A data-driven approach for Carnatic percussion music generation

Konstantinos Trochidis[1], Carlos Guedes[1] and Akshay Anantapadmanabhan[2]

[1] New York University Abu Dhabi, Abu Dhabi, UAE
[2] Independent Musician, Chennai, India
kt70@nyu.edu, carlos.guedes@nyu.edu, akshaylaya@gmail.com

Abstract. In this paper, we present a data-driven approach for automatically generating South Indian style rhythmic patterns. The method uses a set of annotated Carnatic percussion performances to generate new rhythmic patterns. All excerpts were manually annotated with beats, downbeats, and stroke registers. To model the rhythmic structure and the generation process of the talas, we use different partition templates that form the durations of the talas. We employed a modified version of the mutual nearest neighbor grouping algorithm to segment the rhythm sequences into meaningful grouping patterns that takes into consideration the proximity and the distance between each stroke inter-onset-interval (IOI) and their adjacent strokes. Finally, we use the *K*-means clustering approach to cluster the rhythmic groupings in terms of similarity.

Keywords: Carnatic music, music generation, rhythmic patterns.

1 Introduction

There is an increasing interest in developing computational strategies for the analysis and understanding of non-western music. Work by [1], [2], [3] and [4] in non-western music constitute some of the earlier examples in this area. Our work tries to develop generative models of Carnatic music percussion using a data-driven approach that departs from earlier work such as the one just mentioned. The goal of this paper is to develop expert systems that can reliably generate music in this style of Indian Classical music, envisioning a contribution on two levels: 1) the creation of tools for lay audiences to interact with musical styles beyond the Western ones; and 2) the automatic generation of unlimited amounts of data for training machine learning algorithms. By building applications that can recreate these musical styles we hope to create innovative tools for interaction with musical heritage that go beyond passively listening to the music. Generative music systems, video games, and virtual worlds are increasingly regarded as powerful tools for music education and performance [5]. We intend to continue developing musical applications that will allow their users to produce non-western rhythms through interaction with generative music algorithms. Using these generative systems to train machine learning algorithms would constitute a major contribution towards the creation of more robust computational systems for the analysis of the region's musical styles.

1.1 Rhythmic structure in Carnatic Music

The rhythmic framework of Carnatic music is based on the tala, which provides a structure for repetition, grouping and improvisation. The tala consists of a fixed time length cycle called avartana, which is also called the tala cycle. The avartana is divided into equidistant basic time units called aksaras, and the first aksara of each avartana is called the sama [6]. Two primary percussion accompaniments in Carnatic music are the Mridangam and Kanjira. All training excerpts used in the proposed generation method were performed on the Mridangam and Kanjira drum in the context of separate solo improvisations.

2 Approach

The approach we adopt in this study is to model the aditala cycle as a series of strokes forming a partition. Each partition is formed using different durations of groupings and sequences of strokes. In our study we used 6 templates of partitions of groups of pulses (fig.1), all adding to 32 pulses. The templates of partitions have been validated in terms of the grammar and theory of this music idiom by direct discussion with Carnatic music expert musicians. Given an audio recording, first we obtain an automatic transcription of a sequence of time-aligned events of all stroke types, their durations (IOIs) and velocities. All the recordings are merged into a text corpus of sequences of strokes. We used a grouping algorithm based on the mutual nearest neighbor that takes into consideration the proximity and the distance between each stroke IOI and their adjacent strokes to group them in meaningful rhythmic patterns. Next, all the patterns are indexed in terms of their duration and those patterns with durations between 2-8 secs are kept and used to form the partition variations of the tala. We transform all textual representation of the groupings into vector feature representations by using the bags of words approach and all grouping patterns are clustered based on similarity using the K-means algorithm.

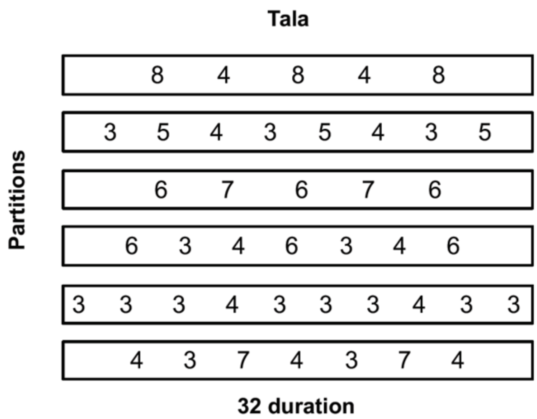

Fig. 1. Partition templates of aditala cycles.

