Interspeech 2018
2-6 September 2018, Hyderabad

Prediction of Aesthetic Elements in Karnatic Music: A Machine Learning Approach

Ragesh Rajan M[1], *Ashwin Vijayakumar*[2], *Deepu Vijayasenan*[1]

[1]National Institute of Technology Karnataka, India
[2]Georgia Institute of Technology, Atlanta, USA

mrageshrajan@gmail.com, ashwinkv@gatech.edu, deepu.senan@gmail.com

Abstract

*Gamaka*s, the embellishments and ornamentations used to enhance musical experience, are defining features of Karnatic Music (KM). The appropriateness of using *gamaka* is determined by aesthetics and is often developed by musicians with experience. Therefore, understanding and modeling *gamaka* is a significant bottleneck in applications like music synthesis, automatic accompaniment, *etc.* in the context of KM. To this end, we propose to learn both the presence and the type of *gamaka* in a data-driven manner using annotated symbolic music. In particular, we explore the efficacy of three classes of features – note-based, phonetic and structural – and train a Random Forest Classifier to predict the existence and the type of *gamaka*. The observed accuracy is $\sim$70% for *gamaka* detection and $\sim$60% for *gamaka* classification. Finally, we present an analysis of the features and find that frequency and duration of the neighbouring notes prove to be the most important features.

Index Terms: *gamaka*, Karnatic Music, Symbolic Music, Random Forest Classifier.

1. Introduction

Karnatic Music (KM) is an ancient classical music system with its origins in South India. A distinct feature of KM is the usage of *gamaka*s – graceful curves in the pitch of a note that serve as embellishments. The aesthetic appeal of a performance is largely determined by the usage of *gamaka* that is dependent both on the *rāga*, the fundamental melodic structure in KM, and the musician. The rules for assigning an embellishment to a particular note in KM is neither too open-ended nor very specific [1]. For example, certain musical phrases demand the use of specific *gamaka*s, while certain other phrases allow some amount of flexibility in choosing *gamaka*s. Further, KM performances are predominantly improvisational, and the usage of *gamaka* varies considerably across musicians. Since there can be multiple aesthetically correct variants of using *gamaka* and may further vary significantly across musicians, it is hard to encode or fix it as part of musical notation. Therefore, performing both composed music and improvising requires a certain amount of musical maturity.

Importance in Music Synthesis. Existing methods that explicitly model *gamaka* [2, 3, 4, 5] overcome the complexity involved in understanding *gamaka* by allowing 1) only a small fraction of *gamaka*s that are predetermined for a particular *rāga* 2) using expert knowledge to construct templated *gamaka* in terms of pitch track, tempo and amplitude. By design, these models have multiple drawbacks including that they do not scale to a large number of *rāga*s and require extensive expert tuning. In the direction of better integration of symbolic music and *gamaka*, the use of *melodic atom*s has been proposed in the literature [6]. The *melodic atoms* include both the note and the *gamaka* as the building blocks. Again, similar to previous synthesis techniques, it requires experts to decide what the *melodic atom*s for a *rāga* are, as there can be innumerable *melodic atom*s in each *rāga*.

***Gamaka*s in Karnatic Music.** Although a comprehensive and fairly general understanding of *gamaka* has been elusive, traditional musicologists have classified *gamaka*s into ten, typically based on the playing techniques of the musical instrument, *vīna* [7]. Further attempts by musicologists to adapt this to singing and more general music performance has seen a trend of increasing complexity and variance across different schools of thought. Therefore, this structured approach is largely abandoned in practical KM training, and most musicians develop a sense for the usage of *gamaka* by ear and experience. However, Subbarāma Dīkshita, an influential musicologist and composer, documented music compositions along with a novel annotation scheme for *gamaka* in his seminal work, Sangīta Sampradāya Pradarshini [8] in the early 20th century. While it is possible to perform the same compositions with other variations, we use this version as the *gold standard* and learn the nuances of *gamaka* in a symbolic music setting.

