CAMeL: Carnatic Percussion Music Generation Using N-Gram Models

Konstantinos Trochidis
New York University
Abu Dhabi
kt70@nyu.edu

Carlos Guedes
New York University
Abu Dhabi
carlos.guedes@nyu.edu

Akshay Anantapadmanabhan
Independent
Musician
akshaylaya@gmail.com

Andrija Klaric
New York University
Abu Dhabi
ak4867@nyu.edu

ABSTRACT

In this paper we explore a method for automatically generating Carnatic style rhythmic. The method uses a set of annotated Carnatic percussion performances to generate new rhythmic patterns. The excerpts are short percussion solo performances in ādi tāla (8 beat-cycle), performed in three different tempi (slow/moderate/fast). All excerpts were manually annotated with beats, downbeats and strokes in three different registers — Lo-Mid-Hi. N-gram analysis and Markov chains are used to model the rhythmic structure of the music and determine the progression of the generated rhythmic patterns. The generated compositions are evaluated by a Carnatic music percussionist through a questionnaire and the overall evaluation process is discussed. Results show that the system can successfully compose Carnatic style rhythmic performances and generate new patterns based on the original compositions.

1. INTRODUCTION

Automatic generation of music has been a focus of computational music research for a long time. Researchers have been designing systems to imitate or compose various musical styles from Classical to Jazz music [1], [2]. Despite the progress achieved so far in the development of generative music systems for Western music genres there is limited work regarding methodologies of automatic generation of music in non-western styles.

In this paper, we propose CAMeL an automatic music generation system, which focuses on the generation of Carnatic style rhythms. Carnatic music is an art music tradition from South India with a long history, which has its own musical grammar and significant musicological literature [3]. Carnatic music has a very well defined rhythmic framework and an interesting rhythmic structure, which makes it interesting and challenging to explore in an automatic music generation system. The approach proposed in this paper is focused on percussion-based Carnatic music style rhythms using a set of annotated training data of music excerpts. The annotations include the stroke register (Lo-Mid-Hi), the inter-onset interval duration of the strokes and the amplitude of the music excerpts. By extracting these features the system is capable of automatically generating new rhythmic progressions stylistically similar to the training compositions. N-gram analysis and statistical learning is used to model the rhythmic structure using the extracted features. Markov chains are then used to build the rhythmic development and describe the pattern transition likelihoods of the generation sequences. The system generates rhythmic patterns based on an n-gram input. If a five-gram analysis is selected then the algorithm generates the strokes using the transition probability of the five-grams.

The proposed method for generating rhythmic pattern progression of Carnatic style music was evaluated by a professional Carnatic percussionist — Akshai Anantapadmanabham. The same percussionist composed and performed the datasets for training the system. The evaluation is based on feedback of the rhythmic structure and development of the generated sequences compared to a human-based performance. The results of the evaluation provide insights into the rhythmic organization and interpretation of the generated rhythmic patterns.

Musicians can use the proposed system for creative purposes in their performance and training. It can be also used as a tool in music education as a means of actively enculturing lay people into this music style; for example, by creating software applications that include generative systems of Carnatic music, allowing users to “play” Carnatic music percussion on mobile devices and get entrained in this style by getting familiar with the underlying rhythmic structure and grammar of this music.

The paper is organized as follows: section 1.1 presents background information on the rhythmic structure in Carnatic music while section 2 presents previous research on automatic music generation methods. Section 3 describes the proposed approach while section 4 discusses the evaluation of the method. Discussion and Conclusions are drawn in sections 5 and 6 respectively.

1.1 Rhythmic structure in Carnatic music

The rhythmic framework of Carnatic music is based on the tāla, which provides a structure for repetition, grouping and improvisation. The tāla consists of a fixed time length cycle called āvartana, which can also be called the tāla cycle. The āvartana is divided into equidistant basic time units called akṣaras, and the first akṣara of each āvartana is called the sama [3]. Two primary percussion accompaniments in Carnatic music are the Mridangam and Kanjira. The Mridangam is made of a cylindrical shell with stretched membranes on either side of the instrument body. While one side is loaded with a black paste that creates a pitched tone, the other membrane creates a bass-like sound. The Kanjira on the other hand is a frame-drum, with a tonally rich membrane. Unlike the Mridangam, the Kanjira is not tuned to a specific key, but it can cover a wide range of frequencies with especially rich lower frequencies. The rhythmic complexities of Carnatic rhythm are especially showcased during the solo or taniavartanam. First, each instrument performs separately and then they trade off in shorter cycles with a precise question-answer like session, followed by a joint climactic ending. All training excerpts used in the proposed generation method were performed by the Kanjira drum in the context of a concert solo. We decided to use the Kanjra compositions as a training corpus because the strokes had a simpler frequency distribution compared to the Mridangam.