2.1 Dataset

The training corpus consisted of 23 percussion solo compositions and groove patterns in aditala (8 beat-cycle) in three different tempo levels: slow (70bpm), moderate (85bpm) and fast (105bpm). The duration of each composition is around 2.5 minutes. The compositions were performed by professional Carnatic percussionist Akshay Anantapadmanabhan with the Mridangam and Kanjira drum. All excerpts were recorded using a metronome and were manually annotated including the sama and the other beats comprising the tala.

2.2 Encoding the strokes

In the Mridangam dataset each stroke event was encoded as a string based on five registers (Lo/Mid1-2-3/Hi), the hand (left (L) or right (R)) used for initiating the stroke, the inter-onset interval (IOI) between strokes and a value (V) indicating the velocity of the stroke. In the Kanjira drum data we used three register values (Lo/Mid/Hi), the inter-onset interval (IOI) between strokes and the velocity (V) of the stroke. For the Mridangam strokes, we also coded composite strokes played simultaneously with left and right hands. Although the Mridangam and Kanjira have a richer variety of registers and strokes, the reduction to three registers for the Kanjira and five for the Mridangam was a step to compromise the different stroke definition. This reduction was validated by Anantapadmanabhan as a process to accurately encode the different strokes in both percussion instruments. The normalized velocity values of the strokes were obtained by computing an onset detection function by combining energy and phase information in the complex frequency domain, and estimating its amplitude level with a value between 0.2 and 1 according to the strength of the stroke. For example, LoLV2T4, indicates a stroke in the Low register (Lo) using the left hand (L) on the mridangam with velocity 0.5 (V2) and duration of a dotted quarter note (T4).

3 Grouping of strokes

The encoded strokes are parsed using a grouping algorithm that groups them into meaningful rhythmic patterns according to the 6 partition templates shown in figure 2. We used a modified version of the mutual nearest neighbor algorithm [7]. It works based on the proximity of the strokes by measuring the distance between adjacent strokes. Strokes are grouped together if they are nearest neighbors to each other.

A constraint of the algorithm is that every grouping has a minimum number of 2 strokes. We adopted this constraint to avoid very small groupings of individual strokes. The algorithm stops parsing and form new groupings using a threshold that represents the largest duration that a grouping pattern can take. Figure 2 illustrates an example of grouping. The algorithm starts grouping the first two strokes in the rhythmic sequence based on the initial constraint and then compares the distances D1 and D2 between the last stroke in the grouping (blue dot) with the the next two strokes (green and red dots). D2 is less than D1 so the algorithm finds a boundary, saves the current grouping, and creates a new one. The algorithm is iterative and works hierarchically, e.g. when it finishes parsing the strokes and comparing the distances we get two layers of groupings four groupings for the first and two groupings for the second layer.

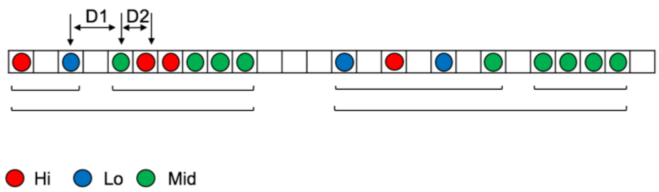

Fig. 2. Grouping of strokes based on the mutual nearest algorithm where brackets indicate groupings of strokes

4 Clustering analysis

4.1 Feature representation

All the groupings of strokes were represented initially as symbolic notation, i.e. a string of text. To transform the textual symbolic information into meaningful feature vectors we used the bag of words approach and extracted all bigrams of the groupings. We generated feature vectors of the groupings by counting the times each bigram two stroke events occur in a grouping. This leaded to a feature matrix, which could be further used for clustering analysis. Finally, the K-means clustering approach was used to cluster the groupings of strokes in terms of similarity.

4.2 Visualizing data using t-SNE

The next step was to convert the high-dimensional data set representing the center of the clusters from the clustering analysis into a matrix of pairwise similarities to enable the visualization of the resulting data. Traditional dimensionality reduction techniques such as Principal Components Analysis are linear techniques that focus on keeping the low-dimensional representations of dissimilar data points far apart.

In our analysis, we used the so called t-distributed stochastic neighbor embedding (t-SNE), for visualizing the resulting similarity data [8]. Compared to methods discussed previously, t-SNE is capable of capturing much of the local structure of the high-dimensional data, while also revealing global structure such as the presence of clusters at several scales. Figure 3 illustrates the result of the t-SNE transformation on the clusters of the groupings. A 2-dimensional axis compromising two components were used for the t-SNE analysis and the pairwise distances between the cluster centers of the groupings are plotted in the axis.