Related Work. Most of the existing works on *gamaka* prediction and classification use acoustic data for the task. Very few works have addressed the problem of predicting *gamaka* from annotated symbolic music. As part of expressive music synthesis, the ornamentations in *Jazz Guitar* has been predicted [9] using the RIPPER algorithm [10]. The extracted features were pitch and duration of the present note and neighbouring notes, chord, key and the perceptual parameters. Using a reduced feature set, an average accuracy of 70% was observed. Note that this was better than a random classifier by only 3.45%. With an increased dataset of 27 songs, classifiers such as SVM, ANN, Decision Trees and KNN were tried [11]. Interestingly, decision trees performed the best at this classification task yielding the best accuracy (78.68%), which was better than the random classifier by only 4.9%. In the context of Indian Classical Music, multiple works have addressed *gamaka* identification from raw audio signal [12, 13]. Other research efforts on the automatic music composition [14, 15, 16, 17] and singing voice synthesis [18] for Indian Classical Music do not address the problem of predicting *gamaka*s from the score files.

Contributions. In this paper, we describe an approach to detect the presence and type of *gamaka* associated with a note in KM from the annotated symbolic music. To the best of the authors' knowledge, this work is the first one of its kind. We propose three sets of features based on the notes, lyrics and

10.21437/Interspeech.2018-991

the structure of the song. We train a Random Forest Classifier (RFC) for the detection and classification of *gamaka*s and further, analyse the importance of these features for our task.

2. Background

In the commonly followed tradition of KM, an octave contains a maximum of 12 note positions. Unlike western music, the frequencies associated with the note positions are not fixed in KM. The note frequencies are defined relative to an arbitrary reference frequency. In this work, we denote the note positions using English alphabets and numbers. Deviating from standard convention practiced by musicians, we label the note positions across octaves using numbers, to mean octaves 1,2 and 3 stand for lower, middle and upper octaves, respectively. For example, the 12 note positions in the lower octave are represented as *S1*, *r1*, *R1*, *g1*, *G1*, *m1*, *M1*, *P1*, *d1*, *D1*, *n1* and *N1*. Corresponding middle octave positions are represented as *S2*, *r2*, ..., *N2*.

Rāga, the fundamental melodic structure in KM, is characterized by an ordering of notes and associated note-sequences. *Rāga* also serves as the basis for improvisation and evoking various emotions. Different *rāga*s can have different number of note positions per octave. For example, the *rāga* named *Mōhanam* is a penta-tonic *rāga*, which has only five notes – *S*, *R*, *G*, *P* and *D* – per octave, and these may occur in any arbitrary order.

*Gamaka*s account for the variability of notes in KM. They add naturalness to the music from the KM perspective. A note is considered to be properly defined only when the *gamaka* associated with it is properly addressed. Note that for the same note, the usage of *gamaka* may vary from one *rāga* to another. The performance of both improvisational and composed music depends on the musician's understanding of the *rāga* and the usage of *gamaka* in that melodic framework. Sangītha Sampradāya Pradarshini [8] classifies *gamaka*s into 15 types, out of which, some are very specific to the musical instruments.

3. Data Details

For our experiments, we consider all the songs in one of the popular *rāga* (*Kalyāni*) and its derived *rāga*s[1] from the English translation of Sangīta Sampradāya Pradarshini [8]. We digitize the information given in this book to construct a dataset. This dataset contains the lyrics, notes, duration of notes, *gamaka*s and other structural information of 25 songs across nine composers. The number of songs taken from each *rāga* is listed in the Table 1. There are 9986 notes, out of which, 4013 notes have *gamaka*s and 5973 notes are plain notes. There are only 10 different types of *gamaka*s in the dataset. Due to lack of training samples (less than 10), we neglect two types of *gamaka*s and consider only eight different classes. The different *gamaka*s considered in this paper are: *Podi*, *Kampitham*, *Sphuritham*, *Nokku*, *Ētra Jāru*, *Irakka Jāru*, *Orikkai* and *Othukkal*.

4. Experiments

Initially we explore the possibility of detecting the presence of *gamaka* in a note. For this purpose three different feature sets based on adjacent notes are proposed – absolute and relative frequencies, lyrics (phonetic classes like vowels, fricatives, etc.) and the structure (such as beat cycle information). We select a training set with 13 songs covering 4 *rāga*s. Two test sets are constructed. Test set 1 (seen *rāga* test set) is formed by 10 songs belonging to the same set of *rāga*s in the training set. The test set 2 (the unseen *rāga* test set) consists of two songs belonging to *rāga Hamvīru*, which is not present in the training set. Thus it is possible to analyze the performance of the system for seen/unseen *rāga*s separately. Table 1 lists the number of songs in training/test sets. There are a total of 6198 notes in the training set, 3260 notes in the seen *rāga* test set, and 528 notes in the unseen *rāga* test set.

