Computational Study of Ornaments in Hindustani Music

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science By Research
in
Exact Humanities

by

S Achyuth Narayan
201256001
sachyuth.narayan@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
February 2018

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled " Computational Study of Ornaments in Hindustani Music" by S Achyuth Narayan, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Prof. Navjyoti Singh

To My Parents

Acknowledgments

I thank Prof Navjyoti Singh for his constant support and guidance. Working under him was a great learning experience and my sincere gratitude to him for his patience and meetings, no matter what time of the day it was, he was always there to guide and motivate me endless times.

I would also like to thank Mr. Bahauddin Dagar for rudra veena performance annotations and insights into Dhrupad, Mrs. Saroja for her constant guidance and support and Mr. Irfan for his help in recording/annotating alankaaran on Sitar and his guidance on ornaments in Hindustani music.

I would also like to thank my parents for believing in me and motivating me always. You have always been there for me at every step of my life, thanks for your love and affection.

Abstract

Indian classical music is monophonic in nature. Melody is key to any performance. The entire performance is based on a single raga (melodic mode). Some performances are based on a coalescence of two or more ragas. Only a small duration in any musical performance in Indian art music is the pre-composed portion. The rest involves a gradual systematic exploration of the rāga using various forms of improvisation. The performer has great freedom in exploring his own improvisational patterns in order to delve into the intricacies of the given rāga. However the performer is bound by the framework of the rāga, tāla and other factors such as gharānā (the musical school which he belongs to), laya (tempo). There is a great variation in quality and nature of improvisation among different practitioners based on their cognitive and imaginative abilities. There are two important tools used in improvisation: alankār/alankāran (musical ornamentation) and tānas. In this thesis we restrict our study to the present day usage of alankāran [1].

Ornamentation is an indispensable part of improvisation. The performer tries to evoke the mood of the rāga (melodic-mode) with the delicate exploration of flourishes and ornamentation. Indian classical music is cognitively associated with these decorations instead of the actual underlying notes. We are interested in discovering the hidden musical patterns and structures in Indian music which are fundamental in unfolding of a raga. Recognizing these patterns can further help in music search, retrieval, recommendation and pedagogy. This thesis makes an attempt to study these ornaments computationally. We begin the study with the objective to computationally detect these ornaments in Dhrupad performances on rudrā veeṇā. Dhrupad is the oldest surviving form of Hindustani classical music, is monophonic in nature and has a wealth of ornamentation. We have crowd-sourced multimedia annotations of primary alankāra-s, namely, meeṇda (glides between notes), āndolana (vacillations in the note region) and gamaka (two notes alternation). These were used as the ground truth. Two signals in the time domain, a query signal, which consists of fundamental ornament templates, and a reference signal were compared. Using a variant of dynamic time warping, we then detect the matching subsequences in the reference signal with high accuracy. This method of iterative template matching is naive and has drawbacks, some of which are: 1.) Restrictiveness of the experiment to a set of non-exhaustive list of ornament templates, 2.) Computational overheads of comparing huge pitch vectors using dynamic time warping.

While DTW matching was an effective method of detecting these ornaments, we didnt use any domain specific knowledge to detect/identify these ornaments computationally. Cognitively rāgas are

[1] Most of these definitions are either explained in the introduction section or in the glossary at the end

associated with ornamentation.These along with the ways in which notes are approached,sustained and quitted constitute flows. We call them flows due to the constant change in the frequency and amplitude domains. The rate of change in frequency and amplitude domains together constitute the fundamental nature of ornamentation. We thus model ornamentation as first and second derivative of frequency and amplitude information to see whether the property of rate of change is fundamental to the idea of flows. We were able to achieve very high clustering accuracy by using this representation and our custom defined distance measure. We achieved an accuracy of 81% in distinguishing between Murki vs Kan and 84.5% in Āndolana vs Ghaseet. This again shows that transition is at the heart of the definition of flows and ornaments. However this doesnt account for hidden musical patterns which might not be captured by the manually engineered representation and custom distance.

It is important to note that ornaments and flows are intricate Indian music decorations which only experienced musicians can recognize and comprehend. In all of our previous experiments we have used small annotated datasets which do not exhaustively capture patterns underlying ornaments. The biggest challenge in learning and making sense of flows or ornaments is the unavailability of annotated music data. Thus, to have a big data driven exhaustive exploration of these melodic structures implies having either 1.) a multi performer crowdsourced effort to create a large exhaustive dataset, or 2.) an unsupervised/semi-supervised method of learning. To avoid the shortcomings of non-exhaustive small datasets and hand engineered representations, we also develop a robust unsupervised method to learn fixed length representations of ornaments. In the domain of speech processing, neural networks have recently successfully been used for encoding acoustic information into fixed length vectors. Existing research has shown that it is possible to transform audio segments into fixed dimensional vectors. In this thesis we extend the same model for learning fixed length features using sequence to sequence autoencoder to represent audio segments of ornaments and flows in musical data without human annotation (Curated dataset of 50 hours). We show that these learned audio representations capture essential sequential structural information and are useful in distinguishing between few different types of ornaments (recorded in isolation) in the vector space.

We started this study to have a big data exploration of ornaments in Indian music but we discovered that ornaments are not present in an isolated form in most Indian music performances. Thus we conclude that its important to study ornaments as a combination of multiple melodic structures. We have also discovered that some ornaments have similar musical properties as some others which is why their learned representations are closer in the vector space. Further, we have also noticed that alankār based on tāla and rhythm can very well be distinguished from the rest of the ornaments in the vector space. We finally discuss about how this model can be used for real time ornament transcription systems, ornament retrieval and tagging. This further has great applications in Indian music pedagogy, search, retrieval and recommendation.

Contents

List of Figures

List of Tables

Datasets Used

1. Sitar Sarod Dataset [SSD]: https://drive.google.com/open?id=0B2swnUYaKr7HeUlSNVdON21fTlE
2. Flows Dataset [FD]: https://drive.google.com/open?id=0B2swnUYaKr7HZEEtV20ycTIwNGs
3. Cluster Samples Studied [CSD]: https://drive.google.com/open?id=0B2swnUYaKr7HV1ZvX0tfaUhGX2s

Please note that the datasets are referred by the abbreviation mentioned above.

Chapter 1

Introduction

1.1 Introduction

Indian classical music is one of the oldest form of music which can be rooted to back to ancient scriptures, traditions and Vedas. It can be broadly classified into two major traditions: Hindustani classical music and Carnatic music. While Hindustani music was developed in North India, [5] Carnatic music evolved in the southern part of India. In this thesis, we restrict our study to Hindustani classical music. Tradition is an important part of Hindustani music and its elements are brought out in various musical performances [5]. Like Western classical music it divides the octave into 12 semitones. It uses something like just-intonation tuning instead of equal-temperament tuning system. This implies the intertonal gaps may vary.

Hindustani classical music is monophonic in nature. Melody is the key to any performance [31]. The entire performance is based on a single rāga (melodic mode). Some performances are based on a coalescence of two or more rāgas. *Puriya Dhanashree* is one such rāga, that has shades of multiple rāgas. Unlike Western music, there is no chord progression in Hindustani music [31]. In the next sections we describe in detail about the various features of Hindustani classical music, improvisation in Indian music, ornamentation and the overview of thesis.

1.2 Features of Hindustani Classical Music

1.2.1 Notes

Hindustani music is known for its complex use of microtones but for notation and explanation we divide an octave into 12 notes, they constitute a solfege known as *sargam* [36, 5]. These are similar to the twelve notes in western music. Table 1 lists out the notes and their western equivalents. However, originally octave was divided into 22 intervals called srutis, which were the smallest possible musical intervals. In addition to the 12 notes mentioned in Table 1 performers use microtones between the standard intervals.

Note	Hindustani	Western
Sa	shuddha (natural) S	C
re	komala (flat) r	D Flat
Re	shuddha (natural)	R
ga	komala (flat) g	E Flat
Ga	shuddha (natural) G	E
ma	shuddha (natural) m	F
Ma	teevra (sharp) M	F Sharp
Pa	shuddha (natural) P	G
dha	komala (flat) d	A Flat
Dha	shuddha (natural) D	A
ni	komala (flat) n	B Flat
Ni	shuddha (natural) N	B

Table 1.1 Notes in an Octave

1.2.2 Rāga, Tāla Framework

Rāga can be simply described as a collection of notes [5]. They provide the melodic framework out of which entire Hindustani musical performances are based [2]. All rāgas can further be grouped into 10 classes known as thāts [36]. A rāga can have any combination of 5,6 or 7 notes. For instance, a rāga can have 5 notes when ascending and 7 notes while descending a sargam. The jāti of the rāga defines the number of notes the rāga has in ascending and descending manner. When the notes are in ascending order we have Āroh and when they are in descending order we have Āvaroh. Vādi is the main note in the rāga, most of the performances set in a rāga revolve around this note. Samavādi is the second most important note in the rāga.

Since Hindustani music is monophonic and as performances are based on a fixed set of notes, improvisation becomes a quintessential part of a performance. Each rāga has a rasa (essence) associated with it. The artists main objective is to evoke this rasa of the rāga through delicate explorations of flourishes and ornamentation. The characteristics of the rāga are known as rāga-roopa [5].

Tāla is the metrical structure of a musical performance. It is any rhythmic beat or strike that measures musical time. Tāla is the rhythmic counterpart to rāga which together constitute the framework on which a performance in Hindustani music is set [5].

1.3 Improvisation in Indian Music

Improvisation is a fundamental part of Indian classical music. A performer has a great amount of freedom to introduce his own musical patterns in the gradual systematic exploration of the rāga [29]. This in turn brings out the *sthāyi bhava* of the rāga. There is a great variation in improvisation depending on various factors such as:

1. Intellectual skill of the artist
2. Emotional capabilities and mental conditioning of the artist
3. Inherent faculties of the artist
4. Gharāna (Background and training)

Every performer has a different amalgam of these factors which leads to a diverse and rich body of improvisational patterns. However, improvisation can be divided into two main types:

1. Planned Improvisation
2. Creative Improvisation

1.3.1 Planned Improvisation

In Hindustani classical music it is necessary to internalize pre-learned melodic phrases (that constitute a rāga) into the performance. This is mastered during the course of training with the guru. Over a period of time the performer develops a stock of musical expressions and phrases which he introduces into the performance to create a beautiful rendering of the rāga. This is known as planned improvisation [29]. The performer constantly rearranges these musical expressions, phrases, standard note combinations, ornaments that he has developed over years of practice. Each rendering of a rāga by a performer is different from the previous one since the rearrangement is completely spontaneous. Thus every performance is an improvised version, restricted by the framework of the rāga and tāla. The rearrangement of previously learned melodic phrases may take various forms:

1. Entire musical phrases may be rearranged. For instance, once the performer has learned the phrases 1, 2, 3 and 4. He may develop many permutations such as 1234, 2341, 3412 etc to present them. We will use this example in the following explanations.
2. Entire musical phrases may be arranged but with the judicious introduction of repetitions. In the current example, the performer might start to present the phrases: 12341, 342424 etc to get a wider range of improvisation
3. If a musical phrase can be subdivided into multiple small phrases the performer might decide to make permutations and combinations with the smaller phrases instead of the whole phrases. (with or without repetitions). Considering our previous example, if 1 can be broken down into 1a and 1b, 2 can be broken down into 2a , 2b and 2c, the performer might decide to present the phrase 1a2b34 or 1b2a2a2b1a etc. This again leads to much wider range of improvisation
4. The performer might adopt different ways of executing phrases, such as varying dynamics (volume level), applying different ornamentation, sustaining on certain notes for a longer duration etc.

5. The performer may even create contrasts by performing a phrase and, immediately thereafter, either repeating it an octave lower or higher.

1.3.2 Creative Improvisation

The form of improvisation which transcends pre-learnt melodic phrases is known as creative improvisation [29]. The performer delves into new possibilities of the rāga, which go beyond the permutations and combinations of the pre-learnt material. Without creative improvisation music would be stagnant. Creative improvisation accounts for the free manifestation of originality and creativity of the musical mind. It is important to note here that the deviations from the orthodox structure of the rāga to internalize ones creativity into the performance can be aesthetically and structurally pleasing only after establishing the fundamental personality of the rāga (rāga roopa). Otherwise such originality/creativity may be perceived as waywardness.

