Musical Transcription of Drum Patterns Using Main Audio Features as Feature Vector in KNN

Charles Jayson L. Dadios and John Emmanuel I. Encinas

I. Introduction

Music has the power motivate, stir emotions, and even revive a persons memory [1], and in contemporary setting, it is usually accompanied by a percussion instrument which is mainly the drums. The drums is a rhythmic percussion instrument that is composed mainly of seven main instruments, namely bass drum (kick), snare, tom-toms, floor tom, hi-hat, crash, and ride cymbal. In a band setting, it allows listeners to follow the beat of a composition and helps establish its dynamics and overall feel. It is also known as the backbone of the band along with an electric bass.

Automatic Drum Transcription (ADT) in simple terms is the process of converting a drum performance into a record usually as a drum notation printed as a music sheet.

A. Statement of the Problem

Most beginner drummers, and even those who are drummers by profession, may face difficulty in learning how to play a song, because their method is to learn it through familiarization and memorization. This has been observed by especially in folk, rock, and pop musicians, aside from the fact they are informally trained, and that music is difficult to memorize [2]. Another method that is common for a formally trained drummer would be to read a drum transcription of a song. Unlike a chart with lyrics and chords for chordal instruments, most popular songs do not have a drum transcription, and it would be difficult and time-consuming for themselves. Although it is relatively easy, with training and practice, to learn and read a basic drum sheet pattern, writing one would require a challenging mental operation. Although ADT is more conveniently done using a musical instrument digital interface (MIDI) controller, using one may not always be practical.

B. Significance of the Study

Since there are very few songs that have written drum patterns for another player to learn, this study suggests a solution to automate the audio to musical transcription.

C. Objectives

The general objective of this proposal is to develop an application which can produce a musical transcription of drum beat patterns from digital drum audio samples.

Specific:

1) To produce physiologically realistic drum sound combinations using downloaded drum instrument audio samples
2) To extract main audio features from the data set
3) To classify the sounding instruments at given time intervals
4) To evaluate the accuracy of the produced transcription

Presented to the Faculty of the Institute of Computer Science, University of the Philippines Los Baños in partial fulfillment of the requirements for the Degree of Bachelor of Science in Computer Science

II. Related Literature

Studies on music transcription have mostly been interested on the melodic instruments, but there have been some which focused on classification of audio signals for ADT, usually using the the audio features. [3]

The common procedure in most implementations of automatic music transcription (AMT) and audio is deriving features [12]. This is also used in ADT, only the implementations differ.

Derry FitzGerald, identifies ADT as a sound source separation problem. The approach of using simple heuristics was able to identify individual drum parts without any form of rhythmic modelling. He used the general source separation and redundancy reduction techniques, such as Independent Component Analysis and Independent Subspace Analysis. There were only a few issues of misidentifying a low tom as a kick drum, and a loud hi-hat being detected as a drum hit [5].

Carl Southall, Ryan Stables and Jason Hockman. Used Convolutional Neural Network (CNN) in their implementation of ADT. They used spectral features to produce a segmented spectrogram to be the training data for CNN. Their system was trained to identify only the kick drum, snare drum, and hi-hat sounds. They evaluated the system with a drum solo, drum mixture (with drum set parts other than kick, snare, and hi-hat), and with multi-instrument mixture for the purpose of testing their systems ability to adapt from unseen timbres [6].

The study of Marius Miron, Matthew E.P. Davies, and Fabien Gouyon [13] also used feature extraction for the feature vectors and the K-Nearest Neighbor algorithm (KNN) was used for classification. But the algorithm was intended for live input and only short samples. They were able to produce an almost 90% accuracy in recognizing the sounded instruments.

In the study of Olivier Gillet and Gal Richard, they performed an ADT from an audio-video file by extracting its audio features and visual features.They compared the recognition rate when using only the audio features, only the video features, joint feature vectors, best among the two, and fusion. They found out the using the joint feature vectors [7].

The methodology of an undergraduate study [14] will be adopted because of their same case of using KNN to classify heartbeat using low-level features.

III. Methodology

A. Data Gathering

There are 7 different instruments present in a drum kit, namely bass, snare, tom-tom, floor tom, hi-hat, crash, and ride, where three more sounds may be produced, namely open hihat, stick, and ride bell, for a total 10 different sounds. Samples of each will be downloaded from several websites but mostly from [19], providing a compilation of initially labeled samples individual of each class in wav format of no more than 3 seconds.