[1] a derived *rāga* consists only of a subset of the notes of the parent *rāga*

Table 1: *Number of songs in each* rāga *in the dataset*

Rāga	Training Set	Test Set 1	Test Set 2	Total Songs
Kalyāni	5	4	–	9
Mōhanam	3	2	–	5
Sāranga	2	2	–	4
Yamunā kalyāni	3	2	–	5
Hamvīru	–	–	2	2
Total	13	10	2	25

4.1. Note-based Features

Note based features represent the frequency positions and the duration of the notes in the song. The 36 notes covering the three octaves (12 notes per octave) are sorted in the increasing order of frequency. The frequency positions of these notes are represented using numbers from 1 through 36.

We consider a context of five notes, centered around the note under consideration, for feature extraction. The frequency position of the middle note is used as a feature. In addition, the difference in frequency positions of the four adjacent notes with respect to the middle note are used as features. Features also include durations of all the five notes in the context. In addition, we have four more *rāga*–dependent features. Here, we consider only the notes present in the *rāga* of the song. For creating these features, *rāga*–specific frequency positions are assigned to notes. For example, in the *rāga* called *Mōhanam*, only five notes – *S*, *R*, *G*, *P* and *D* – are present in one octave. Thus the three octaves are represented by numbers from 1 through 15. The *rāga*–dependent relative positions of the four neighbouring notes are found out with respect to the middle note. Thus the 14 dimensional feature set is formed.

For example, let a set of contextual notes be *S2 R2 S2 D1 S2*, from a song belonging to the *rāga Mōhanam*. Let *S2* in the middle of the set be the note under consideration. The middle note's absolute frequency position can be found out to be 13. The relative position of the preceding 2 notes with respect to the middle note will be +2 (note *R2*) and 0 (note *S2*). Similarly, for the succeeding notes *D1* and *S2* it will be -3 and 0, respectively. From the *rāga*–based representation of *Mōhanam*, the relative frequency positions of the neighbouring notes *S2*, *R2*, *D1* and *S2* are 0, +1, -1 and 0, respectively.

We use a Random Forest Classifier (RFC) for detecting the presence of *gamaka*s in the notes. The parameters of the classifier are set to the default values. In order to evaluate the performance of note-based features in detecting the *gamaka*s, we perform a cross validation across all the 13 songs in the training set. The cross validation is performed in a round-robin fashion, by excluding each of the 13 songs from the training set and testing on the excluded song. The average accuracy obtained is 71%. From the accuracy score, it can be concluded that this

feature set contains information about the presence/absence of *gamaka*s associated with the notes.

We also explore the use of Artificial Neural Network (ANN) classifier to evaluate the performance of note–based features. Maximum cross validation accuracy obtained using ANN is found to be 67%. We observe that the training accuracy itself is inferior to that using RFC by around 15%. This may be because of the nominal nature of the features used. Therefore we do not explore ANN classifier for the rest of our experiments.

4.2. Phonetic Features

Phonetic features contain information about the phonetic classes of the lyrics of the songs. We try to divide the lyrics into five phonetic classes, namely, vowel, fricative, stop, nasal and glide. Each phoneme in the lyrics of the songs falls into any of the five phonetic classes. To construct this feature set, we consider a context of four phonetic entities from the lyrics associated with each note. Starting and ending phonemes associated with the current note form two entities. Starting phoneme associated with the succeeding note, and the end phoneme associated with the preceding note form the other two phonetic entities. A five dimensional binary vector is formed for each of the above phonetic entities, indicating the phonetic class of that entity. Thus the feature set is a 20 dimensional binary vector.

We use the same set up as in the case of note-based features for conducting experiments to evaluate the performance of phonetic features. The average accuracy is observed to be 56.74%. It seems that the lyrical information does not play a major role in detecting the presence of *gamaka* in a note.