1.4 Techniques of Improvisation

Although improvisation in a musical performance results from the unique personality and varies significantly there are a certain well defined methods through which it can be achieved. These methods can be broadly classified into two main categories: [29]

1. Improvisation related to the composition
2. Improvisation free of the constraints of the composition

1.4.1 Composition related Improvisation

In a *bandish* (composition) based on a certain rāga, there is a fixed structure and fixed set of configurations. In this kind of improvisation, variations can be made by altering some areas of configurations while keeping the fixed structure intact. These areas need to be such that they dont disturb the characterization of the rāga to which the composition is set [29]. An effort needs to be made to make sure that the aesthetic state of the rāga is not disturbed. There are various ways to achieve this, the chief ones are:

1. By varying note lengths
2. By substituting notes
3. By varying direction
4. By varying order of notes
5. By varying ornaments
6. By varying rhythmic patterns

7. By varying timbre and vowels

8. By varying volume

1.4.2 Free Improvisation

It is important to note that in a Hindustani musical performance the *bandish* (composition) is sung only once and doesnt constitute the majority of the performance. The rest of the performance consists of improvisation. The various methods outlined in 1.3.1 apply here too. The primary requirement in this form of improvisation is that it must adhere to the framework of the rāga and the tāla. Free improvisation further consists of 1.) Alap/Vistār, 2.) Layabānt, and 3.) Tāna

Alāp is the tāla-less elaboration of the rāga (vistaar) which enunciates the rāga-roopa (characteristics of the rāga). It is the introductory portion of the recital. Beginning with a slow elaboration of the rāga through systematic resting upon important notes/phrases, it gradually gains momentum and speed in enunciating the rāga-roopa. It is thus a performance in itself. Layabānt is the distribution of the time cycle into various rhythmic sections and subsections by the performer. Tānas are fast passages of the characteristic notes of the rāga arranged in various attractive configurations which make for aesthetically pleasant listening.

There are various methods of employing free improvisation as described below:

1. Slow and systematic progression of notes to bring out the character of the rāga (alaap)

2. Sustaining on particular notes

3. Adding short ornamental phrases before/after the sustained note

4. Executing various kinds of Tānas etc

This is just a brief list, there are other methods of employing free improvisation including those mentioned in section 1.3.1. In this thesis we are mainly interested in studying the ornamental phrases added to improve the aesthetic appeal of the performance. In section 1.4 we in detail explore the different types of alankāran (ornamentation)

1.5 Alankāran

Using alankāran is a very important and age old method of improvisation. Before we explore the different types of alankāran employed by performers of Hindustani classical music it is important to note the difference between alankār and alankāran. Although alankār and alankāran are used synonymously these days, the original definition of alankār is a specific group of notes or group of varṇas. A varṇa is a short group of notes that gets its meaning in the context of a musical phrase. A varṇa has no meaning outside of the musical phrase. For example, the note group S G R has no meaning of its own, but it is

capable of acquiring a meaning in context of a musical phrase. Alankāran on the other hand means an act of ornamentation. There are several modes of achieving alankāran [29], among which the chief are:

1. A set of alankārs may be used in unison to achieve alankāran.

2. *Gamak*: An important way in which ornamentation is achieved is through gamak. Gamak is a certain swaying or oscillation of the pitch of a note so that it goes slightly off its true pitch for a moment and returns to its original pitch immediately.

3. *Sthaya*: Sthaya is a combination of notes with reference to a particular rāga such that the combination exposes a particular bhava of the rāga. It is used to express the emotional content of the rāga. For instance the rāga Bhoopali has a *shanta bhava* (mood of tranquility) associated with it, a note phrase such as (G R S S) which is sung in a certain way to bring out the bhāva is known as *sthaya*.

4. *Kaku*: Kaku refers to the manner of articulating a particular note. This involves: different ways in which notes are approached, sustained and quitted, the volume/vigour with which the note is executed, the timbre used in the utterance etc.

We can also group the different types of alankāran as varṇalankār, Chhandalankār, varṇatiriktālankar and Vadanbheda [29] [5].

1.5.1 varṇalankār

These are Alankārs using varṇas or a group of notes. They can be grouped into:

1. Sthayi - These are alankārs where each varṇa ends on the same note with which it begins, so that there is no tendency towards either ascent or descent

2. Ārohi - These are alankārs that show an ascending trend that is to say, the last note of an ārohi alankār is always higher than its first note

3. Āvarohi - These are alankārs that show a descending trend that is to say, the last note of an ārohi alankār is always lower than its first note

4. Sanchāri - These are alankārs that do not have a fixed direction - ascent or descent throughout, but are of mixed nature. Sangeet Ratnakar gives a list of twenty five such alankārs.

1.5.2 Chhandalankār

When varṇas or note-groups are used in various rhythmic patterns or figures they constitute Chhandalankār. They are further divided into:

1. Tāla-samparkit: If they are related to a tāla or a specific time cycle, they may be called tāla-samparkit.

2. Tāla-asamparkit: However, if there is no connection with any tāla, chhandaalankār may be said to be tāla-asamparkit. If a four-note varṇa SRGM is executed in different ways within the same time duration, by creating new figures each time, then they belong to this category.

1.5.3 Varṇatiriktālankar

When alankāran or ornamentation is achieved without using varṇas but by creating variations in the manner of articulating notes, it is known as varṇatiriktālankar. That is to say, when different characteristics of the voice are brought into play, such as varying the pitch or volume in different ways, and thereby alankār is brought about, we get varṇatiriktālankār. These are also called shabdalankār (shabda is translated as sound), as sound is altered instead of notes.

Dr. Bimal Roy classifies them into:

1. Vishishtālankār - This type of alankār arises when the sound involved in uttering two notes is varied. Some examples are:

   (a) Khatak: A descending jump or skip of two, three or four notes.

   (b) Atak: An ascending jump or skip of two, three or four notes.

   (c) Krintan: A grace note followed by a glide (glissando).

   (d) Jhatkā: This arises when one note is pushed to another note at least a third above.

   (e) Sparsh: A note of a very short duration in time in a stepwise ascent involving two notes.

   (f) Kaṇ: A short note preceding the main note.

   (g) Zamzamā: This occurs when two notes which are a step away are alternated.

2. Varṇavyatiriktālankār - If the sound altered involves more than two notes, we get varṇavyatiriktālankār.Some examples are:

   (a) Ghaseet: A rapid glide (glissando) on a fretless instrument (Eg: Sarod)

   (b) Āns: It occurs when a note is played in a way to glide over to another note (atleast two notes away) so as to touch all intermediate notes in the process.

   (c) Meeṇd: A glide from one note to another which can range from a simple span of 2 notes to a whole octave.

   (d) Murki: It is an alankār using three notes articulated in the phrase of four such that the first and last notes are the same.

(e) Āndolan: The Āndolan alankār is a gentle swing or oscillation that starts from a fixed note and touches the periphery of an adjacent note. In the course of the oscillation, it touches the microtones or shrutis that exist in between.

(f) Gitkiri: Its movement is faster than that of murki and it also uses 3 notes in a phrase of four but unlike murki the first and the last notes are not the same.

3. Sthaayakaaku - When the kaaku employed in the utterance of a single note of a sthaya is considered in isolation, we get sthayakaaku. It refers to variation in inflection of a single note used either on its own or in conjunction with other notes to portray moods of the rāga. Some examples are:

   (a) Aakar: Articulation of a note with an open mouth.

   (b) Bandhakar: Articulation of note with lips closed.

   (c) Prabala: Loud articulation of a note.

   (d) Prabalikaran: Articulation which gets gradually louder.

   (e) Mridu: Soft articulation

1.5.4 Vadanbheda

Some alankārs are best explained with the help of stringed instruments. These have been generally classified under the head of Vadanbheda. This consists of 4 subgroups:

1. Ahobal's Vadanbheda - a list given in Sangeet Parijat of ahobal. Some examples include:

   (a) Chyavita: The da sound of a sitar or veena which consists of a sudden and sharp increase in volume.

   (b) Kampita: The dra sound of a sitar or veena. It consists of two strokes of the plectrum on the string, one following the other in quick succession.

   (c) Pratyahata: This occurs when two stopped notes are produced with a single stroke of the plectrum.

   (d) Sfurita: If two successive plectrum strokes are played louder in relation to the succeeding and preceding notes then they are said to have utilised the sfurita alankār.

2. Somnath's Vadanbheda - a list given in rāga Vibodha of Somnath.Some examples:

   (a) Pratihati: It is also called Krintan-Sparsh since it is a combination of these two using two strokes of plectrum.

   (b) Anuhati: Same as pratihati except the string is vibrated by the nail (instead of plectrum) only once.

(c) Aahati: Notes are produced by the stopping fingers hitting the stationary string against the frets.

(d) Peeda: In a single plectrum stroke three notes are produced.

3. Gamakabheda - from oral tradition. There are different kinds of gamak explained by means of string instruments. Some examples are:

   (a) Sphurita Gamak: This is achieved by lowering the tension of the string by sliding it along the fret from a high note to the next lower note.

   (b) Murki Gamak: This is achieved by pulling the string to a high note in a long glide and then releasing the tension via a short glide touching the intermediate notes.

   (c) Gitkiri Gamak: The string is pulled to achieve gitkiri (explained in 1.4.3) in one glide.

   (d) Pratyagata Gamak: The string is pulled along the fret to a note a third higher and the released to sound the note in between

4. Meeṇdabheda - from oral tradition. While both gamak and meeṇd involve the sliding of the string along a fret, the glide in a gamak is quick while that in a meeṇd is slow. Examples:

   (a) Gitkiri meeṇd: A single glide on the fret. Ex: G R S R on the S fret.

   (b) Murki meeṇd: A murki executed in a meeṇd.

   (c) Ghaseet meeṇd: A meeṇd on a single fret.

   (d) Saada meeṇd: An ordinary meeṇd.

1.6 Motivation

Analysis of improvisation in Indian music is a broad research topic that can be approached from different academic disciplines. In this thesis, we take a data-driven engineering approach and focus solely on the computational aspects of ornamentation analysis. The research methodology used is at the intersection of signal processing and machine learning. In the previous section we have explored in detail the various types of alankāran and their descriptions. Alankāran (Ornamentation) is at the heart of Hindustani musical performance. We have noticed that even though there is a large variation in the types of alankāran there are some fundamental characteristics upon which most of them can be differentiated. The duration of notes used, the intervals/micro-tones used the amplitude of the notes used. These are the simple properties of a sound wave based on which this complex diverse set of ornamentation can be segmented. However we have also noticed that improvisation heavily depends on cognitive and imaginative resources of the performer which adds to the diversity of the superset of all types of ornamentation. Artists add as much variety as possible to alankāran within the periphery defined by the rāga framework. In figure 1.1 we show three characteristic melodic phrases of rāga *Alahaiya Bilaval*

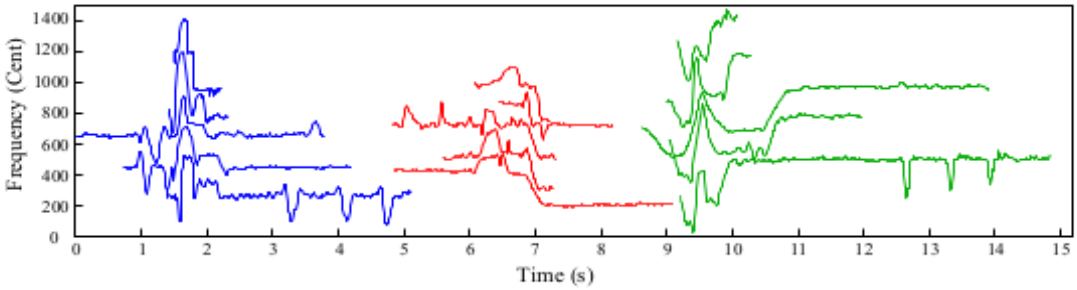

Figure 1.1 Pitch contours of three different characteristic melodic phrases [17]

adjusted in time/frequency. We can clearly see that the melodic regions are highly varied in terms of the duration and presence of ornaments. There are a lot of vocal expressions that are introduced by a performer for a richer rendition which are not taken into account and there arent any pre-existent score systems for Indian classical music which can be used as a source of verification. This poses a huge challenge for detection/classification and characterization of ornaments. The objective of the thesis is to explore the relevance of signal based features in representing these ornaments and distinguishing them from one another. This would lead to a better computational understanding of ornaments which in turn has various applications.

Ornaments are the signature motifs of a performer in a typical Hindustani classical music performance, skills for which are developed through a long and arduous training regimen. It is important to study alankāran because 1.) they constitute a significant portion of the performance , 2.) they reflect on the performers background, training and emotional intellect, 3.) they form the characteristic features of a melody and are representative of melodies, 4.) it helps create a knowledge base of different types of ornaments being used by which, we can create a strong pedagogical tool for performers to explore new frontiers in ornamentation. It is important to study and represent these ornaments computationally as it can further be used in rāga recognition/classification, music search, recommendation and retrieval. Music search and retrieval can be made semantically more relevant with the understanding of ornamentation computationally. For instance, performances in Hindustani classical music can be indexed on the types of ornaments being used instead of some metadata about the performance. As pointed out earlier ornamentation is developed over large number of training years. By understanding ornamentation computationally we can build a pedagogical tool which might simplify the advanced nature of these motifs for beginners. Thus, in this thesis an attempt is made to detect/identify these ornaments in musical performances, explore the relevance of rate of change in the frequency/amplitude domains for ornamentation, and finally build a scalable model to represent/distinguish between alankāran considering the dearth of annotated datasets available.