2. RELATED WORK

Probably the most popular study of musical style imitation is David Cope's Experiments in Musical Intelligence (EMI) system. EMI analyzes the score of MIDI sequences in terms of patterns and stores the patterns in a database where the system learns the style of a composer given a number of training examples [4]. Bel and Kippen [5] present the Bol Processor, a software system that models tabla drumming improvisation. The system is based on a linguistic model derived from pattern languages and a formal grammar that has the ability to handle complex structures by using a set of training examples. Dias and Guedes in [1] discuss a contour based algorithm for real time automatic generation of jazz walking bass lines, following a given harmonic progression. The algorithm generates melodic phrases that connect the chords in a previously defined harmonic grid, by calculating a path from the current chord to the next, according to user-defined settings controlling the direction and range of the melodic contour. Biles in [2] developed a generative system for composing jazz solos based on a genetic algorithm, which starts with some initial musical data initialized randomly or by human input. Using a repeated process similar to biological generation the system produces similar musical data. Dias et al [6] present the GimmeDaBlues app that allows the user to play jazz keyboard and solo instruments along a predefined harmonic progression, by automatically generating the bass and drums parts, responding to the user's activity. Assayag, Dubnov and Delerue [7] proposed a dictionary based universal prediction algorithm that provides an approach to machine learning in the domain of musical style. Operations such as improvisation or assistance to composition can be realized on the resulting representations. The system uses two dictionaries, the motif and continuation. A generation algorithm is used to predict a sequence based on the motif dictionary. The continuation dictionary gives probabilities of various continuations and is used to determine the next symbol. Pachet discusses the continuator [8] an interactive imitation system, which generates new melodic phrases in any style, either in standalone mode or as continuations of musician's input. The system is based on an incremental parsing algorithm to train a variable-length Markov chain that stores possible probabilities of sequences. The system progressively learns new phrases from a musician and develops a robust representation of his or her style. A framework for generating similar variations of guitar and bass melodies is proposed in [9]. The melody is initially segmented into sequences of notes using onset detection and pitch estimation. A set of hierarchical representations of the melody is estimated by clustering the pitch values. The pitch clusters and the metrical locations are then used to train a prediction model using variable-length Markov chain.

3. SYSTEM IMPLEMENTATION

3.1 Dataset

The training corpus consisted of 8 percussion solo compositions in *ādi* tāla (8 beat- cycle) in three different tempo levels (slow/moderate/fast). The compositions were performed by Akshay Anantapadmanabham, in the Kanjira. These examples were recorded using a metronome.

All excerpts were manually annotated using Sonic Visualizer [10] including the sāmā and the other beats comprising the tāla. Each stroke event was coded as a string based on its register (Lo-Mid-Hi), the inter-onset-interval (IOI) between strokes and a value indicating the velocity of the stroke. The fourth author annotated the music excerpts by using the following process: The metronome was recorded in a separate channel and used as reference for each performance. A note onset transformation was estimated for the audio track by which note onsets were detected. By looking at the note onsets, the spectrogram of the sound and by listening to it at a reduced playback speed, the different types of strokes were categorized into three categories and the annotation marker positions were manually adjusted. Based on the spectrogram analysis, the frequency spectrum of the strokes was divided into three frequency bands (low, mid and high) depending on the frequency content of each stroke (110-190 Hz for low, 190-600 Hz for mid and 600-1200 Hz for high strokes). Although the Kanjira has a richer variety of registers and strokes, the reduction to three registers was a step to simplify the different stroke definition. This reduction was validated by Anantapadmanabham as a process to faithfully encode the different strokes in the Kanjira**.** The normalized velocity values of the strokes were obtained by computing an onset detection function, and estimating its amplitude level with a value between 0.2 and 1 according to the strength of the stroke. In the present work, the complex domain onset detection [11] was used to compute the onset detection function implemented in the Vamp-plugins in version of Sonic Visualizer. Table 1 lists the coded feature values used to model each stroke event.