4.3. Structural Features

These features contain information about the structural elements of the song. In KM, a song is divided into different sections such as *pallavi*, *anupallavi*, *charanam*, *swaram*, etc. *Pallavi*, the beginning segment of the song, often gets repeated after every other segment. *Pallavi* is followed by *anupallavi*, *charanam* and *swaram*. We add 4 binary features to represent the section to which the current note belongs. We also consider other structural elements such as the beginning and end points of beat cycles and sub-beat cycles, the notes starting 'off-beat', etc. There are 12 such classes, and a 12-dimensional binary vector indicates the presence of these classes associated with each note. After conducting the experiments on the training set, the average accuracy using the structural features is found to be 57.86%. The accuracy is better than that of the phonetic features by 2.53%. But it is inferior to the accuracy obtained using note-based features by 12.14%. It can be inferred that the structural features do not contain as much information as in the case of note-based features to distinguish between *gamaka*s and non-*gamaka*s.

4.4. Feature Combination

In this experiment, we aim to test the amount of additional information that can be provided by the structural and phonetic features, when used along with the note-based features. We combine all the features to make a 46 dimensional feature vector. With these features, we conduct the experiments on the training set using the RFC. The accuracy obtained after cross validation is 70.13%. This result is inferior to the one obtained for note based features alone. To analyze this, we look at the training accuracy with the same experimental conditions. The training accuracy is found to be 98.6%. The wide gap between the train and cross validation accuracies indicates that there is an over-fit.

The over-fit problem can be reduced by employing regularization. We perform regularization by increasing the number of estimators and/or increasing the minimum number of samples required to split a node. We perform a grid search of these hyper–parameters jointly to get the maximum training cross validation accuracy. Figure 1 shows an example plot of accuracy versus different values of number of estimators.

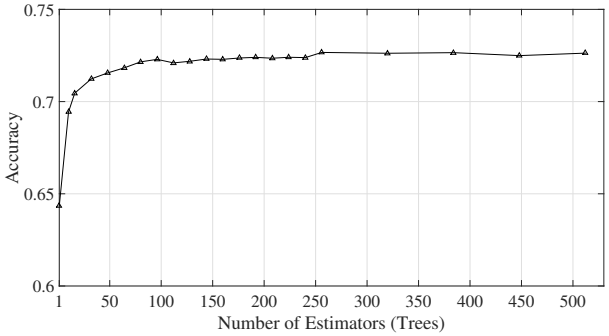

Figure 1: *Change in accuracy with respect to number of estimators*

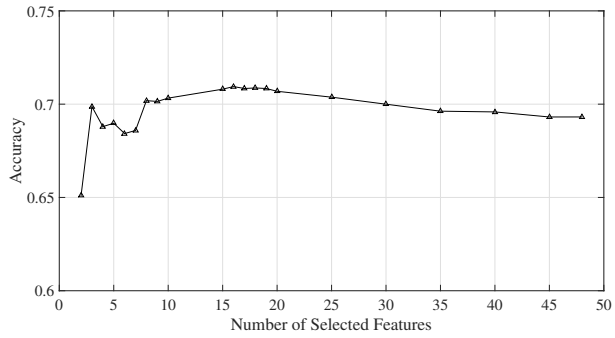

Figure 2: *Change in accuracy with respect to number of selected features*

The optimum values for the number of estimators and the minimum number of samples needed to split the node are found to be 256 and 4, respectively. Using these hyper–parameters, the training accuracy reduces to 96.4% while the cross validation accuracy increases to 72.73%. The cross validation accuracies for different experimental conditions are listed in Table 2.

Table 2: *Cross validation results across experimental settings.*

Default Parameters	Tuned Parameters	
Full Features	Full Features	Selected Features
70.13%	72.73%	73.53%

Another option to tackle the over-fit problem is feature selection. We perform feature selection based on the *feature importance parameter* of RFC. The importance of a feature is computed as the reduction of information when a node is split using this feature [19]. The variation of cross validation accuracy with respect to the number of selected features is illustrated

in Figure 2. 17 best features having maximum information reduction are selected. All the 14 note-based features happen to be in the selected features. This shows their importance in detecting *gamaka*s. From the other two sets of features, only two phonetic features and one structural feature are selected. Hence it can be concluded that in our experiments, the structural and phonetic features do not have much of the information regarding *gamaka*s. The 17 selected features are listed in Table 3.