1.7 Contributions and Scope of the Thesis

As discussed in section 1.1, Indian classical music is divided into Hindustani Classical music and Carnatic music. Although ornamentation is prevalent in Carnatic music as well, we restrict our study to Hindustani Classical music. We have also restricted our study to Hindustani classical music performances on Rudra-veena, Sitar and Sarod. In brief, the main objectives of our thesis are :

- To develop a methodology to mine alankāran in large audio collections comprising of hundreds of hours of data
- To develop an approach that can exploit domain-specific knowledge to effectively characterize alankāran
- To devise a methodology to learn semantically relevant features to distinguish between structurally similar and structurally varying alankāran. HoWever in this thesis we just differentiate between (Murki, Kaṇ) and (Āndolan, Ghaseet). Some types of alankāran are very similar to each other thus they can't be distinguished from each other.

To summarize the thesis we begin the study with the objective to computationally detect ornaments (alankāran) in Dhrupad performances on rudra veena. Dhrupad is the oldest surviving form of Hindustani classical music, is monophonic in nature and has a wealth of ornamentation. We have crowdsourced multimedia annotations of primary alankārs, namely, meeṇdA (glides between notes), Āndolan (vacillations in the note region) and gamaka (two notes alternation). These were used as the ground truth. Then we perform iterative template matching to compare the template and reference signal and assign segments labels based on the matching template with least cost. We use a variant of dynamic time warping to detect the matching subsequences in the reference signal with high accuracy.

The characteristics of ornamentation vary a lot across music traditions. Characterization of the melodic patterns should thus be studied within the context of a specific music tradition. We aim to develop an approach that can exploit domain-specific knowledge to effectively characterize alankāran. Due to the non-exhaustive dataset at our disposal we aimed to build a generic framework to distinguish between ornaments. Thus, we built a model engineered on domain specific knowledge. We model ornamentation as first and second derivative of frequency and amplitude information to show that the property of rate of change is fundamental to the idea of ornaments. We used a dataset of annotated ornaments in sitar performances for this experiment. We were able to achieve very high clustering accuracy by using this representation and our custom defined distance measure. We achieved an accuracy of 81% in distinguishing between Murki vs Kaṇ and 84.5% in Andolan vs Ghaseet. Finally, we develop a robust unsupervised method to learn fixed length representations of ornaments. Existing research has shown that it is possible to transform audio segments into fixed dimensional vectors. In this thesis we extend the same model for learning fixed length features using sequence to sequence autoencoder to represent audio segments of ornaments and flows in musical data without human annotation (Curated dataset of

50 hours consisting both sitar and sarod performances).We show that these learned audio representations capture essential sequential structural information and are useful in distinguishing between few different types of ornaments (recorded in isolation) in the vector space. We finally discuss about how this model can be used for real time ornament transcription systems, ornament retrieval and tagging.

1.8 Overview

The thesis is organized as follows:

- Chapter 2 describes the state of art research in the study or ornamentation.
- Chapter 3 describes the experiment to computationally detect ornaments (alankāran) in Dhrupad performances on rudra veena. The complete procedural details and results are also given (published) in [31].
- Chapter 4 describes how we model ornamentation as first and second derivative of frequency and amplitude information to show that the property of rate of change is fundamental to the idea of ornaments. The complete procedural details and results are also given (published) in [33].
- Chapter 5 describes the model for learning fixed length features using sequence to sequence autoencoder to represent audio segments of ornaments and flows in musical data. We show that these learned audio representations capture essential sequential structural information and are useful in distinguishing between different types of ornaments in the vector space.
- Chapter 6 has the conclusions and further work which can be done to truly realize the potential of what has been proposed. This chapter explains how this work can be extended into building truly intelligent and scalable transcription, search/retrieval engines.

Chapter 2

State of Art in Computational Study of Ornaments

2.1 Introduction

In the previous chapter we have described the background of Indian music, improvisation and alankāran. In this chapter we give our review of the existing literature which adopt a computational methodology to study ornamentation, melodic motifs and pattern mining in music. We begin with the review of state of art in melody pattern processing in Indian music. In the second part we discuss work related to sequence to sequence auto-encoders in speech and its applications for our use case in Indian music.

2.2 Melody Pattern Processing in Indian Music

Analysis of melodic patterns in Indian music is a well studied discipline and has gained a lot of attention in the recent years [18, 25, 19, 42, 31, 32]. However the study of these melodic patterns is not just restricted to a certain types of ornamentation (alankār). This kind of study can be divided into two main categories:

- Pattern Classification (Supervised task)
- Pattern Discovery
  - Supervised Approach (Ground Truth Annotations)
  - Unsupervised Approach (No Annotations)

Pattern discovery further can be classified into two different approaches: Supervised and Unsupervised. In the supervised approach it is possible to easily verify whether the discovered melodic patterns are relevant due to the available ground truth annotations. However it is difficult to verify whether the discovered patterns are relevant in the case of unsupervised approach. Pattern discovery is a lot more computationally challenging compared to classification, since a brute force search for melodic patterns

Authors	Description	Representation	Segmentation	Similarity
Pratyush (2010) [36]	Distinction between ornaments	Pitch detection, Octave error removal	Manual Segmentation	DTW, Auto correlation
C.Gupta, P.Rao et al. (2011) [19]	Objective assessment of ornamentation	Continuous Representation - Pitch detection followed by conversion to cent scale	Manual Segmentation	point to point error calculation; Dynamic time warping and polynomial curve fit based matching
Ishwar et al. (2012) [23]	Distinction between patterns	Continuous Pitch Representation	Manual Segmentation	HMM
Ross, Rao et al. (2012) [40]	Detection of Melodious motifs	Continuous Pitch Representation	Manual Segmentation	DTW
Ross et al. (2012) [25]	Detection of Melodic Patterns	Continuous, SAX-12,1200	Manual Segmentation	Euclidean, DTW
Ishwar et al. (2013) [24]	Detection of Melodic Patterns	Stationary Point Representation	Brute Force	Rough Longest Common Subsequence
Miryala et al. (2013) [42]	Automatically identification of Melodic Patterns	Pitch-curve in terms of straight lines and critical points of inflection	Entropy based Segmentation	Similar to DTW
Dutta, Murthy (2014) [9]	Discovery of Patterns	Stationary Point/Continuous Representation	Brute Force	Rough Longest Common Subsequence
Dutta, Murthy (2014) [10]	Discovery of Patterns	Stationary Point/Continuous Representation	-	Modified Rough Longest Common Subsequence
Rao et al. (2014) [37]	Distinction between patterns	Continuous Pitch Representation	Manual Segmentation	DTW, HMM
Ganguli et al. (2015) [15]	Detection of Patterns	BSS, Transcription	-	Smith-Waterman
Gulati (2016) [17]	Mining melodic patterns	Melodia Pitch Tracking, Error corrected Pitch	-	DTW

Table 2.1 Summary of relevant literature in study of Indian art music.

in high dimensional audio vectors takes a lot of time. Audio data is high dimensional data and even the pitch vectors extracted from this data are extremely large adding to the computational overheads [31].

Most of the relevant literature take the supervised approach of pattern distinction and discovery [37, 10, 15, 17, 40, 24]. However most of the approaches are worked on different datasets which makes it difficult to directly compare their strengths and shortcomings. A common theme across all the approaches is that they are focused on characterizing melodic patterns in such a way so as to arrive at a notion of melodic similarity. This can further be split into three major tasks:

- Melody Representation (Feature Extraction)
- Segmentation
- Similarity Measure

As discussed in chapter 1 there is a great variation in melodic patterns in terms of pitch, amplitude information and timing. These can either be accounted for in the melody representation stage or while coming up with a similarity measure.

We first review the approaches presented in relevant literature in terms of these three tasks.

2.2.1 Review of Representation Methods

Performances in Indian music revolve around a melody line and the transitory regions between notes and micro-tones are very important. This poses a challenge to retrieve discrete melodic representations. Most of the approaches in the relevant literature advocate the use of continuous melodic representations instead of discrete representations. For instance, pitch tracking algorithms have been used by Pratyush (2010) [36] and Gulati et al. (2016), [17] to extract relevant melody representations. Ganguli et al., (2015) [15] and Ross et al., (2012) [25] abstract the melodic representation by using behavioral symbolic sequence (BSS) and symbolic aggregate approximation (SAX). Ganguli et al. (2015) provide a method of obtaining a discrete melody representation using a heuristic based pseudo melody transcription method. Miryala et al. (2013) [42] proposed an algorithm for automatic transcription, using line fitting to obtain canonical representation of pitch curves in terms of straight lines and points of inflection and then using template matching to identify vocal expressions. However, the algorithm often fails to distinguish between steady notes and glides. Ishwar et al. (2013) [24], Dutta et al. (2014) [9] provide abstract melody representations with the use of stationary points. These abstract representations reduce the computational complexity by a great factor. However there accuracy isn't comparable to the continuous melody representations. It has also been observed that these discrete melody representations have been tested on very small datasets and thus its applicability to large datasets with wider variety is questionable. Thus we conclude that the continuous representations are more valuable and accurate in distinguishing between melodic patterns even though they increase the computational complexity by a great margin.

2.2.2 Review of Segmentation Methods

Temporal segmentation based on events in music is important for pattern detection since it provides the potential areas of detection. As described in table 2.1 most of the approaches in the relevant literature either use a brute force segmentation strategy or have ground truth annotations or use a local alignment based distance measure strategies. This is a very challenging task because melodic patterns are hard to recognize. Only trained musicians will be able to detect these with ease. Miryala et al (2013) [42] use the difference between high and low energy frames to figure out relevant segments. While this is an effective method, some audio segments might be obtained which do not contain any significant musical patterns. Ross et al. (2012) [25] detect *sama* locations as landmarks to determine the location of melodic patterns. Ross and Rao et al [40] detect *Pa nyas* as landmarks to detect potential melodic patterns. These approaches are limited in their applicability to a certain style and are not exhaustive. They cannot be generalized to different types of musical datasets.

2.2.3 Review of Melodic Similarity Measures

Melodic Similarity is an important phase of pattern discovery or classification as it is the foundation on which patterns are compared. Gupta et al. (2011), [19] propose objective methods to assess the quality of ornamentation in Indian music by a singer by taking in account an ideal singer as a model or reference. They compare reference meend-s with the meend-s sung by different singers on the basis of (i) point to point error calculation; (ii) Dynamic time warping and (iii) polynomial curve fit based matching. This work while dealing with the quality of ornaments gives valuable insights into the methods that can be used for building an ornament-transcription system. Ishwar et al. (2012), [23] use Hidden Markov Models (HMM) to perform pattern classification and show promising results. Ross and Rao et al. (2012) [40], Ross et al. (2012, 2013, 2014) [25, 37] use variants of Dynamic Time Warping (DTW) [30] for melodic similarity distance. However Ishwar et al. (2013) [24] and Dutta et al. (2014) [9] use rough longest common sub-sequence (RLCS) to characterize melodic similarity. Dutta et al. (2014) [10], also provide with an improvement to the existing RLCS to improve the precision of the system. Recently Ganguli et al. (2015) [15] use Smith-Waterman algorithm to compute melodic similarity (Smith and Waterman, 1981) [46].

Euclidean distance might not be a good choice for melodic similarity since it doesn't measure any temporal alignment. Ross et al. (2012), [25] propose various similarity metrics to detect melodic motifs in Hindustani music. Along with the conventional DTW they also use piece-wise aggregate approximation to convert a non uniform length time series to a uniform length dimension reduced sequence of symbols. A given time series is aggregated into uniform W length sequences and Euclidean distance is used as the similarity measure. Gulati et al. (2016) [17], develops computational approaches for analyzing high-level melodic aspects of music performances in Indian art music. For extracting melodic features he uses the Melodia pitch tracking algorithm and ignores the energy and velocity information. He then uses normalized and error corrected pitch tracked information along with DTW for mining pat-

terns. To identify musically meaningful patterns, he exploits the relationships between the discovered patterns by performing a network analysis, exploiting the topological properties of the network.

From Table 2.1, we can see that most of the approaches use a dynamic programming based approach to characterize similarity between melodic segments. The timing variations introduced by performers of Hindustani music is the biggest reason for the relevance of dynamic programming adopted by most approaches. While the transition of a melodic segment might be between the same notes the trajectory they take to achieve it might be varied, in terms of pitch, amplitude and timing.