5 Generation process

To synthesize and generate the talas, we modelled the 8-beat aditala cycle into a series of partitions of 32 timepoints/cycle, assuming a beat subdivision in 4 parts, and used the partition templates shown in Figure 1. By analyzing and clustering the different groupings of strokes based on their duration and similarity we were able to have an index of different grouping durations for different clusters. In order to fill the duration

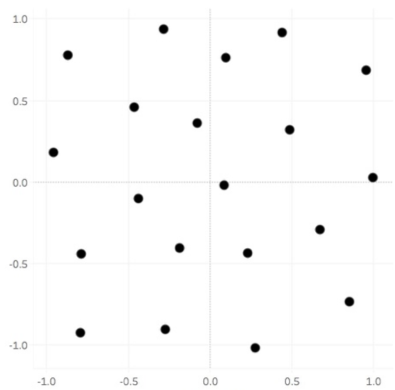

Fig. 3. 2D map of proximity distances between the pattern clusters of groupings

of the tala using a template of partition we used similar groupings in a cluster of different durations given by the partition templates. By clustering the patterns based on similarity we could generate different variations of a specific partition template of a tala using different groupings of the same duration.

5.1 Carnatic music generation application

The results from the analysis were used to develop a generative model that creates rhythmic grooves based on the groupings of strokes. The model was implemented as a Max patch that used as inputs the partition templates, the clusters of the groupings, the durations of the groupings and the coordinates of the cluster centers after the t-SNE data visualization analysis. This tool not only synthesizes the results from the analyses but it can be also used as a computational application for creative and learning exploration of these rhythms. This latter aspect is of particular interest as it provides the gateway to develop software applications for automatic rhythm generation in non-western music styles. Figure 4 depicts a screenshot of the Max patch. The user can interact with the clusters of groupings by travelling in the 2D space and generate talas of preference based on a set of template partitions in various tempo of choice. He can also filter smaller rhythmic values, or create variations by having the program probabilistically choose between different stroke collections of the same duration in the cluster.

6 Discussion and Future work

This work presents a method for automatically generating new Carnatic style rhythmic patterns based on a set of training examples. The approach we adopt in this study is to model the aditala cycle using a series of partition templates. Each partition is formed using different durations of groupings and sequences of strokes. To improve the current methodology of rhythmic grouping we aim to adopt a new approach based on a dictionary method of pre-recorded Carnatic phrases.

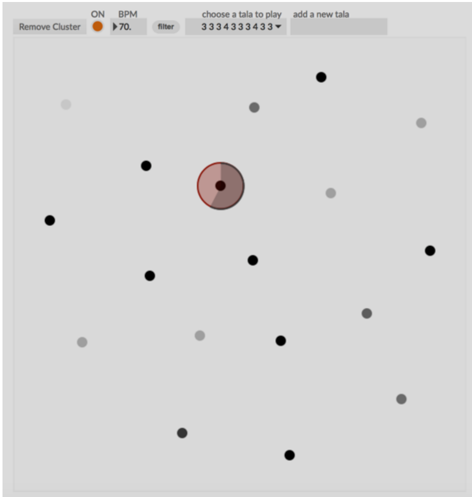

Fig. 4. Max patch of Carnatic music generation application

This method will use a dataset of well-formed Carnatic grouping dictionary of phrases performed with different variations and durations. These phrases will be later used as groupings to form the duration of the partition templates and generate the talas. Future work will also test the method on a larger dataset of recordings and evaluate the effectiveness of the method by conducting a perceptual study using a group of professional Carnatic musicians in Chennai.

References

1. Serra, X.: A Multicultural Approach in Music Information Research. In Proceedings of the 5th International Society for Music Information Retrieval Conference (ISMIR), Miami, pp. 151–156, 2011
2. Srinivasamurthy, A., Holzapfel, A. & Serra, X.: In search of automatic rhythm analysis methods for turkish and indian art music. Journal of New Music Research, 43:94–114, 2013
3. Bozkurt, B., Ayangil, R., & Holzapfel, A.: Computational analysis of Turkish makam music: Review of state of-the-art and challenges. Journal of New Music Research, 43(1), 3–23, 2014
4. Herremans, D., Weisser, S., Sörensen, K., Conklin, D.: Generating structured music for bagana using quality metrics based on Markov models. Expert Systems With Applications. 42(21):424–7435, 2015
5. Tobias, E.: Let's play! Learning music through video games and virtual worlds. In G. McPherson (Ed.), Oxford handbook of music education, Volume 2. (pp. 531-548). New York: Oxford UP, 2012
6. Sambamoorthy, P.: South Indian Music Vol. I-VI, The Indian Music Publishing House, 1998
7. Tousaint, G.: Measuring the Perceptual Similarity of Middle-Eastern Rhythms: A Cross-Cultural Empirical Study. In: Proceedings of the Fourth International Conference on Analytical Approaches to World Music. New York, 2016
8. Van der Maaten, L.J.P., Hinton, G.E.: Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research, 9(Nov):2579-2605, 2008