The 10 sounds produced from the 7 pieces of the instrument will become the classes representing the sounds triggered by a single hit. These will be combined to produce a total of 142 classes, where all the data set will come. About 500 representatives will be chosen from each class. The single class samples were duplicated up to a total of about 500. This was done to balance the weight of samples for KNN. Other derived classes' samples can grow exponentially.

The the classified data that to be used will come from the websites providing compilation of labeled pre-recorded individual drum kit instruments in wav format, no more than 10 seconds. These samples will have been classified into 10: bass, snare, stick, tom-tom, floor tom, hihat, hihat-open, crash, ride, and bell. These classes will be combined using a Python script to be included in the program to produce all the said necessary classes.

The classes are combined in reference to a drummer's physiology as seen in table 1.

Instrument count per beat	Derivation(count)	Subtotal
1	kit pieces (7) + stick, hhopen and bell (3)	10
2	two sticks (36) + one stick and bass (9)	45
3	two sticks and bass (48) + two sticks and foot hihat (20)	68
4	Two sticks, foot hihat and bass (19)	19
total classes		142

Table 1: Breakdown of the derived classes

10% from each class will be used as the test set for evaluating the KNN accuracy.

B. Feature Extraction

Feature extraction is the process of creating a new set of k-dimenstional features from the original data [4]. It is done to reduce the computational complexity of signal processing algorithms [9]. Three feature sets will be used for classification, and these features will be extracted using LibROSA [17], a Python package for music and audio analysis. Using librosa, the extracted features will be returned as a Numpy array, wherein each element will represent the computed feature. The feature set is composed of low-level audio features which are readily computed abstractions that LibROSA provide for processing raw audio inputs. The following features will be used:

1) Spectral bandwidth describes the density of the signal. It can be computed by calling the function librosa.feature.spectral_bandwidth(y), where y is the audio time series array.
2) Spectral contrast describes the difference of the high and low points of the audio signals spectrum, computed as librosa.feature.spectral_contrast(y).
3) Zero crossing rate describes how often the audio signal crossed the 0 voltage axis, computed as librosa.feature.zero_crossing_rate(y).
4) The Mel-Frequency Cepstral Coefficient which represents the shape of the spectrum, as how a human auditory perception would react with very few coefficients, computed as librosa.feature.mfcc(y)
5) The Spectral centroid which describes the central mass of the input signal, computed by calling the function librosa.feature.spectral_bandwidth(y).
6) Spectral flatness is the quantification of a sound from being noise-like versus being tone-like, computed as librosa.feature.spectral_flatness(y)
7) Spectral roll-off describes the distribution of the power along the frequency spectrum, and is computed as librosa.feature.spectral_rolloff(y)

C. Classification

A K-NN algorithm, written using Python3 packages from Scikit-learn [18], will be performed to classify the partitioned validation set to evaluate its accuracy in classifying an individual data. All the audio features in the form of Numpy, an array of the computed feature for each frame. These audio features will have their mean as their single numerical representative. So each audio feature will be a value for a feature set's feature vector.

KNN decreases its classification performance as it increases feature dimension, so the data set size must be increased to compensate [16]. Thus, the large sample size of about 500 for each class was chosen.

The algorithm will be utilized to perform the classification for longer audio streams of drum beat patterns. The audio input will be segmented by the length of its beats, using a Librosa beat tracker library to monitor the duration of notes. Librosa also has a library for tracking onsets. Every onset will mark a note. Each note will then be fed into the classifier, for every given beat interval. The output will be a MIDI format file using the Python MIDIUtil library, and a PDF format using Lilypond [20].

D. Evaluation

The algorithm used for classifying the sounds will be evaluated for its accuracy in correctly identifying the sounding

classes. The Scikit-learn has a built-in library for evaluating KNN accuracy.

Also, the final algorithm will be evaluated using three types of input.

1) A briefly composed midi drums will be played through LMMS (digital audio workstation) and be imported as a wav audio file and will be an input for the application to produce another midi file.
2) A digital audio recording of a drum set being played, accurately playing what was indicated on a drum sheet will be fed through the classifier. The output notation will be evaluated by its correctness per beat with the original notation.