Since most approaches use dynamic programming computational overheads are a big concern. Since most approaches use small datasets this issue wasn't addressed. However to build scalable pattern detection/discovery systems this is major shortcoming. Using abstract representations might reduce the computational complexity but will hinder the accuracy of the system. However a two stage approach can be used to optimize this procedure. As proposed by Ishwar et al. (2013) and Dutta et al. (2014) first use an abstract representation to detect regions which might contain relevant patterns and then a finer representation can be used to detect patterns in these regions. This reduces computational complexity by a great factor.

Its important to take note that most of the approaches operate on very small datasets which make their relevance and applicability to varied and large datasets questionable. Also most of the approaches are worked on different datasets which makes it difficult to directly compare their strengths and shortcomings. Thus we conclude that there is great scope of improving the generic pattern processing capabilities of existing systems which are very restrictive and constrained.

2.3 Auto-encoders in Speech Processing and its Applications

The biggest challenge in learning and making sense of ornaments is the unavailability of annotated music data. Thus, to have a big data driven exploration of these melodic structures implies having an unsupervised or semi-supervised method of learning to reduce the annotation effort.

Also, raw audio data is extremely large in size and any information retrieval/search or machine learning task on this data leads to extremely large computational overheads and often not very accurate results. Linear classifiers built on top of such high dimensional raw data usually dont tend to learn much about the structure of the underlying data. By representing the audio segments in a compressed fixed length vector format we can reduce the computational overheads by a significant margin. We can reduce the cost of indexing, cost of retrieval and other computational overheads in various machine learning experiments.

Word2Vec [48, 27] which transforms each word in a given text into a vector of fixed length has been proven to be very useful in various applications in natural language processing. These vectors were shown to carry a lot of semantic information about the words in consideration.

Following this, in the domain of speech processing, neural networks have been recently successfully been used for encoding acoustic information into fixed length vectors [4]. Existing research has shown

that it is possible to transform audio segments into fixed dimensional vectors. Word audio segments with similar phonetic structures were closely located in the vector space created by the transformed fixed dimensional vectors. Recently Chung et. al., [49] have developed an unsupervised approach to produce fixed length vector representations for variable audio segment data using sequence to sequence Autoencoders. These vectors were shown to describe the sequential phonetic structures of the audio segments to a good degree, with real world speech applications such as query-by-example Spoken Term Detection. They have even outperformed conventional approaches such as DTW with much lesser computational overheads.

In the generic framework of a sequence to sequence autoencoder there is an RNN to encode the input audio sequence into a fixed dimensional vector and there is an RNN to decode the fixed dimensional vector to reconstruct the original audio input sequence [49, 3]. This approach has been used in text processing, video processing and more recently in signal processing for speech applications. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. This makes them suitable for capturing sequential information since the outputs depend on previous computations. In this thesis, we try to extend the same model for learning fixed length features using sequence to sequence autoencoder to represent audio segments of ornaments and flows in musical data without human annotation (Curated dataset of approx. 50 hours). We expect that these learned audio representations capture essential sequential structural information, as the rate of change/sequence of changes in amplitude and frequency is the underlying foundation of ornaments. We here expect that similar ornaments and flows would be nearby in the vector space created by the fixed length representations. This is completely an unsupervised task since we dont need any annotations to build the fixed length representations

Neural Network approaches however are very data hungry and they do not learn anything useful if there is a low resource dataset at hand. We thus build a huge dataset with around 50 hours of Sitar Hindustani musical performances. More about this is explained in Chapter 5.

Chapter 3

Detection of Alankaaran in Dhrupad

In the previous chapter we have seen that most of the existing approaches have used dynamic programming approaches to identify melodic patterns. Most of these were published during the course of this dissertation. In this chapter we use a variant of Dynamic Time Warping, (extensively used in speech processing applications for pattern matching) to detect a few specific types of alankāran. We restrict our study to a small dataset of annotated alankāran in rudra-veena performances. We then discuss the strengths and challenges of this approach before we move on to building a more general purpose approach for Indian music.

3.1 Introduction and Background

Although a lot of techniques for studying ornaments have been developed for Indian Classical music in general, not much work has been done, distinguishing Dhrupad as a separate form of Hindustani Classical music. Dhrupad is the oldest surviving form of Hindustani classical music and is monophonic in nature [43]. It is a complex musical genre with many elements such as style and ideology which are formed and transmitted with tradition. Tradition is of very high importance to Dhrupad artists and they try to bring out its elements with the help of performances. A performance of Dhrupad involves a re-creation of traditional elements which are gathered, re-arranged and assembled spontaneously [43]. The study of these performances hence involves a complex study of improvisation. We hope to experiment with a large variety of alankār, thus we chose Dhrupad for our study. In this study, we divide the process of detection into 5 stages as shown in figure 3.1.

In Dhrupad, as in any other form of Indian classical music, melody is the key to any performance. The entire performance is based on a single rāga (melodic mode). Some performances are based on a coalescence of two or more rāgas. *Puriya Dhanashree* is one such rāga we examined, that has shades of multiple rāgas. *Puriya Dhanashree* has most of the lakṣaṇa-s (characteristics) of Puriya leaving the tivra madhyam which replaces the shudhha madhyam [43].

Any performance begins with the artist delving into the melodic improvisation of any given rāga (melodic mode). This melodic exposition (ālap) introduces the rāga and is unmetered and unaccompa-

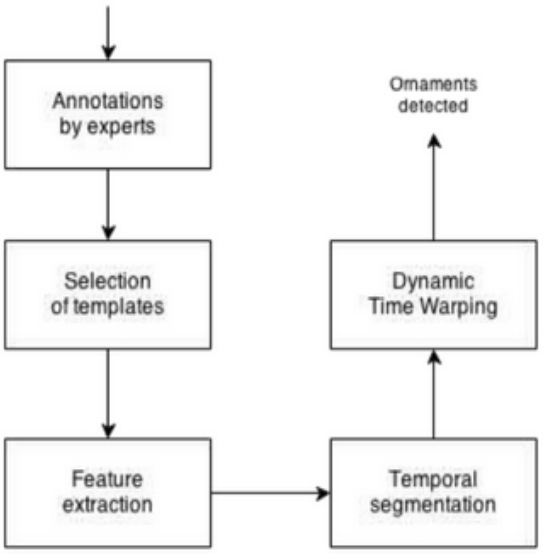

Figure 3.1 Procedure for Detection of alankār

nied by any instrument. This is usually the longest part of the performance [43]. The tempo gradually increases and then one goes into the composition (bandish). The composition is set in the same rāga and it is accompanied by a *pakhawaj* (Indian musical instrument). In our study, we have considered only instrumental performances (rudra veena) without any pakhawaj accompaniment. In instrumental music, Ālap consists of three stages which constitute the entire performance. They are: Ālap, Jor and Jhāla. The Ālap is considered to be rhythm free, jor has a regular rhythmic pulse and jhāla features very rapid rhythmic patterns. Jhāla is usually the finishing part of the performance. These stages are different for vocal music [43].

A performance in Dhrupad has a wealth of ornamentation, as there is no chord progression and it is based on a single melodic line. Our main purpose of this study is to detect ornamentation. For an effective music transcription system it is essential that these ornaments are detected appropriately. It is very difficult to identify these ornaments because, they are dependent on the performer and they consist of various micro-tones which are very feeble. Hence these are not easily recognized by the modern day systems. In our study, we have considered three main types of alankār-s present in most of the Dhrupad performances for our study. They are: Meeṇd, Āndolana and Gamaka. Meeṇd refers to the glissando or the glide from one note to the other, Āndolana is a gentle oscillation on specific notes that starts on a specific note and touches the boundaries of other notes and gamaka is a rapid to and fro oscillation between distinct notes [43]. We have further classified meeṇd into two types for detection purposes as: Basic meeṇd-s and meeṇd-s resting on other intermediate notes. Long meeṇd-s are one of the characteristics of Dhrupad. Given a rāga, we can reduce the total number of possible ornamentation into a smaller abstract set based on the vadi and samvadi, and Aroha and Avaroha. This reduction is also possible, partly because of the repetitive nature of Āndolana and gamaka. For every performance we

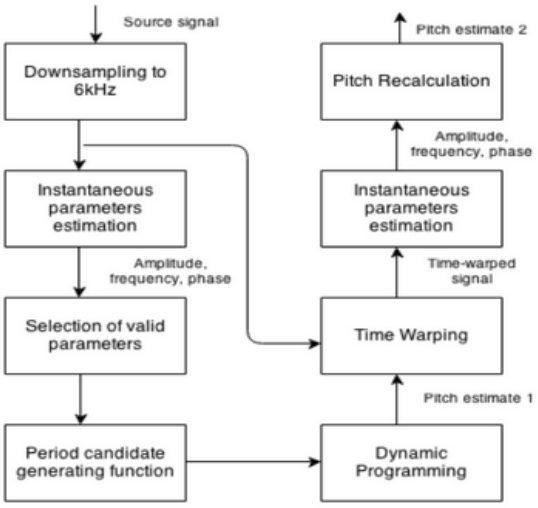

Figure 2: Various stages of Pitch Estimation [4]

Figure 3.2 Various stages of Pitch Estimation

have selected some standard set of alankār-s as templates (pitch vectors) for detection. Since the crowd sourced database and the selected performances did not consist of many occurrences of gamaka-s, we dont have substantial validation for the detection of gamaka-s using the time warping technique.

3.2 Feature Extraction for Ornamentation

Melody is a time domain sequence of discrete notes. Each note essentially corresponds to a fundamental frequency (pitch). So we extract pitch vectors from the given audio samples and these will be our feature vectors. We choose pitch vectors of certain alankār-s as our fundamental ornament templates and then we perform time domain based matching. For detecting ornaments it is also essential to identify the boundaries in which ornaments have to be searched (onset detection).

Azarov, E. et al., [1] provide a comparative study of various existing pitch tracking algorithms and we find that IRAPT, is the most suitable algorithm for our study. IRAPT is an instantaneous pitch estimation algorithm and what sets it apart from the other existing pitch tracking algorithms is its insensitivity towards rapid frequency modulations and accurate values with high frequency resolution. IRAPT uses a period candidate generating function based on instantaneous harmonic parameters and it has an additional time warping stage which makes the estimation very accurate. The various stages involved in this pitch estimation are shown in the figure 3.2.

3.3 Segmentation for Ornamentation

Now using the pitch vectors we compare the time domain signal of various ornament templates and the time domain signal of the actual voice recording. To perform dynamic time warping, we need to first determine the potential areas in which the ornament can be detected. Since we consider only rudra veena performances we use an onset based approach.

Temporal segmentation based on events in music is important for our analysis since it gives us the potential areas of detection. Whenever a string of the rudra veena is plucked there is a sudden burst in energy. This stark change in energy is considered to be an onset and we detect these onsets in order to segment the audio files. However, when there are 'softer' signals and low pitched non percussive onsets this approach fails as described in [11]. The signal was initially downsampled to 11025Hz; thresholds were then dynamically allocated based on the peak values. We use a derivative of the energy function since it points out to the change in energy [11].

Firstly, the potential areas (onsets) in which we can find the ornaments were detected, then a variant of the conventional dynamic time warping algorithm is used to find the matching subsequences. Dynamic time warping (DTW) [30] is very similar to the dynamic programming algorithm edit distance. It is explained in detail in [30]. In this technique, we try to find the optimal alignment between two time domain sequences under certain constraints. One of the sequences is the pitch vector of the ornament template (X) and the other is that of the reference audio signal (Y) in which we try to find the ornaments. Given two time domain sequences X = (x1, x2... xN) of length N and Y = (y1, y2...yM) of length M, we have a feature space F constituted by (xn, ym) where n belongs to [1: N] and m belongs to [1: M]. To compare these feature vectors we need to find a local cost measure [30]. We use the Euclidean distance between the two features as our local cost measure. Lower the cost, more similar are the feature vectors. Evaluating the local cost measure for each pair of X and Y we obtain the similarity matrix D (X, Y). Now using this, we find the optimal alignment between the sequences X and Y such that it has minimal overall cost. This alignment is calculated using a recursive formula given by:

$$D(i,j) = d(i,j) + minD(i,j-1), D(i-1,j), D(i-1,j-1) \tag{3.1}$$

Here 1 i M, 1 j N and d (i, j) represents the distance between xi and yj. The resulting alignment has the value D (M, N). However, there are certain restrictions under which DTW is performed:

- Boundary condition (First and last elements of X and Y are aligned to each other)
- Monotonicity condition (Order of elements has to be maintained)
- Step size condition (No elements in X and Y are omitted)

In our variant of dynamic time warping we relax the boundary condition, as we are not aware of the boundaries of the potential regions which matches with the given template. We use DTW over other algorithms in time series matching like the Longest Common Subsequence (LCSS) because; LCSS is

Name	Occurrences	Detected	Ratio
Basic Meeṇds	135	107	0.79
Meeṇds resting on other intermediate notes	18	14	0.77
Āndolana	7	5	0.71

Table 3.1 Ratios of Successfully detected ornaments

not prone to rhythmic contractions and expansions. So when an artist increases or decreases the rhythm for a richer rendition, LCSS wont work even though the sequences are very similar. For example, if the artist playing the Marwa rāga introduces a glide from Re to Ga, he can either go directly from Re to Ga or play an extended version of Re and then slowly glide into Ga. These both are essentially the same meeṇd but this is not recognized by the LCSS algorithm. Such contractions/expansions and in accuracies in rhythm are accounted for in DTW. In Dhrupad, when one goes into Jor and Jhala, rhythm becomes very important. DTW effectively takes into account of such changes in rhythm to detect ornamentations.