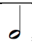
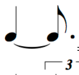
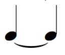
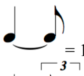
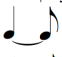
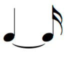
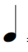
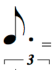
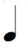
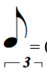
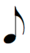
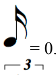
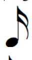
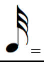
Features	Value
Register	Lo-Mid-Hi
IOI duration (sec)	T1  = 2
	T2  = 1.75
	T3  = 1.66
	T4  = 1.5
	T5  = 1.33
	T6  = 1.25
	T7  = 1
	T8  = 0.75
	T9  = 0.66
	T10  = 0.5
	T11  = 0.33
	T12  = 0.25
	T13  = 0.16
	T14  = 0.125
Velocity	V1 (0.2) V2 (0.5) V3 (1.0)

Table 1. Features for modeling stroke events.

3.2 N-gram model

All coded stroke events from the compositions were merged in a single training corpus to learn a statistical model. We used n-gram analysis to model the underlying rhythmic progression of the strokes in the training data. The general n-gram definition is given in (1), while the representations of a unigram, bigram and trigram are given in (2), where s denotes a stroke event.

$$p(s_i \mid s_1,\ldots,s_{i-1}) = p(s_i \mid s_{i-n+1},\ldots,s_{i-1}) \tag{1}$$

$$\begin{array}{ll} \textit{unigram}: & p(s_i) \\ \textit{bigram}: & p(s_i \mid s_{i-1}) \\ \textit{trigram}: & p(s_i \mid s_{i-2}, s_{i-1}) \end{array} \tag{2}$$

An example of a trigram encoding the strokes is given below:

MidT10V1 LoT12V3 LoT12V3

This trigram consists of three stroke events. The first stroke has a Mid register with an eighth note duration performed with 0.2 velocity followed by two strokes with Lo register and sixteenth note duration performed with 1.0 velocity.

We estimated the n-gram probabilities up to a five-gram by counting the frequency of the strokes on the training corpus where N is the total numbers of stroke events in the training data. The unigram and bigram probabilities are calculated using equations (3) and (4) where s_a denotes a particular stroke event, s_b its preceding stroke and c the count of a stroke:

$$\hat{p}(s_a) = \frac{c(s_a)}{N} \tag{3}$$

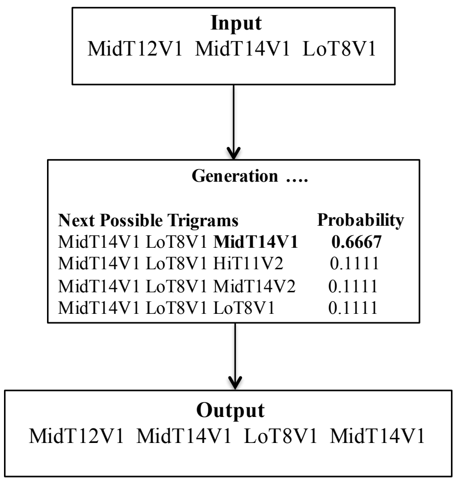
$$\hat{p}(s_b \mid s_a) = \frac{c(s_a,s_b)}{\Sigma_{sb} c(s_a,s_b)} \approx \frac{c(s_a,s_b)}{c(s_a)} \tag{4}$$

The n-gram model provides the transition probabilities between stroke events. For example, consider the case of a bigram model where two stroke events are present. The first tagged as Mid stroke register with a quarter note duration and with 0.2 velocity value and the second as a Lo stroke register with a sixteen note duration and velocity value of 0.2. What would the probability be that the next stroke will be a Lo stroke register with a sixteen note duration and high velocity given the previous stroke events representation?

We computed all the n-grams probabilities up to a five-gram because we wanted to test how past information and size of accumulated memory could affect and change the generation process. All n-gram probabilities were stored in tables to be used later during the generation process. The generation process used these data to generate new strokes events sequentially. Given a sequence of strokes, a stroke event is generated based on the weighting probability of the most likely stroke to follow given the previous strokes.

3.3 Generation

The generation process depends on the n-gram selection of and on the number of stroke events. If a trigram is selected the generation starts with the first trigram of the training file. The next stroke event is generated based on the probabilities of trigrams that start with the last two stroke events in the generated sequence. When the stroke event is generated the algorithm looks for the next last two stroke events in the sequence to generate the next stroke and search again for the highest probability of trigrams that start with the last two stroke events. This process is iterative until the number of initial selected stroke events is reached. The overall process is presented in Figure 1.