A confusion matrix will be used to visualize and evaluate the accuracy of the produced transcription. The row and column length will be equal to count each underivable drum kit sound, instead of cardinality of the classes.

IV. RESULTS AND DISCUSSIONS

The algorithm was successful in combining basic drums audio samples to produce more sample classes which are physiologically realistic. Main audio features were also extracted and used for classifying the samples. The input drum recordings were transcribed with great accuracy, among the 142 classes or drum hit combinations.

A. K-Nearest Neighbor

Using the built-in method of Scikit-learn's KNeighborsClassifer module, the evaluated accuracy was high, but varied depending on the value of k and the number of classes. The classes to be considered are adjusted by whitelisting or blacklisting a sound or removing specific combinations. The varied average accuracy over 5 trials can be seen in table 2.

Classes	k=1	k=3	k=5	k=7	k=9
Simple	98.29%	95.44%	93.15%	91.41%	91.68%
Common	97.03%	93.59%	90.66%	88.26%	86.54%
All	87.41%	75.40%	68.35%	63.91%	60.82%

Table 2: KNN accuracy in varying k value and number of classes

The highest accuracy was achieved when k=1. Furthermore, the accuracy improves as the number of classes considered were reduced. The simple class pertains to the sound classes bass, snare and hihat which are the most commonly heard in pop songs. The common class pertain to most parts of the drum kit, except that the least common sound combinations were eliminated. The class all are physiologically possible drum sound combinations.

B. Transcription

The audio input tempo and beat can be tracked using the Librosa package for common time signatures, but it is not always successful when given polyrhythmic beats. The detected tempo is sometimes double of the original, thus, the correct tempo was made as an optional argument to the program. Beat onsets or hits are also detected, but misses a few when not loud enough. Another concert for transcription was finding the downbeat, or the time where to count "1" in the song, as there are no solutions yet to this problem. Thus for this work, the first onset in the input recording should be the downbeat, and the time signature should be 4/4, although is already common in playing drums.

Two live and one digital drum tracks were used as input to test the performance of the transcription algorithm. The table 3 shows results of the former which use bass, snare and hihat patterns; while table 4 shows the result of the later which uses rhythm patterns of bass, snare, hihat, open-hihat, tom, floor, crash, and ride.

Actual vs prediction

Class	bass	snare	hihat
bass	8	0	5
snare	0	4	0
hihat	2	4	4
Recall	80%	38%	88.71%
Precision	68.57%	95%	63.22%

Ovarall accuracy 69.1%

Table 3: Confusion matrix of bass, snare and hihat transcriptions

Actual vs prediction

Class	bs	sn	hh	hho	tm	fl	cr	rd
bass	22	7	5	1	2	3	2	1
snare	5	15	1	0	0	1	1	0
hihat	0	7	18	1	0	0	0	0
hhopn	0	0	0	7	0	0	0	1
tom	2	0	0	0	0	1	0	1
floor	2	0	0	0	2	0	0	0
crash	0	0	0	1	0	0	3	0
ride	0	1	16	0	0	0	2	2
Recall	51%	65%	69%	88%	0%	0%	75%	10%
Precisn	71%	50%	45%	70%	0%	0%	38%	40%

Ovarall accuracy 50.38%

Table 4: Confusion matrix for commonly used classes

Compared with the evaluation of the initial KNN test set, the actual performance when using the input recordings was less accurate. Still, the application results were good since most of the precision and recall levels were above 50%. Two main factors were considered to affect the classifier performance. One is that the KNN dataset were based sampled from a controlled environment, usually to minimize ambient noise. This is most evident for recordings with fast tempo. Whereas, the recorded inputs for transcription has more reverberation, thus affecting the succeeding onsets on another class. Also, the input being a continuous stream, the features of the currently evaluated class is affected by the previous class, whose features leak to the current. In addition, the segmentation of the input into individual onsets were not as exact as the controlled samples. The onsets of controlled samples often start its signal near 0.0 time of the audio segment, whereas the input segmented by the algorithm vary in onset positions relative to 0.0 time of the segment. The second is that drum set sounds have similar features. Depending on how the audio dynamics, some features are absorbed by other classes, and thus missed.

V. CONCLUSION

Using main audio features in KNN is effective in Automatic Drum Transcription (ADT), with the highest overall accuracy achieved when k=1, even with the feature vector dimension of 77, in this case.