3.4 Templates

- Basic meend-s: Re to Ga, Ga to Ma, Dha to Ni, Ma to Dha, Dha to Re, Re to Sa, Ma to Ma.
- Meends resting on other intermediate notes: Ni to Sa to Re, Ma to Dha to Ni, Re to Sa to Ni, Dha to Ni to Re, Ga to Ma to Dha.
- Andolana: Re andolith, Ni andolith.

3.5 Results

In this section we explain the accuracy of our technique. We have taken 3 performances in Dhrupad on rudra veena performed by Mr. Baha'ud'din Mohiuddin Dagar. Each performance is about 20-30 minutes long. We have manually annotated ornaments extensively in each of these performances. Each performance maintains the structure of a Dhrupad performance: starts with the ālap and then goes into the jor and the jhala. We perform iterative DTW, using each of the templates on the entire signal. A label is assigned based on the least DTW cost. The entire signal is segmented based on changes in energy and then we find out the longest common subsequence (subsequence with the least cost in the similarity matrix) in each of the segmented portions. The accuracy with which the ornaments were detected in the Marwa rāga is explained in table 3.1:

The accuracy for a single template is very high in case of ālap but the accuracy goes down in the jhala with the introduction of rapid rhythmic patterns. The occurrences have been manually annotated and hence are subject to error. True Predictions have been manually sorted out and differentiated from

Basic Meeṇds		**Predicted** Negative	Positive
Actual	Negative	11	14
	Positive	28	107

Table 3.2 Confusion Matrix for Basic Meeṇds

the false positives and they are based on dynamically allocated thresholds; otherwise an area with no ornament can also be classified as detection. These thresholds are different for different rāgas since different rāgas have different notes and consequently different templates. Even after setting thresholds, a template can detect some other ornament other than itself because of the similar notes present in them. For example, in Marwa rāga the template for Re to Ga meeṇd will detect a Re to Sa to Ni meeṇd even after setting a threshold because of the presence of the same note Re. We call these predictions as the false positives. We give a confusion matrix [Table 3.2] to depict these false positives for detection of meeṇd-s in Marwa. These rates vary for different classes of ornaments and consequently for different rāgas. False positive rate for this matrix is 0.56. True negatives and false positives have been tremendously affected by the temporal segmentation and threshold setting. In all these computations we have taken the manual annotations as the ground truth.

3.6 Strengths and Challenges of using Iterative Template Matching

We were successfully able to deploy the time warping technique to detect ornaments to a large extent. Since we primarily indulge in iterative template matching, classification of different ornaments has not been done. This leads to high false positive rates since thresholds needs to be allocated dynamically. This is heavily dependent on the dataset and templates available and thus very restrictive and non-exhaustive in its applicability to other datasets. A classifier which can distinguish between different alankāran needs to be implemented (which also recognizes various sub-classes). This might overcome the challenges faced by dynamic thresholding.

The templates we have chosen are based on some expert advice and our intuition; a learning model to extract these templates can make the entire process of detection much more efficient. There are also very few templates available, therefore the number of templates available and the variety of alankāran they cover can be increased.

Dynamic time warping is still very rigid in terms of the constraints [30]. No elements can be omitted and no replications are allowed in the alignment. Since in Dhrupad, many vocal expressions which are added differ from vocal style, artist and rāga, a procedure which is much more accomodative/flexible will increase the rates of detection.

We thus aim to build a generic model to classify ornamentation which takes into account the changes in both the frequency and amplitude domain. We aim to move away from the dependency on templates

for computationally understanding alankāran as these might or might not be representative of the many different kinds of alankāran.

Chapter 4

Engineering Features based on Domain Knowledge

As discussed in the previous chapter, in this chapter we describe our method to build a generic model to classify ornamentation which takes into account the changes in both the frequency and amplitude domain. This chapter explores the relevance/importance of transitions in improvisational decorations to understand the nature of flows in Hindustani music. Hindustani Music is fundamentally improvisational in nature. The artist's main objective is to evoke the mood of the rāga (melodic-mode) and this is achieved with the delicate explorations of flourishes and decorations. Music is cognitively associated with these decorations instead of the actual underlying notes. These decorations along with the varied methods of articulating notes together constitute flows. We call them flows due to the movement/change in the frequency/amplitude domains and they are mainly characterized by the rate/nature of change. We show that the sequences of change are fundamental to the idea of flows. We represent them by the first derivative and second derivative of a combination of frequency and amplitude data and then cluster them based on a custom defined distance. We successfully run spectral clustering on a pairwise affinity matrix and achieve an accuracy of 81% in distinguishing between Murki vs Kan and 84.5% in Āndolan vs Ghaseet. We thus develop a novel method of content based music analysis which has applications in music search, recommendation, retrieval and pedagogy.

4.1 Introduction

In Hindustani music improvisation is of paramount importance [29, 2]. The artist has a lot of freedom with which he renders the performance by using his own style of articulating notes and melodic patterns. In fact an artist is also judged by his ability to improvise well. These improvisations are mainly done using various alankār and alankāran [29]. These can vary from singing a group of notes that express the meaning (bhava) of the rāga to the manner of articulation of notes. For instance, execution of notes with a certain amount of shaking or a smooth transition from one note to another while touching other microtones.

Every performance in Hindustani Music is based on a melodic mode (rāga) [2]. Each rāga can evoke a harmonious aesthetic state given that there is a proper rendering of notes and the improvisation is

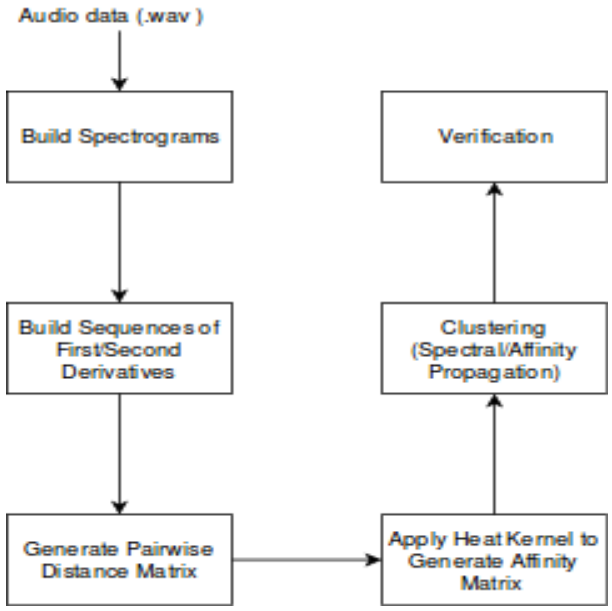

Figure 4.1 Flowchart of the domain-engineered approach.

aesthetically sound. Cognitively rāgas are associated with these decorations. These along with the ways in which notes are approached, sustained and quitted constitute flows. We call them flows due to the constant change in the frequency and amplitude domains. These flows are a prominent part of a performance and vary from performer to performer. The true nature of ornaments is that they are specific to the performer and vary a lot from performer to performer and style to style. The nature of rate of change cannot be captured just by looking at the pitch tracking information, not just because it ignores the energy changes but also because there might be subtler changes in the partials/overtones of fundamental frequency which might hold a lot of importance in understanding the rate of change in ornament based and other flows. In this chapter we want to capture the importance of rate of change in frequency and amplitude domains to conceptualize flows. We do this by clustering known groups of alankāran (one type of flow) based on a custom distance. This distance compares the sequences of first derivatives and second derivatives of a combination of frequency and amplitude data. The affinity/similarity between two flows is more when there is less distance between the sequences of change. We verify this hypothesis by first creating a pairwise distance matrix, then we use a heat kernel to convert this into an affinity matrix on which we run a spectral clustering algorithm. Since we know the types of flows in the given dataset we then verify if similar flows are being grouped into the correct known labels. We also try to find representative exemplars within these flows by later running the affinity propagation algorithm. The various steps involved in the procedure are shown in figure 1.

4.2 Background

As described in chapter 1 alankāran (an act of Ornamentation) is constituted by alankār, gamak, sthaya and kaaku. A specific group of notes that get its meaning in context of a musical phrase are known as alankār. Gamak is the execution of notes with a certain amount of shaking. Sthaya is a combination of a certain notes of the rāga which expresses a certain emotion of the rāga. Lastly, Kaaku refers to the manner of articulating a particular note. These are the different modes in which alankāran can be achieved. We are interested in those modes of alankāran which involve fair bit of improvisation and are specific to the performer/music style [29]. alankār further consists of Varnālankār, Chhandalankār, Varnatiriktaalankār and Vadanbheda. Varnālankār are alankārs using groups of notes. When note-groups are used in rhythmic patterns Chhandalankār arises. When alankāran is achieved by creating variations in the articulation of notes, varnatiriktalankār is achieved. In this paper we mainly focus on these kinds of alankārs. We particularly focus on Murki, Kan, Āndolan, meeṇds and Ghaseet. We give explanations of these terms even though some of them have been explained earlier in chapter 1.

1. Murki: A murki is cluster of notes that sounds like a short, subtle taan. It is a fast and delicate ornamentation employing two or more notes [29].

2. Kan: Kan are the linking or grace notes. They are played in a very subtle manner [29].

3. Āndolan: The Āndolan alankār is a gentle swing or oscillation that starts from a fixed note and touches the periphery of an adjacent note [29].

4. meeṇd: meeṇd refers to the glissando or a glide in the wave from one note to another [29].

5. Ghaseet: This is a kind of meeṇd specific to stringed instruments. While its literal meaning translates to pull, it usually refers to a fast glide [29].

North Indian classical music is not a staccato form and the transitions between notes are of utmost importance [36, 31]. The variations brought about in the rendering of notes such as: 1.) gentle oscillations, 2.) subtle increase/decrease in volume/speed, 3.) sudden variation in volume/speed, 4.) sustenance on one particular note for long, 5.) touching micro-tonal shrutis whilst performing alankāran etc. are instrumental to the performance and are representative of the style of music [29]. Cognitively we associate a particular performance with these improvisations and methods of articulation. More specifically, we associate music with such melodic patterns. These melodic patterns involving different methods of articulation of notes, transitions between notes and improvisational motifs constitute flows. Flows are characterised by rate of change. For instance, in a ghaseet there is a rapid change in frequency in a short interval of time since multiple micro-tonal shrutis are touched sequentially. Whereas, in a kan the change is more subtle. In mathematical terms flows can be understood by looking at the derivative of the signal in frequency/amplitude domains. In this paper we try to capture the change in frequency and amplitude domain together since there is a constant change in both the frequency and the amplitude

domain when a musician is trying to achieve alankāran. For instance, when Āndolan is being performed, the musician gently touches the periphery of other notes. Thus along with a change in the frequency domain there is also a notable change in the amplitude. This is the domain based understanding that we use to devise the entire experiment along with the features. Hence, we look at the first/second derivative of amplitude weighted frequency data.

4.3 Feature Extraction

In this section we will discuss about the various steps involved in representing flows in mathematical terms. As discussed earlier a flow can be mathematically expressed as the first/second derivative of the frequency/amplitude data. In this chapter we mainly are interested in certain ornaments for the study of flows. We have recorded various ornaments played on sitar and stored them as separate ornament flows. Now that the flows have been recorded we initially compute their spectrograms [12]. Spectrograms can be used as a way of visualizing the change of a non-stationary signals frequency content over time. Since we are interested in looking at the changes at a sub-second level we look at the frequency content at every hundred millisecond window. The audio files are sampled at 44100 Hz, so we consider the length of each segment to be 4410 samples.

After the spectrograms have been created, we are interested in finding the first and second derivatives of frequency and amplitude content. We approximate the calculation of derivatives by computing delta and double deltas. As discussed earlier, during the act of ornamentation there are changes both in frequency and amplitude content at once. To capture this dependence of frequency and amplitude while computing the derivatives, we dont look at frequency and amplitude separately but take a product of frequency and amplitude. We have thus chosen this measure over a linear combination of frequency and amplitude content.

