, we would then take fixed number of samples starting from the point of detection and that will be passed through the classification algorithm. In our case, our training set was consistent on 8192 samples of data so we preferred taking 8192 samples of the testing data for the classification. However, 8192 samples corresponds to almost 0.4 seconds, so to take into account faster tempo, we had to only consider the first 4096 samples starting from the onset detection point of the input sequence and then pad 4096 zeros towards the end to make the hit 8192 samples long. This increased the accuracy of for faster tempos considerably.

2.4. Classification

Next, we use LPC coefficients to create a filter based on the previously computed typical sound. LPC (Linear Predictive Coding) coefficients are a set of coefficients which are commonly used in speech coding problems. What the LPC coefficients mainly do is, they predict the current sample of a signal as a weighted sum of the past p samples of the signal. The LPC coefficients are nothing but the weights that are used to define the current sample of the signal.

$$y(n) = \sum_{k=1}^{P} \alpha_k y(n-k) \qquad \alpha_k \; are \; the \; LPC \; coefficients$$

Figure 2. LPC coefficients

An alternate way to understand LPC coefficients is that when you design an all pole filter using the LPC coefficients, and you excite this filter, you get an output which is almost the same as the signal for which you computed the LPC Coefficients. For excitation of the filter you may use either white noise or a quasi periodic function, the former used for unvoiced speech generation and the latter used for voiced speech generation problems. The squared magnitude of the frequency response of the filter is a parametric estimate of the power spectral density function of the output random process. [1, 2]

$$H(z) = \frac{G}{1 - \sum_{k=1}^{P} \alpha_k z^{-k}}$$

Figure 3. All-pole filter defined by LPC coefficients

By doing this we can expect that the total energy of the output of one of the LPC filters is highest when its incoming signal was the same signal off of which the filter‘s coefficients were based. In other words, the frequency responses of the LPC filters definitionally matches those of the typical signals from which they were computed. We can thus know, by testing for maximum filter output, the identity of the unknown input. E.g., our typical kick drum passed through the kick LPC filter will yield a larger output signal energy than that of the snare or hihat LPC filters. On this basis our system can make an informed decision of the unknown hit

signal, and appropriately classify it and its location in the test pattern sequence for actual drum synthesis later.

2.5. Synthesis

Our next step was to synthesize a signal which consists of actual drum sounds at the exact locations where the beatboxing hits were detected. We start by importing .wav files of the actual drum sounds which were kick, snare, open hihat and close hihat. These were obtained from a free database online. The drum samples were made sure to be cleared off the silence at the start. These datasets were sampled at 44100kHz. Using a sampling ratio of 5:9, we resampled these signals to a sampling rate of 24.5kHz which is not very different from 24kHz. An array of zeros of the length of the original input sequence was generated. At every onset point the samples in this array were replaced with the corresponding actual drum signal samples. Since the drum signals were sampled at 24.5kHz and the playback was at 24kHz, this would ideally mean that the drum sound is being played marginally slower than the original speed, but that doesn?t audibly affect the output and was within the acceptable error range.

An entertaining addition made to the project during synthesis was that every time a pattern such as 'Not a Hihat - Hihat - Hihat' was detected, the initial hihat was assigned to be an open hihat and in the synthesis, the open hihat sound was used in that place. All the other hihats used were closed hihats. This is also a very simple example of probability based pattern recognition that could be added to the project at a later stage.

3. Results

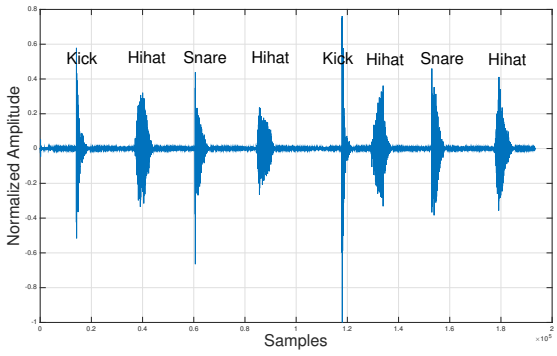

Figure 4. Test Pattern #1

Figure 4 displays the amplitude waveform of Test Pattern #1 v. sample number. The ground truth drum element labels (kick, snare, hihat) are inserted into the figure for reference, but are unknown to the system. Figure 5 illustrates the results of our onset detection scheme. One can see that easily discriminable energy peaks are achieved, aiding the process of extracting the unknown hit.

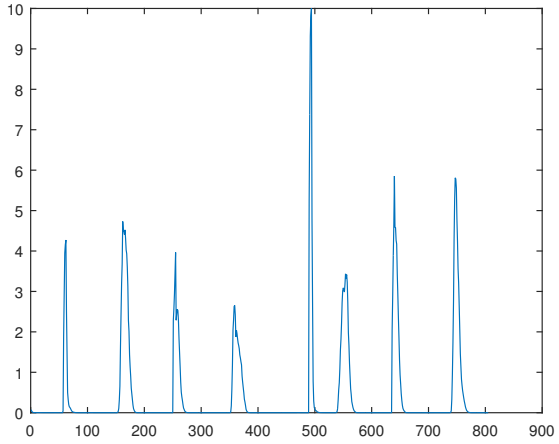

Figure 5. Normalized scaled energy vs. window number.