VI. RECOMMENDATION

Just like speech recognition, ADT may improve with the help of other machine learning techniques like Hidden Markov Model.

REFERENCES

[1] E. Mannes, "The Power of Music: Pioneering Discoveries in the New Science of Song"
[2] Ginsborg et al., "Musical Excellence: Strategies and Techniques to Enhance Performance," p. 123, 2004.
[3] C.-W. Wu, "A Review of Automatic Drum Transcription," 2017.
[4] E. Alpaydin, "Introduction to Machine Learning Second Edition," 2010.
[5] D. FitzGerald, "Automatic Drum Transcription and Source Separation," 2004.
[6] C. Southall, R. Stables, and J. Hockman, "Automatic Drum Transcription For Polyphonic Recordings Using Soft Attention Mechanisms And Convolutional Neural Networks," 2017.
[7] O. Gillet, and G. Richard, "Automatic Transcription Of Drum Sequences Using Audiovisual Features," 2005.
[8] M. Sokolova, and G. Lapalme, "A systematic analysis of performance measures for classification tasks," 2009.
[9] J. Breebart, and M. F. McKinney, "Features for Audio Classification," 2004.
[10] C.-F. Liao, "Understanding the CMU Sphinx Speech Recognition System," 2009
[11] M. Miron, M. E.P. Davies, and F. Gouyon, "An Open-source Drum Transcription System For Pure Data And Max MSP," 2013.
[12] McKinney, Martin, and Jeroen Breebaart. "Features for audio and music classification." (2003).
[13] Miron, Marius, Matthew EP Davies, and Fabien Gouyon. "An open-source drum transcription system for pure data and max msp." 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2013.
[14] C. C. L. Cepe, and M. A. A. Clario, "Heartbeat Classification using Low-Level MFCC Features with Naive Bayes and KNN," 2018.
[15] Kekre, H. B., and Kavita Patil. "Standard deviation of mean and variance of rows and columns of images for CBIR." International Journal of Computer, Information, and Systems Science, and Engineering 3.1 (2009).
[16] Shetty, Badreesh, and Badreesh Shetty. Curse of Dimensionality. Towards Data Science, Towards Data Science, 15 Jan. 2019, towardsdatascience.com/curse-of-dimensionality-2092410f3d27.
[17] B. McFee et al., "librosa: Audio and Music Signal Analysis in Python," 2015.
[18] Pedregosa, F., et al, "Scikit-learn: Machine Learning in Python," 2011.
[19] https://freewavesamples.com/
[20] lilypond.org

APPENDIX I
ENUMERATION OF CLASSES

The following figures would enumerate classes that are appropriate to the physiology of a drummer.

Instrument pieces (7) and available sounds (3)

1) bass
2) snare
3) tom
4) floor
5) hihat
6) crash
7) ride
8) stick
9) hihat-open
10) bell

One stick and bass (9)

1) bass snare
2) bass stick
3) bass tom
4) bass floor
5) bass hihat
6) bass hihat-open
7) bass crash
8) bass ride
9) bass bell

Two sticks (36)

1) snare stick
2) snare tom
3) snare floor
4) snare hihat
5) snare hihat-open
6) snare crash
7) snare ride
8) snare bell
9) stick tom
10) stick floor
11) stick hihat
12) stick hihat-open
13) stick crash
14) stick ride
15) stick bell
16) tom floor
17) tom hihat
18) tom hihat-open
19) tom crash
20) tom ride
21) tom bell
22) floor hihat
23) floor hihat-open
24) floor crash
25) floor ride
26) floor bell
27) hihat crash
28) hihat ride
29) hihat bell
30) hihat-open
31) hihat-open crash
32) hihat-open ride
33) hihat-open bell
34) crash ride
35) crash bell
36) ride bell

Two sticks and bass (48)

1) bass snare tom
2) bass snare floor
3) bass snare hihat
4) bass snare hihat-open
5) bass snare crash
6) bass snare ride