Then we compute the derivatives of this combined measure. To make the idea clear lets take an example of a flow X, whose spectrogram has been computed. X has frequencies in the range (Fmin, Fmax) and in each 100 ms time window these frequencies can have varying energy content ranging from (0, Emax). Now lets say in the first window F has the highest energy content E. We compute sqrt(+E)*F and call this combined measure C which stores the product for the highest energy frequency in the first window. Now we go to the next time window and compute C by looking at the maximum energy frequency and amplitude. Then D1 is the difference between C in time window 2 and 1. Similarly computing for all time windows we get sequences of deltas for the highest energy frequency (D1, D2). If there are n time instances then we have the delta sequence of length n-1. These sequences are then calculated for the next highest power frequencies. At the end of this, each flow will have n sequences of deltas wherein n is the total number of frequency components in the spectrogram. Similarly we compute the second derivatives from the first derivative data. For an n length first delta sequence we get a n-1 length second derivative sequence. Here again we have n second derivative sequences for n number of frequency components in the spectrogram of the flow being considered. In conclusion, a flow

which has n frequency components and t time windows can be represented by two sequences: K first derivative sequences and K second derivative sequences, of length t-1 and t-2 respectively. Where K is set dynamically based on the amplitude level.

4.4 Distance Measure

Now that we have sequences of change representing the flows given, we come up with a distance metric to identify the similarity between two flows. Fundamentally to compare two time sequences the most commonly used idea is that of Dynamic Time Warping. Dynamic Time Warping is a dynamic programming algorithm which tries to find an optimal alignment between two time domain sequences [30]. Since we are interested in capturing the similarity between sequences of change this is suitable for our problem. Consider two time domain sequences X = [x1, x2...xM] and Y = [y1, y2...yN]. Evaluating the local cost between each pair of X and Y we obtain a distance matrix D(X, Y). Using this matrix we try to find the optimal alignment using the recursive formula:

$$D(i,j) = d(i,j) + min(D(i,j-1), D(i-1,j), D(i-1,j-1)) \tag{4.1}$$

Here d(i,j) is the local cost measure between Xi and Yj. D(M,N) now stores the value of the optimal alignment. Now that we have established a metric to find the distance between two time varying sequences we integrate it into our setting of multiple sequences of deltas/double-deltas representation.

Lets consider that we have two flows X and Y. X has n1 sequences of first derivative data and M time steps. Y has n2 sequences of first derivative data and N time steps. So X is a matrix with dimensions [n1, M] and Y is a matrix with dimensions [n2, N]. The difference is number of sequences is due to the varying frequency content in both the sequences. Thus we only look at top k sequences which correspond to sequences of top k highest energy frequencies in the spectrogram. So for top K highest power frequencies the first derivative distance (FDdist) between two flows can be given by:

$$FDdist = EuclideanNorm(D1(M,N), ..Di(M,N), ..Dk(M,N)) \tag{4.2}$$

Wherein, Di is the optimal alignment distance defined in Equation (4.1) between ith highest power frequency first derivative sequence of X with ith highest power frequency first derivative sequence of Y using the earlier defined Dynamic Time Warping algorithm. Similarly the second derivative distance (SDdist) between two flows can be given by:

$$SDdist = EuclideanNorm(D1(M,N), ..Di(M,N), ..Dk(M,N)) \tag{4.3}$$

To compute the ith-distance we compare the ith second derivative sequence of X with ith second derivative sequence of Y. The total distance between two flows is given as:

$$Dist(X,Y) = FDdist + SDdist \tag{4.4}$$

Where Dist(X,Y) is the distance between two flows X and Y.

4.5 Clustering

Since the distance metric between two flows has been established we can now perform clustering to check how well we can group the flows into their known labels. If we have n number of flows in our dataset we construct a nxn matrix of pairwise distances. We had initially run the K-Medoids clustering on this distance matrix to find that the clustering on distance wasnt very good [26]. Therefore now we construct an affinity matrix out of the distance matrix using a heat kernel, on which we can run the spectral clustering algorithm. In the context of clustering, an affinity measure is just the converse of a distance i.e. a distance of 0 means highest affinity. If values in the affinity matrix are not well distributed the spectral problem will be singular and not solvable. Thus we apply the below heat kernel on the given distance matrix:

$$similarity = e^{\frac{-distance^2}{(2.*(max-distance-min-distance)^2)}} \tag{4.5}$$

where max-distance is the maximum of all the distances and min-distance is the minimum of all distances.

4.5.1 Spectral Clustering

The goal of spectral clustering is to cluster data that is connected but not necessarily compact or clustered within convex boundaries [34]. It is efficient if the affinity matrix is sparse. It needs us to specify the number of clusters upfront and works well for small number of clusters. For two clusters, it solves a convex relaxation of the normalised cuts problem on the similarity graph. In scikit-learn spectral clustering does a low-dimension embedding of the affinity matrix between samples, followed by a K-Means in the low dimensional space. Steps involved in this type of clustering:

1. First we construct an affinity matrix A.
2. Then we construct the graph Laplacian from A. There are many ways to define a Laplacian. Normalized, generalised, relaxed etc.
3. Compute eigenvalues and eigenvectors of the matrix. Each eigenvector provides information about the connectivity of the graph. The idea of spectral clustering is to cluster the points using these ”k” eigenvectors as features.

4. Map each point to a lower dimensional representation based on the computed eigenvectors.

5. Assign points to clusters based on the new representation.

In this we essentially try to find a transformation of our original space so that we can better represent manifold distances for some manifold that the data is assumed to lie on [34]. When the data is projected into a lower dimensional space it makes the data easily separable and thus the clustering algorithm works. However, this still retains many properties of K-means since after we find a low dimensional embedding, we run the K-means algorithm. But the fact that we should know the labels before hand is not a problem for us since we have a labeled data-set. However within these labeled flows as well, there might be other intra clusters which we arent aware of because of the subtle nature of variations present in the flows. Thus we also try to find these smaller groups and representative exemplars by running the affinity propagation algorithm [8]. Spectral clustering works wells for us also because the size of the data-set is small enough. If there were items in our data-set in the order of 10^5 then we would have to construct an affinity matrix of size 10^{10} which would bloat up the main memory.

4.5.2 Affinity Propagation

Affinity propagation is a clustering algorithm which doesn't need any predefined number of clusters to be given as input. It finds the exemplars which are representative of the clusters in the data-set [8]. It views each data point as a node in a network, and recursively transmits real-valued messages along edges of the network. This is done until a good set of exemplars and corresponding clusters emerges. These messages are updated on the basis of formulas that search for minima of a chosen energy function. The magnitude of each message reflects the current affinity that one data point has for choosing another data point as its exemplar.

This method takes as input a real number s(k, k) for each data point k so that data points with larger values of s(k, k) are more likely to be chosen as exemplars. The number of identified clusters is influenced by the values of the input preferences. It is also affected by the message-passing procedure. The algorithm proceeds by alternating two message passing steps, to update the responsibility and the availability matrices. The responsibility matrix has values r(i, k) that reflect how well-suited xk is to serve as the exemplar for xi, relative to other candidate exemplars for xi. The availability matrix has values a(i, k) that reflects accumulated evidence as to how well-suited it would be for xi to pick xk as exemplar, taking into account the support from other points preference for xk as an exemplar. These matrices can be seen as log probability ratios.

Initially the availability is zero and then responsibilities are updated by the rule:

$$r(i,k) \leftarrow s(i,k) - max\{a(i,k') + s(i,k')\}, where \quad k' \quad s.t. \quad k' \neq k \tag{4.6}$$

Then the availability is updated as follows:

$$a(i,k) \leftarrow min\{0, r(k,k) + \sum_{i' \ni \{i,k\}} max\{0, r(i',k)\}\} for \quad i \neq k \quad and \tag{4.7}$$

$$a(k,k) = \sum_{i' \neq k} max(0, r(i',k)) \tag{4.8}$$

The iterations are performed until convergence, at which point the final exemplars are chosen, and hence the final clustering is given [8]. The data points whose responsibility+availability is positive are chosen as exemplars.

4.6 Results

Our data-set consists of few hundreds of samples of ornament flows comprising of Murki, Kan, Āndolan, Ghaseet (type of meeṇd). [FD] These ornaments were performed in different rāgas to create a comprehensive data-set. The ornaments were performed in rāgas: Malhar, Marwa, Mishra Piloo, Bhagyashree and Yaman. We test the validity of the distance metric and our representation by first performing spectral clustering and then running the affinity propagation algorithm. We first look at accuracy as the performance evaluation metric. Accuracy is given by:

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{4.9}$$

where tp = true positive, tn = true negative, fp = false positive, fn = false negative We find that the accuracy of clustering Murki and Kan by giving a predefined k equal to 2 is 0.81, i.e, the clustering algorithm correctly clustered murki and kan into their respective groups in 81% of the cases. The accuracy of clustering Āndolan and Ghaseet in a similar fashion is 0.84. These accuracies are very high considering that in our data-set of ornaments we had a variety not just in terms of rāgas but also in the articulation of these ornaments. For instance, our kan data-set has both two note and three note variants. The length of the ornaments in our data-set are also varied ranging from one second to many seconds. This shows the suitability of our distance metric and representation for varying length ornaments. We compare Murki/Kan and Āndolan/Ghaseet because they are fairly different and they usually dont occur together. In general musicians tend to combine Āndolan and murki in performance. Even ghaseet co-occurs with murki and Āndolan sometimes.

The Fowlkes-Mallows score FMI is defined as the geometric mean of the pairwise precision and recall [13]. It ranges from 0 to 1 and high score indicates good similarity between clusters. Perfect labeling has a score of 1.0. FMI is given by:

$$FMI = \frac{TP}{\sqrt{(TP + FP)(TP + FN)}} \tag{4.10}$$

where TP = True Positives, FP = False Positives, FN = False Negatives

FMI scores of 0.70 and 0.77 indicate that the precision and recall of our clustering is also very good. The values corresponding to performance evaluation metrics shown in Table 1 prove that the representation and distance metric have performed well in grouping similar flows together.

However we also test the legitimacy of our clustering by other metrics as well. Adjusted rand index (ARI) [21] is the number of pairs of objects that are either in the same group or in different groups in both partitions divided by the total number of pairs of objects. It ranges from -1 to 1 and a positive score indicates similar clustering.

$$ARI = \frac{RI - E[RI]}{max(RI) - E[RI]} \tag{4.11}$$

where RI is the rand index. The raw rand index is given by:

$$RI = \frac{a + b}{C} \tag{4.12}$$

where a = the number of pairs of elements that are in the same set in ground truth and in actual labels, b = the number of pairs of elements that are not in the same set in ground truth and in actual labels, C = is the total number of possible pairs in the dataset.

The ARI is positive and reasonably high for both clustering experiments. Higher ARI shows that the partitions are in agreement with each other. ARI is larger for the case of Āndolan vs Ghaseet.

The Mutual Information [47] is a function that measures the agreement of the two assignments, ignoring permutations. Bad/Independent labellings have negative scores. The calculated value of the mutual information is not adjusted for chance and will tend to increase as the number of different labels (clusters) increases, regardless of the actual amount of mutual information between the label assignments. Given that we just have two labels this is not a very good metric to assess our results. However a positive score is a good indicator.

Given that we have the knowledge of the ground truth class assignments, it is possible to define some intuitive metric using conditional entropy analysis. Homogeneity tells whether each cluster contains only members of a single class, while Completeness tells whether all members of a given class are assigned to the same cluster [39]. We observe that the homogeneity and completeness scores are low, which implies that there is a little bit of overlap in clustering. But since these functions increase at the pace of the logarithmic function, the numbers might improve significantly with more samples or clusters. V-measure or the harmonic mean is actually equivalent to the mutual information normalized by the sum of the label entropies [39]. Previously an analysis has been done on the impact of the number of clusters and number of samples on these metrics dependent on conditional entropy [44]. The observation is that the values of these metrics increase significantly with increase in number of samples and the number of clusters. Since our dataset is quite small these metrics can be ignored. For small sample sizes or number of clusters it is safer to use an adjusted index such as the Adjusted Rand Index (ARI).

Murki vs Kan	**Accuracy**	0.81
	Fowlkes Mallows Score	0.70
	Adjusted Rand Index	0.66
	Adjusted Mutual Information Score	0.46
	Homogeneity	0.47
	Completeness	0.48
	V-measure	0.47
Āndolan vs Ghaseet	**Accuracy**	0.84
	Fowlkes Mallows Score	0.77
	Adjusted Rand Index	0.75
	Adjusted Mutual Information Score	0.51
	Homogeneity	0.52
	Completeness	0.54
	V-measure	0.53

Table 4.1 Table containing values corresponding to various clustering metrics to validate the quality of spectral clustering.

Murki vs Kan	**Number of Clusters**	15
	Silhouette Coefficient	0.65
Āndolan vs Ghaseet	**Number of Clusters**	13
	Silhouette Coefficient	0.75

Table 4.2 Table containing the values corresponding to metrics of affinity propagation.