Input
MidT12V1 MidT14V1 LoT8V1
Generation ….
Next Possible Trigrams
Probability
MidT14V1 LoT8V1 MidT14V1 0.6667
MidT14V1 LoT8V1 HiT11V2 0.1111
MidT14V1 LoT8V1 MidT14V2 0.1111
MidT14V1 LoT8V1 LoT8V1 0.1111
Output
MidT12V1 MidT14V1 LoT8V1 MidT14V1

Figure 1. Generation process using a trigram model and probability estimation.

4. EVALUATION

Several approaches for the evaluation of generative music systems have been proposed in the past. Researchers have tried to use Turing tests [12] to compare the output between computer-aided and non-computer aided compositions by measuring the degree of perceptual quality. This model has been criticized in the past for its application in executing and evaluating listener surveys [13]. Pearce and Wiggins [14] use a set of musical examples to train a genetic-algorithm based system. A discrimination test is used to evaluate whether the output of the system can be distinguished from the training compositions. Cont, Dubnov and Assayg [15] evaluate a generative system using the same model as a classifier. The model is trained for a particular style of music and outputs a probability to a given music excerpt. A quasi-Turing test is used in [5] to evaluate the Continuator. The evaluation is used to assess to what extend a listener can determine that a melody generated by the system was composed or played by a human or by a machine. Collins [16] evaluates an algorithmic system that creates electroacoustic art music by using three expert composer judges. They evaluate the system based on how music material was assembled its form, structure and instrumentation. The authors in [9] use a group of experts to evaluate an automatic guitar and bass phrase continuation melody system. Their feedback is related to the type of similarities and differences they notice between the original and generated examples and the aesthetic outcome.

Since we are interested in generating new sequences of Carnatic music percussion from the training data and there is no benchmark dataset for music generation performance of other systems we decided to conduct a preliminary evaluation based on the feedback of the musician who also provided the dataset. The fact that Anantapadmanabham is an expert in Carnatic music percussion and also provided the dataset that we used for analysis provides a unique set of conditions to do a preliminary evaluation of the generative model and this approach.

A questionnaire was prepared and presented to Anantapadmanabham. The new sequences were generated using different n-grams (bigram, trigram, fourgram and fivegram) with duration of 2 minutes each. He was asked first to listen to the compositions as many times needed to get familiar with the rhythmic structure and development of the excerpts and then answer the questionnaire.
Examples of the generated excerpts can be downloaded at https://github.com/Trochidis/CAMeL-Carnatic-Percussion-Music-Generation-Using-N-Gram-Models.
Anantapadmanabham was first asked to judge if the generated compositions contained recognizable Carnatic music rhythmic patterns, which he positively answered. The next question of the form was related to the short-term level of rhythmic structure asking if the rhythmic patterns were occurring in metrical appropriate positions. He answered that sometimes they were and others they were not. Based on his feedback there were certain strokes, particularly in the percussive roll sections of the generated compositions that they were repeated consecutively. This sometimes created a feeling that the same succession of strokes kept playing without variation which does not usually happen in the rhythmic structure of Carnatic music. The next question was related to the long-term evolution and rhythmic progression asking if the rhythmic structure of the generations evolved in time as expected in this style. He answered that most of the generated compositions in particular the ones with the shorter memory (bigram-trigram) failed to capture the long-term rhythmic structure and the correct transition between longer rhythmic structures.
His additional comments were that n-grams with larger memory such as fourgrams and fivegrams were more successful in capturing Carnatic rhythm groupings compared to bigrams or trigrams and contained more rhythmic patterns in resemblance with the original Carnatic music patterns.

5. DISCUSSION

This work presents a method for automatically generating new Carnatic style rhythmic patterns based on a set of training examples. An n-gram analysis and Markov Chains are used to model short and long-term patterns and represent rhythmic progressions. Based on the expert's feedback the method is able to generate recognizable Carnatic-style rhythmic patterns with some success. The evaluation indicates that the n-gram analysis is more successful on capturing short-term rhythmic patterns compared to long-term ones. Moreover, larger n-grams such as fourgrams or fivegrams generate more appropriate and interesting Carnatic style rhythmic patterns. This is due to the fact that they are more successful in modeling the long-term pattern transitions compared to shorter structures such as unigrams and bigrams. An improvement over the current method will be to implement a cluster analysis to the larger n-grams i.e fivegrams or sevengrams and generate rhythmic patterns based on the cluster transitional probabilities. This might improve the rhythmic representation and progression of long-term rhythmic structures compared to the one implemented in our current system.