To evaluate the performance on the test data, we use the optimized hyper–parameters and reduced feature set. The results obtained for different test sets are listed in Table 4. RFC performance is better than the random classifier (classification by chance) by around 10%. It is also clear that the test accuracies obtained for the seen *rāga*s do not deviate much from the cross validation accuracies. But in the case of unseen *rāga*, there is a reduction in test accuracy, especially with the selected features. This may be because of the fact that, the features selected are tuned only for the seen *rāga*s.

Table 3: *Selected features*

Note_Post$_1$	Note_Rāga_Post$_1$
Note_Post$_2$	Note_Duration_Post$_1$
Note_Pre$_1$	Note_Duration_Pre$_1$
Note_Current	Note_Duration_Post$_2$
Note_Rāga_Pre$_1$	Note_Duration_Pre$_2$
Note_Pre$_2$	Charanam
Note_Rāga_Pre$_2$	Prev_Note_End_Stop
Note_Duration_Current	Post_Note_Begin_Stop
Note_Rāga_Post$_2$	

Table 4: *Test accuracies for different features*

Test Set	Random Classification Accuracy	Accuracy	
		Full Features	Selected Features
Seen *Rāga*s	60.21%	72.47%	72.1%
Hamvīru	57.57%	70.62%	67.03%

4.5. *Gamaka* Classification

In the next set of experiments, we try to classify the *gamaka*s associated with a note into 8 different classes detailed in Section 3. The training and test sets contain the same songs as in the previous experiment. But for the *gamaka* classification experiment, we consider only the notes which contain *gamaka*s. The plain notes are removed from the test and training sets. Notes having *gamaka* are selected manually from the ground truth. The training set for this experiment consists of 2534 notes, and the seen and unseen *rāga* test sets consist of 1261 notes and 218 notes, respectively.

For this experiment, we follow the same procedure as in the detection experiment. 22 features are selected, which yield the best accuracy. Out of these 22 features, 17 are the selected features from the detection experiment. Five more phonetic features are also selected. The classification accuracies for the RFC using two different feature sets and for different test sets are listed in Table 5. The accuracies for the random classification (selecting the most probable class) are also listed in the table. It can be seen that the accuracies using RFC are better than that for the random classification, by around 40%.

Table 5: *Test accuracies for* gamaka*s*

Test Set	Random Classification Accuracy	Accuracy for RFC	
		Full Features	Selected Features
Seen *Rāga*s	22.8%	58%	58.8%
Hamvīru	24.5%	58.7%	63.5%

Precision and recall values for individual *gamaka*s for the seen and unseen *rāga*s are shown in the bar graph given in Figure 3. The *gamaka* with the least value for precision/recall is *Kampitham*. This *gamaka* has less training examples (∼180). We also observe a confusion in deciding between the *gamaka*s *Ētra Jāru* and *Othukkal*, which may account for the reduction in the recall value of *Othukkal*. In these experiments, we had access only to a limited amount of data. We envisage that the accuracies can be improved by conducting the experiments on a larger dataset.

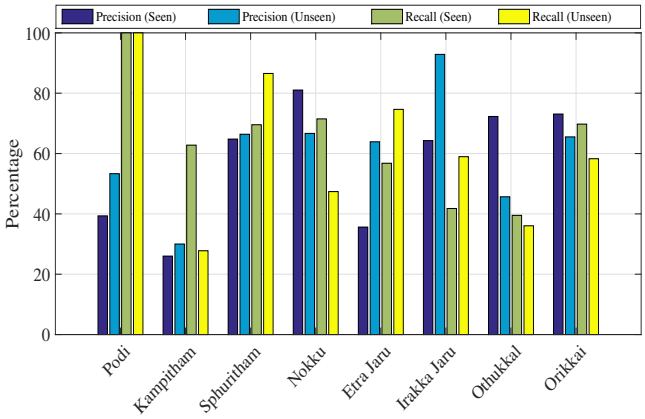

Figure 3: *Precision and recall for different* gamaka*s (for both the seen and unseen rāga test sets)*

5. Conclusion and Future Work

In this work, we detect the presence as well as the type of *gamaka*s from KM notations, using a data-driven approach. We propose three different sets of features – based on the frequencies and durations of the notes, phonetic class of the lyrics and the structure of the song. RFC is used for the detection and classification. *Gamaka* detection experiments on different feature sets reveal that note-based features contain the most relevant information. Detection accuracies obtained with the full/selected features are around 72% for the seen *rāga*s test set. The corresponding results for the unseen *rāga* test set are 70.62% (full features) and 67.03% (selected features).