7) bass snare bell
8) bass stick hihat
9) bass stick hihat-open
10) bass stick tom
11) bass stick floor
12) bass stick hihat
13) bass stick hihat-open
14) bass stick crash
15) bass stick ride
16) bass stick bell
17) bass tom hihat
18) bass tom hihat-open
19) bass tom floor
20) bass tom hihat
21) bass tom hihat-open
22) bass tom crash
23) bass tom ride
24) bass tom bell
25) bass tom bell hihat
26) bass floor hihat
27) bass floor hihat-open
28) bass floor hihat
29) bass floor hihat-open
30) bass floor crash
31) bass floor ride
32) bass floor bell
33) bass hihat hihat-open
34) bass hihat crash
35) bass hihat ride
36) bass hihat bell
37) bass hihat-open crash
38) bass hihat-open ride
39) bass hihat-open bell
40) bass crash hihat
41) bass crash hihat-open
42) bass crash ride
43) bass crash bell
44) bass ride hihat
45) bass ride hihat-open
46) bass ride bell
47) bass bell hihat
48) bass bell hihat-open

Two sticks and foot hihat (20)

1) snare stick hihat
2) snare tom hihat
3) snare floor hihat
4) snare crash hihat
5) snare ride hihat
6) snare bell hihat
7) stick tom hihat
8) stick floor hihat
9) stick crash hihat
10) stick ride hihat
11) stick bell hihat
12) tom floor hihat
13) tom crash hihat
14) tom ride hihat
15) tom bell hihat
16) floor crash hihat
17) floor ride hihat
18) floor bell hihat
19) crash ride hihat
20) crash bell hihat
21) ride bell hihat

Two sticks, foot hihat and bass (19)

1) bass snare tom hihat
2) bass snare floor hihat
3) bass snare crash hihat
4) bass snare ride hihat
5) bass snare bell hihat
6) bass stick tom hihat
7) bass stick floor hihat
8) bass stick crash hihat
9) bass stick ride hihat
10) bass stick bell hihat
11) bass tom floor hihat
12) bass tom crash hihat
13) bass tom ride hihat
14) bass floor crash hihat
15) bass floor ride hihat
16) bass floor bell hihat
17) bass crash ride hihat
18) bass crash bell hihat
19) bass ride bell hihat

APPENDIX II
IMAGES

This is a sample of a musical drum transcription sheet

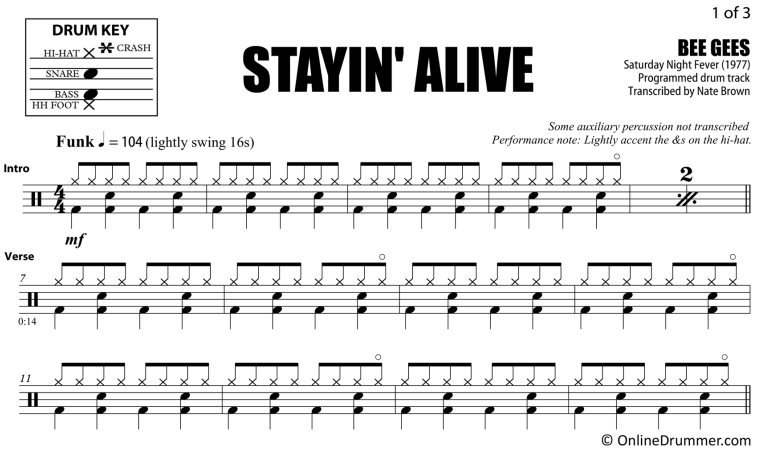

Fig. 1. Stayin' Alive drum transcription sample

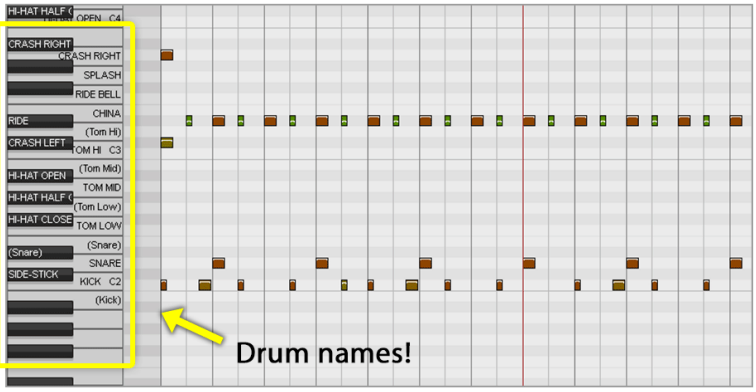

Fig. 2. MIDI drum track in a DAW.