Another approach to tackle the problem of long-term representation of rhythmic progression is to use a Long Short Term Memory Recurrent Neural Network (LSTM-RNN) [17]. LSTM RNNs are capable of learning long-term dependencies and have been used successfully in language modeling and speech recognition. The LSTM RNNs architecture is based on a dynamic memory with cells that stores information about the previous states. They can combine previous states and current memory to make decisions and efficiently capture long-term dependencies by dynamically changing their memory.

6. CONCLUSION

In the present paper, a method for automatically generating Carnatic style rhythmic patterns is explored. By extracting features such as the stroke register (Lo-Mid-Hi), inter-onset interval duration of the strokes and amplitude of the strokes the system is capable of automatically generating new rhythmic progressions stylistically similar to the training compositions. N-gram analysis and statistical learning is used to model the rhythmic structure and build the rhythmic development using the extracted features. The generated outcome was evaluated by a professional composer and percussionist of Carnatic music in terms of rhythmic development and musical aesthetics. Feedback from the evaluation shows that the method is capable of generating new interesting Carnatic style rhythmic patterns by training on previous data. Future work will 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. Furthermore, we would like to perform a statistical analysis of the evaluation results to test the percentage and the strength of the generated excerpts that were positively evaluated by the human-experts. Finally, we aim to test the method against other approaches such as clustering and deep belief networks.

7. REFERENCES

[1] R. Dias, C. Guedes, "A Contour-Based Jazz Walking Bass Generator." Proceedings of the Sound and Music Computing Conference, 2013.

[2] J.A. Biles, "GenJam in Transition: from Genetic Jammerto Generative Jammer", International Conference on Generative Art, Milan, 2002

[3] P. Sambamoorthy, South Indian Music Vol. I-VI, The Indian Music Publishing House, 1998.

[4] D. Cope, Experiments in musical intelligence (Vol. 12). Madison, WI: AR editions, 1996.

[5] B. Bel & J.Kippen. Modelling music with grammas: formal language representation in the Bol Processor. Computer Representations and Models in Music, Academic Press, pp.207-238, 1992

[6] R. Dias, T. Marques, G. Sioros and C. Guedes, "GimmeDaBlues: an intelligent Jazz/Blues player and comping generator for iOS devices". in Proc. Conf. Computer Music and Music Retrieval (CMMR 2012), London 2012.

[7] G. Assayag, S. Dubnov, & O. Delerue. "Guessing the composer's mind: Applying universal prediction to musical style", In Proceedings of the International Computer Music Conference, 1999, (pp. 496-499).

[8] F. Pachet. "The continuator: Musical interaction with style", in J. New Music Research, 2003, 32(3), 333-341.

[9] S. Cherla, H. Purwins, & M. Marchini, "Automatic phrase continuation from guitar and bass guitar melodies", in Computer Music Journal, 2013, 37(3), 68-81.

[10] C. Cannam, C. Landone, & M. Sandler, "Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files", in Proc. Int. Conf. on Multimedia, 2010, (pp. 1467-1468). ACM.

[11] C. Duxbury, J.P Bello, M. Davies & M. Sandler, "Complex domain onset detection for musical signals", in Proc. Digital Audio Effects Workshop (DAFx), 2003, (No. 1, pp. 6-9).

[12] A. M. Turing, "Computing machinery and intelligence". Mind, 1950, 59(236), 433-460.

[13] C. Ariza, "The interrogator as critic: The turing test and the evaluation of generative music systems", Computer Music Journal, 2009, 33(2), 48-70.

[14] M. Pearce, and G. Wiggins, "Towards a Framework for the Evaluation of Machine Compositions", in Proc. of the AISB01 Symposium on Artificial Intelligence and Creativity in the Arts and Sciences. Brighton, UK: SSAISB, 2001, pp. 22–32

[15] A. Cont, S. Dubnov, & G. Assayag, "Anticipatory Individual Behavior-Anticipatory Model of Musical Style Imitation Using Collaborative and Competitive Reinforcement Learning", Lecture Notes in Computer Science, 2007, 4520, 285-306.

[16] N. Collins, "Automatic composition of electroacoustic art music utilizing machine listening", Computer Music Journal, 2012, 36(3), 8-23.

[17] N. Boulanger-Lewandowski, Y. Bengio and P. Vincent, "Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription", In Proceedings of the 29th International Conference on Machine Learning (ICML), 2012