For *gamaka* classification, we use the same experimental set up as in the detection experiments. The classification accuracies for the seen *rāga*s are around 58% for the full and selected features. Accuracies for the unseen *rāga* are around 57% using full features, and around 64% using selected features.

Due to the limited amount of training data, we still observe an over-fit in our experiments. We would like to extend our work by constructing a larger dataset, for better modeling of *gamaka*s associated with notes.

6. References

[1] S K Subramanian, *Modeling Gamakas of Carnatic Music as A Synthesizer for Sparse Prescriptive Notation*, Ph.D. thesis, 2013.

[2] M Subramanian, "Analysis of gamakams of carnatic music using the computer," *Sangeet natak*, vol. 37, no. 1, pp. 26–47, 2002.

[3] M Subramanian, "Synthesizing carnatic music with a computer," *Journal of Sangeet Natak Akademi*, pp. 16–24, 1999.

[4] M Subramanian, "Carnatic music-automatic computer synthesis of gamakams," *Sangeet Natak*, vol. 43, no. 3, pp. 28–36, 2009.

[5] V S Viraraghavan, R Aravind, and H A Murthy, "A statistical analysis of gamakas in carnatic music," in *Proceedings of the 19th ISMIR Conference*, 2017.

[6] A Krishnaswamy, "Melodic atoms for transcribing carnatic music," in *Proceedings ISMIR*, 2004.

[7] T M Krishna and V Ishwar, "Karnātik music: Svara, gamaka, phraseology and rāga identity," in *Proceedings 2nd CompMusic Workshop*, 2012.

[8] S Dikshitar, *Sangita Sampradaya Pradarsini - English translation available at http://www. ibiblio. org/guruguha/ssp. htm*, 1904.

[9] S Giraldo and R Ramírez, "A machine learning approach to ornamentation modeling and synthesis in jazz guitar," *Journal of Mathematics and Music*, vol. 10, no. 2, pp. 107–126, 2016.

[10] W W Cohen, "Fast effective rule induction," in *Proceedings of the twelfth international conference on machine learning*, 1995, pp. 115–123.

[11] S I Giraldo et al., *Computational modelling of expressive music performance in jazz guitar: a machine learning approach*, Ph.D. thesis, 2016.

[12] H M Vyas, S M Suma, S G Koolagudi, and K R Guruprasad, "Identifying gamakas in carnatic music," in *Contemporary Computing (IC3), 2015 Eighth International Conference on*. IEEE, 2015, pp. 106–110.

[13] A Narayan and N Singh, "Detection of micro-tonal ornamentations in dhrupad using a dynamic programming approach," in *Proceedings of the 9th Conference on Interdisciplinary Musicology–CIM14. Berlin, Germany*, 2014, pp. 388–391.

[14] P Sinha, "Artificial composition: an experiment on indian music," *Journal of New Music Research*, vol. 37, no. 3, pp. 221–232, 2008.

[15] S Mohapatra, A Awasthi, and A Mukherjee, "Automatic music generation for indian classical music," .

[16] J Varadharajan, G Sridharan, V Natarajan, and R Sridhar, "Automatic synthesis of notes based on carnatic music raga characteristics," in *Advanced Computing, Networking and Informatics-Volume 1*, pp. 145–152. Springer, 2014.

[17] D Das and M Choudhury, "Finite state models for generation of hindustani classical music," in *Proceedings of International Symposium on Frontiers of Research in Speech and Music*, 2005.

[18] V Arora, L Behera, and P Sircar, "Singing voice synthesis for indian classical raga system," in *IET Irish Signals and Systems Conference (ISSC 2009)*, June 2009, pp. 1–6.

[19] F Pedregosa, G Varoquaux, A Gramfort, et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.