To verify the results of the affinity propagation algorithm we look at the silhouette coefficient [41]. Higher the value implies better the clustering assignments. The Silhouette Coefficient [41] is defined for each sample and is composed of two scores a: The mean distance between a sample and all other points in the same class. and b: The mean distance between a sample and all other points in the next nearest cluster. The Silhouette Coefficient s for a single sample is then given as:

$$s = \frac{(b-a)}{max(a,b)} \tag{4.13}$$

The exemplars (centers) extracted by this algorithm have been verified and have proven to show distinct musical significance.

4.7 Strengths and Challenges

We have shown that the property of rate of change is fundamental to the idea of flows and can be represented by the first derivative and second derivative data. We were able to achieve very high clustering accuracies by using this representation and our custom defined distance measure. We achieved an accuracy of 81% in distinguishing between Murki vs Kan and 84.5% in Āndolan vs Ghaseet. This again shows that transition is at the heart of the definition of flows and ornaments. However we have not compared some other flows such as the approaching of notes/quitting of notes separately across performances. Further study could take up only looking at patterns between these types of flows across individual performers and gharanas. Since this dataset is still very small, a comprehensive dataset can be built to check for the exhaustiveness of this hypothesis.

In this chapter we have just looked at one combination of frequency and amplitude data. Many other combinations can be explored to find the correct dependence. This can further lead to different representations of flows.

This theory of rate of change might be suitable for any other form of monophonic music as well where melody plays a major role. The transitional nature of melodic phrases can be expressed using a similar representation based out first and second derivative data. This type of content based audio analysis plays a prominent role in interacting with large volumes of audio recordings and also for developing novel tools to facilitate music pedagogy. It can have applications in music information retrieval, search and recommendation systems as well.

However there are a certain challenges associated with this approach. As discussed earlier in 1.4.1 there is a large variety in alankāran mainly because improvisation heavily depends on cognitive and imaginative resources of the performer and these can vary significantly from individual to individual. This dataset used is still not exhaustive to capture this variety. The approach proposed in this chapter relies on manual annotations for study. Ornaments and flows are intricate Indian music decorations which only experienced musicians can recognize and comprehend. It is thus very hard to generate a big annotated dataset. We are interested in having a big dataset driven exploration of these melodic patterns and also build a system which is scalable to hundreds of hours of music data. Scalability is a very important factor to consider while building such a transcription system.

Hand engineered approaches while being important, a unison of hand engineered approaches with actual data science insights can help build smarter approaches.

Chapter 5

Neural Network Based Approach: Learning fixed length representations

Ornaments and flows are intricate Indian music decorations which only experienced musicians can recognize and comprehend. We are interested in discovering the hidden musical patterns and structures in ornaments and flows which are fundamental in unfolding of a rāga. Recognizing these patterns can further help in music search, retrieval, recommendation and pedagogy. The biggest challenge in learning and making sense of flows or ornaments is the unavailability of annotated music data. Thus, to have a big data driven exploration of these melodic structures implies having an unsupervised or semi-supervised method of learning to reduce the annotation effort.

In the previous chapter while we were able to see that hand engineered representations were successful in distinguishing between ornaments we would ideally like to capture the underlying properties of ornaments without any hand engineered representations to build an even more generic framework.

5.1 Motivation for Neural Network Based Approach

Raw audio data is extremely large in size and any information retrieval/search or machine learning task on this data leads to extremely large computational overheads and often not very accurate results. Linear classifiers built on top of such high dimensional raw data usually dont tend to learn much about the structure of the underlying data. By representing the audio segments in a compressed fixed length vector format we can reduce the computational overheads by a significant margin. We can reduce the cost of indexing, cost of retrieval and other computational overheads in various machine learning experiments [20].

Word2Vec [48, 27] which transforms each word in a given text into a vector of fixed length has been proven to be very useful in various applications in natural language processing. These vectors were shown to carry a lot of semantic information about the words in consideration. In the domain of speech processing, works have focused on transforming audio segments into lengths of fixed dimensionality which has been very useful in various speech applications. For instance, these vectors representing audio segments have been given as input to standard classifiers for speaker/emotion classification tasks etc. But audio segmentation representation is still an open problem.

The vector representations developed in the previous chapter might not be able to efficiently capture the sequential information of audio since the representations were handcrafted instead of learning from raw audio data. Thus we adopt the approach of using sequence to sequence autoencoder.

In the domain of speech processing, neural networks have been recently successfully been used for encoding acoustic information into fixed length vectors [4]. Existing research has shown that it is possible to transform audio segments into fixed dimensional vectors. Word audio segments with similar phonetic structures were closely located in the vector space created by the transformed fixed dimensional vectors. Deep learning approaches however are very data hungry and they do not learn anything useful if there is a low resource dataset at hand. For an unsupervised learning approach to be devised we would need a big dataset with tens of hours of recordings.

Recently Chung et. al. [49], have developed an unsupervised approach to produce fixed length vector representations for variable audio segment data using sequence to sequence Autoencoders. These vectors were shown to describe the sequential phonetic structures of the audio segments to a good degree, with real world speech applications such as query-by-example Spoken Term Detection. They have even outperformed conventional approaches such as DTW with much lesser computational overheads. In the generic framework of a sequence to sequence autoencoder there is an RNN to encode the input audio sequence into a fixed dimensional vector and there is an RNN to decode the fixed dimensional vector to reconstruct the original audio input sequence [3, 7]. This approach has been used in text processing, video processing and more recently in signal processing for speech applications. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. This makes them suitable for capturing sequential information since the outputs depend on previous computations [7].

In this work, we try to extend the same model for learning fixed length features using sequence to sequence autoencoder to represent audio segments of ornaments and flows. We expect that these learned audio representations capture essential sequential structural information, as the rate of change/sequence of changes in amplitude and frequency is the underlying foundation of ornaments. We here expect that similar ornaments and flows would be nearby in the vector space created by the fixed length representations. This is completely an unsupervised task since we dont need any annotations to build the fixed length representations.

5.2 Segmentation

The first step in learning fixed length representations for audio segments is to extract the audio segments from an audio recording of a musical performance. The features of segments extracted here are further given to the sequence to sequence autoencoder. We use a semi-supervised event detection method to segment the audio signals into musical events of interest. This approach is inspired by a similar algorithm implemented in the pyAudioAnalysis library [16]. The method takes an uninterrupted audio recording as input and returns segment boundaries that correspond to various musical events of

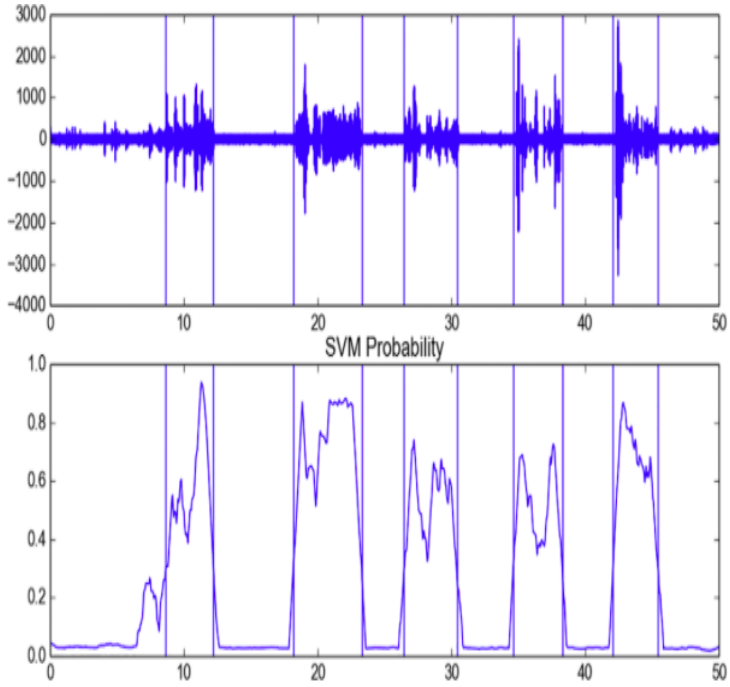

Figure 5.1 Signal after Segmentation is shown in the bottom [16]

interest. It also removes the silent areas of the recording. We achieve this through a semi-supervised approach which performs the following steps:

- The short-term features of the entire musical recording are extracted
- We now train an SVM model to distinguish between high-energy and low-energy short-term frames. First we pick out the top 15% of the highest energy frames (which correspond to ornamentation events) along with the top 15% of the lowest energy frames and give them as input to train the SVM model. The top 15% of events picked here have both long length and short length ornamentation events.
- The SVM classifier is applied (with a probabilistic output) on the complete audio recording, resulting in a sequence of probabilities that correspond to a level of confidence that the respective short-term frames belong to an audio event (and do not belong to a silent segment)
- We then use dynamic thresholding to detect the musical segments and ignore silence

This approach is performed individually for each audio recording in the dataset as the high and low energy frames vary a lot from performance to performance.

5.3 Event Features

Once the audio segments have been computed, we extract acoustic features of interest which are further given to the sequence to sequence autoencoder. Various acoustic features used in our experiments are:

- MFCCs: The MFCC are the coefficients that make up the Mel cepstrum [30]. The Mel-cepstrum is the cepstrum computed on the Mel-bands (scaled to human ear) instead of the Fourier spectrum [30]. In MFC the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response.

- Spectrogram: A spectrogram is a visual way of representing the signal strength over time at various frequencies present in a particular waveform. A common format is a graph with two dimensions: the x axis represents time, whereas the y axis represents frequency; the amplitude of a particular frequency at a particular time is represented by the color of each point in the image. This can be read as an n-dimensional matrix or tensor mathematically [12].

- First/Second Derivatives: This is same as the deltas and double deltas explained in chapter 4. After the spectrograms have been created, we find the first and second derivatives of frequency and amplitude content. We approximate the calculation of derivatives by computing delta and double deltas. As discussed earlier, during the act of ornamentation there are changes both in frequency and amplitude content at once. To capture this dependence of frequency and amplitude while computing the derivatives, we dont look at frequency and amplitude separately but take a product of frequency and amplitude.

However we have noticed that MFCC gave us the best results which are explained in section 5.6. It is important to note here that within each time group there can be slightly varying length ornament segments. We thus perform one-hot encoding (append 0s to the extra dimensions) to normalize the lengths of all segments.

5.4 Proposed Approach

The objective here is to transform a musical segment represented by acoustic features such as MFCCs or Spectrograms, x= (x1, x2,..xT) where xt is the audio feature at time t and T is the length of the vector representation into a compressed vector of fixed length dimensionality. It is expected that this compressed fixed length representation can capture the sequential information of ornaments (represented by the audio segment) which is essential in distinguishing between different types of ornaments.

We split our dataset of ornaments into different classes based on their duration and perform analysis on each of these splits separately. For instance, consider an audio file representing a Hindustani musical performance. Post the segmentation phase, segments of varying length duration are obtained. Segments

in the range 0-2 second are grouped into time group 1 (TG1), segments in the range 2-4 seconds are grouped into time group 2 (TG2), segments in the range 4-6 seconds into time group 3 (TG3). All segments beyond 6 seconds are grouped into a single class time group 5 (TG4). It is expected that most musically significant events/ornaments are at the microsecond level and segments which are relatively large are less significant. Once the time groups are segregated we perform our experiments on each of these time groups respectively. In the following sections, we start with explaining a RNNs, simple autoencoder framework, sequence to sequence autoencoder framework, and finally denoising autoencoders.

5.4.1 RNNs

Recurrent Neural Networks are a class of neural networks which are very good for capturing dynamic temporal information. Given a sequence x = (x1, x2,..,xT) an RNN updates it hidden state ht according to the current input and ht-1. The hidden state acts as internal memory that enables it to capture dynamic temporal information and deal with sequences of variable lengths [3, 7]. This makes them highly applicable for tasks such as speech recognition , handwriting recognition etc. This also makes it highly suitable for our task in hand. By learning valuable sequential and temporal structure of data it can help us understand and discriminate between flows. Gated recurrent units (GRUs) are a gating mechanism in recurrent neural networks. We use GRU units in our auto-encoder [6].

5.4.2 RNN Encoder-Decoder Framework

In a simple RNN encoder-decoder framework there is an encoder RNN and decoder RNN. RNN encoder first reads the given input vector x = (x1, x2,..,xT) sequentially and the hidden state ht is updated respectively. Each of these input objects in x = (x1, x2,..,xT) are further vectors at different time steps and thus we have a hidden state vector ht at each time step. After the last input object xT is processed hT is considered to be the learned representation of the whole input data. The decoder RNN now picks up this hidden vector and produces an output sequence y = (y1, y2,.., yT) where T might or might not be equal to T. Which means the length of x and y can vary. However in our use case we are not interested in having ys length to be different than xs. Hence, we arrive at autoencoders [3, 7].