Figure 6 displays four amplitude waveforms. The top left is the raw detected unknown hit. The top right, bottom left, and bottom right are the outputs of the kick, snare, and hihat LPC filters with that hit as their input, respectively. We see that the energy of the kick LPC filter output is greater than that of the other two filters – which is what we expect given that the ground truth of the first hit is, indeed, a kick drum. We store the classified identity of that drum, as well as the sample number of its onset, in a map that we access during actual drum synthesis. We proceed with the remaining detected hits in the same fashion.

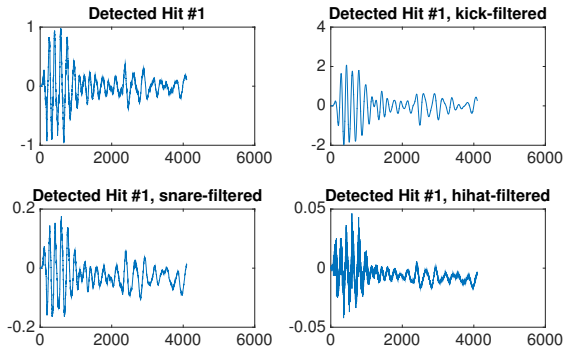

Figure 6. Unknown detected hit #1 filtered through all three LPC filters. Note that the highest energy belongs to the top right plot – the output of the kick LPC filter excited by the unknown signal.

Table 1 compares two test patternsǵround truth labels and our system?s classification labels. Accuracy is excellent.

Table 2 summarizes the accuracy metrics of our system. Note that these metrics are obtained over all eleven test pat-

Selected Classification Results			
TP #1	System	TP #2	System
K	K	H	H
H	H	S	K
S	S	H	H
H	H	K	K
K	K	K	K
H	H	H	H
S	S	S	S
H	H	S	S
		H	H

Table 1. K = kick, S = snare, H = hihat, TP = Test Pattern. Vertical progression from the top down indicates the progression of Test Patterns #1 and #2. System columns denote the corresponding classified patterns of the respective patterns. Overall classification results excellent, though system misclassified first snare in Pattern #2 as kick.

Accuracy Metrics	
Kick	100
Snare recognition	95.65%
Hihat recognition	100%
# Insertions	0
# Deletions	0
# Submissions	1
WER	0.0092

Table 2. . Accuracy metrics obtained for performance on all eleven test patterns, number of beatbox words N = 109. WER = (I+D+S)/N

terns.

4. Discussion

Several methods were tried for classification as well as onset detection. For classification, the simple initial approach of ad hoc spectrogram search for determination of center frequency and bandwidth of classification filter showed impressive results but were unreliable for larger generalized data sets. The LPC approach makes the classification filter the LPC model of our dataset to achieve generalization and scalability. This automates the determination of spectral profile of training data by the system.

For onset detection, we employed comparison of gradient of windowed energies, energy of the correlation of successive windows, etc. Computation of spectrogram of the signal and energy comparison in each time window gave the most prominent peaks and was the most reliable technique. Normalization of the signals at every stage was the key to obtaining accurate results.

5. Conclusions and Future Scope

Hence, a rudimentary speaker-dependent system which achieved a successful transcription of the input sequences with a per-drum accuracy of over 95% was developed. The system accuracy however, is severely affected in the presence of noise.

There are several immediate future additions that can be made to the project to make it a more robust and fully functioning one. To discuss a few, we can have additional system classes such as Tom, Crash, Ride, Open Hihat, etc. As opposed to the speaker dependent system that we have currently, we can make a speaker-independent system such that it works for any beatboxer. Also, since every beatboxer might have a different standard for the way he/she imitates a particular drum sound per se, our system should work good for a wide variety of sounds for one class. Also, one goal would be increasing the accuracy and reliability of both the classification and onset detection further along with improving the noise robustness of the system.

This project also has an immense potential involving extensive machine learning and pattern recognition. Examples of the far stretch future goals would be predicting overlapping drums based on probabilistic pattern recognition, correcting erroneous detection based on past input patterns, introducing velocity of the drums as a parameter, making the system real time, etc.

6. Acknowledgements

We thank Prof. Bhiksha Raj for instructing this fascinating course. Many thanks to Teaching Assistants Zhiding Yu and Bing Liu as well. Prof. Richard Stern was instrumental in guiding this project.

References

[1] Rabiner, L. R., and Schafer, R. W. *Digital Processing Of Speech Signals.* Englewood Cliffs, N.J.: Prentice-Hall, 1978. Print.

[2] Stern, Richard M. *Lecture notes of 18-792 (ADSP)* 2015