5.4.3 Autoencoder Framework

An autoencoder in the context of deep learning is a unsupervised learning algorithm to learn efficient encodings. It has an input layer, an output layer and a hidden layer (in its simplest form) and the number of nodes in the input layer is same as the number of nodes in the output layer. By setting the output values same as the input values it tries to learn an identity function or an efficient encoding of data which can capture some interesting structural properties of the given input data. It uses backpropagation to reconstruct its own input, thus it is an unsupervised learning algorithm. They are data specific and

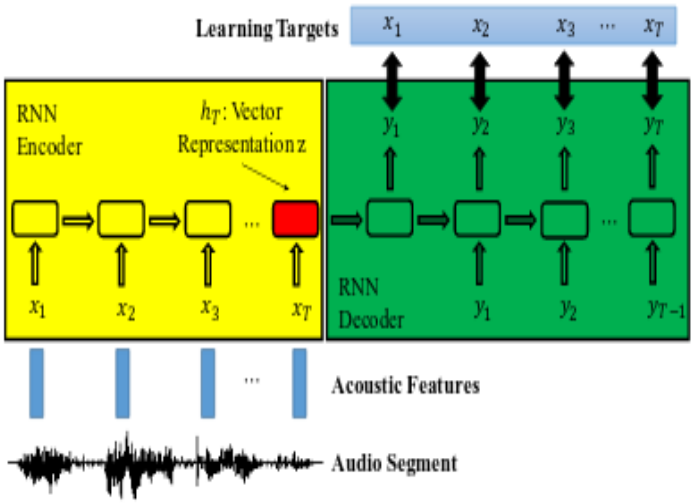

Figure 5.2 Sequence to Sequence Autoencoder [49]

work only on the datasets they have been trained on. In our experiments we use sequence to sequence autoencoders which are better explained below [3].

5.4.4 Sequence to Sequence Autoencoder

By integrating the RNN encoder-decoder framework and the autoencoder framework we obtain the sequence to sequence autoencoder framework [22]. Since our input is a sequence we are interested in capturing the temporal information. Given an input sequence x = (x1, x2,..,xT), there is an RNN encoder which reads each input feature at time t, xt sequentially and updates the hidden vector ht. After processing the entire input sequence it gives the hidden state ht as the learned representation to the RNN decoder. We call this representation z. The RNN decoder takes ht as its first input and tries to produce its first output y1, then it uses y1 to produce y2 and so on. The framework of both the encoder and decoder is jointly trained by minimizing the mean squared error.

Denoising Sequence to Sequence Autoencoder: While the representation learned above is good we are interested in learning much more robust representations. Thus here we corrupt the initial input data (x) with some noise and give the corrupted input (x) to reconstruct the original input vector x [35].

Stacking Denoising Sequence to Sequence Autoencoders: To give the model more expressive power, we can add multiple layers of autoencoders to process the data. The output of one layer becomes the input to the next autoencoder and so on. The last hidden layer in this stacked architecture is used as the learned representation for our experiments [35].

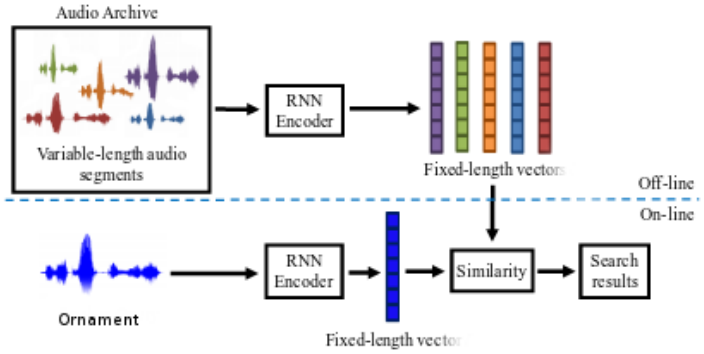

Figure 5.3 The example application of Ornament retrieval. All audio segments in the audio archive are segmented and represented by fixed-length vectors off-line. When an ornament is given as input, it is also represented as a vector, and the similarity between this vector and all vectors for segments in the archive are calculated, and the audio segments are ranked accordingly [49]

5.5 Example Application: Ornament Retrieval

The learned fixed length feature vectors in the previous step can be used in various scenarios. Here we present a small example in the domain of ornament/flow retrieval. The objective here is that given an audio segment of an ornament or a flow we want to be able to search for the closest resembling ornaments or flows in a large archive (not annotated) of North Indian musical performances. This has various applications in building a self learning musical platform for pedagogy. A user can give input as an ornament and retrieve diverse/ richer and similar renditions of the same ornament.

In figure 5.3 we explain how our learned representation can easily be used for building such a ranking and retrieval system. There are two phases in building such a retrieval system. In the initial offline method, we take audio segments from a huge audio corpus and run the segmentation algorithm explained above to get varying length audio segments. Now based on the length we split them up into groups of similar length. The acoustic features of these audio segments are then retrieved and given to a RNN autoencoder. The RNN autoencoder then learns fixed length representations of these segments. Since segments belonging to the same group can also have slightly varying lengths we use one hot encoding to normalize the lengths. This approach is inspired from [49] and it is similar to the Query By Example experiment in the speech processing domain [14].

Now that we have learned the fixed length representations we can perform the online analysis task. In the online analysis task, given an ornament audio segment query we want to be able to retrieve and rank the top most similar audio segments (representing ornaments/flows) in the corpus. When an audio segment is given as an input to the system, the RNN encoder returns a learned fixed length feature vector. This when given to the ranking and retrieval system, it returns a list of audio segments in the archive ranked according to the cosine similarities evaluated between the vector representation of the ornament query and those of all segments in the archive.

5.6 Experiments

In this section we report the dataset used, the experimental setup, different approaches to cluster analysis, online experiments analysis and results.

Experimental Setup: Our dataset comprises of 50 hours of North Indian musical performances rich in ornaments and flows. [SSD] We primarily look at instrumental music pieces instead of going into the complexities of vocal performances. This dataset of 50 hours is used for training the sequence to sequence autoencoder. We also have a small annotated dataset or ornaments which we use for testing purposes. The entire corpus was initially segmented using the algorithm explained in the section above and grouped according to the duration of the segment. As explained earlier, we perform experiments separately on each of these time groups. All the musical performances in the dataset are converted to mono audio and are sampled at 44100 Khz. For feature extraction we set the length of frame to be 4410 samples and frame overlap to be equal to the half of it. We experiment with different features: 1.) MFCC (13 dimensional feature vector) 2.) Spectrogram (256 length feature vector) 3.) First and Second derivatives of Spectrogram. Features are extracted for each time step. For instance, if we have a 2 second audio segment we obtain 40 time steps. Thus MFCC features for this time step will be 13X40 tensor. We flatten this into a one dimensional vector and give it as input to the Sequence to Sequence autoencoder. We use Keras and Tensorflow for implementing the Autoencoder. The architectural details of the Sequence to Sequence Autoencoder were as follows:

- We use GRU as the gating mechanism in our RNNs. Both the RNN encoder and decoder had 3 hidden layers with the number of GRU units equal to one fourth the original feature vector dimension. So the number of GRU units based on whether the input feature vector is MFCC or SPEC. Thus each input audio segment was mapped to a fixed length representation one fourth its size.

- The network was trained using Stochastic gradient descent (to minimize the mean squared error) with learning rate of 0.1 and without Nesterov momentum.

- We use the hyperbolic tangent function as the activation function. The tanh function is sigmoidal and returns a value between (-1, 1). Strongly negative inputs to the tanh will map to negative outputs. The network was trained using this activation function and (0-600) epochs. We tested with different epochs (in an increasing order upto 600) to verify if we have a consistently decreasing loss function. For inner activation we use the hard sigmoid function.

- For corrupting the input data we use the zero-masking approach similar to [35], in which we randomly set some elements in the input vectors to zero. This corrupted input is then given to the Denoising Sequence to Sequence Autoencoder to reproduce the original input.

The testing set served two purposes: It was used to test if the learned representations are accurately clustering similar flows/ornaments into their respective groups. It was also used in the online analysis task to retrieve and rank the top most similar audio segments.

5.7 Verification of Learned Representations

In this section we wish to verify that the representations learned by the sequence to sequence autoencoder capture the sequential properties of flows i.e, we expect similar flows/ornaments to have similar representations.

5.7.1 Cluster Analysis

Before we can verify the similarity of learned representations we wish to cluster the flows/ornaments based on the learned representations. We perform simple K-means clustering [28] to group these flows into their respective categories. Representations learned in the previous phase by the sequence to sequence autoencoder are given as the feature vector input to K-means clustering method. The biggest challenge in clustering is to figure out the ideal K value. We use silhouette analysis to arrive at this K value.

Silhouette as discussed in the previous chapter is a method of validation of consistency within clusters of data. The method provides a representation of how each data point lies within its cluster. The Silhouette Coefficient is defined for each sample and is composed of two scores a: The mean distance between a sample and all other points in the same class. and b: The mean distance between a sample and all other points in the next nearest cluster. The Silhouette Coefficient s for a single sample is then given as:

$$s = (ba)/max(a, b) \tag{5.1}$$

Silhouette value of +1 indicates that the sample is far away from the neighboring clusters and it is well matched within its cluster. A value of 0 indicates that the sample is on or very close to the decision boundary between two neighboring clusters. Negative values indicate that those samples might have been assigned to the wrong cluster. Silhouette analysis provides a measure of how close each point in one cluster is to points in the neighboring clusters and thus can be used to find out the appropriate K value. We perform iterative analysis to come up with an ideal K value. Figure 5.4 corresponds to our experiment in which we iteratively increase the K value and perform silhouette analysis at each iteration. This experiment is performed on time group 1 (TG1). Time group 1 has 9780 data points. The plot is for cluster labels and their corresponding silhouette values. We see that when the number of clusters is 3,5,7 there are wide fluctuations in the size of silhouette plots. We also see that there are clusters well below the average silhouette scores. However with k=11 we see that clusters are lot more balanced and there isnt much fluctuation in size of silhouette plots. The silhouette scores for different values of K in the figure 5.4 are as follows: for K=3 score=0.77, for K=5 score=0.78, for K=7 score=0.83, for K=11 score=0.96

We can also perform hierarchical clustering [38] to group these flows into different categories. Hierarchical clustering is a method of cluster analysis which seeks to build a hierarchy of clusters. The advantage Hierarchical clustering provides is that there is no apriori information about the number of

Figure 5.4 This figure has four silhouette plots with x-axis: silhouette coefficient value and y-axis: different colour corresponding to different cluster label. In each plot the value of k is varied. Top left K=3, top right K=5, bottom left K=7, bottom left K=11. We see that with increasing K value the clusters become lot more uniform with less fluctuations.

clusters required. However the time complexity is large and it can add to computational overheads i.e, takes lot of time for large datasets. It is also sensitive to noise and outliers. We choose K-means to have more control over the number of clusters and for higher scalability. Once K has been determined and the dataset has been split up into different clusters we then perform cosine analysis to see how well the learned representations can be used to group similar ornaments/flows into the same clusters.

5.7.2 Analysis of Learned Representations

In this section we wish to verify if the representations learned by the sequence to sequence autoencoder capture the sequential properties of ornaments/flows. We start with simple cosine analysis [45] as sanity tests i.e, we wish to verify if objects in the same clusters have high similarity scores and objects belonging to different time groups have largely varying similarity scores. Later we wish to compare the DTW scores between ornaments belonging to same/different time groups.

As expected the cosine similarity of audio segments belonging to the same time group and same cluster (same time group intra-cluster) are extremely high averaging 0.9 (out of 1) for almost all clusters of all time groups. This is indicative of the appropriate choice of K in our initial phase of clustering.

Time group	Average Similarity Score	Median Similarity Score	Range (clusterwise)	Variance
TG0	0.4	0.4	[-0.07,0.9]	0.05
TG1	0.6	0.61	[0.41,0.83]	0.02
TG2	NA	NA	NA	NA
TG3	NA	NA	NA	NA

Table 5.1 Cross cluster, Same time group Cosine Similarity Analysis

We also compute the cosine similarity scores between members of different clusters but the same time group. For example, we take each member of [cluster 1 time group 1] and compare them with each member of [cluster 2 time group 1] and so on. In some cases the average similarity was over 0.9 which is indicative that there are similar ornaments in more than one cluster group. Whereas in most other cases the average cosine similarity score was well below that score. In table 5.1, we show the average cross cluster similarity score in all the time groups. The mean similarity score in case of time group 0 (TG0) is 0.4 which is fairly low considering high intra-cluster same time group similarity values. We do not calculate these scores for time groups 2 and 3 since the number of samples is very low and the values wouldnt be representative of anything valuable. They have been marked as NA (not applicable). Here it is important to note that the values are not deterministic since with every run of k-means we can get different distribution of clusters.

Then, we compute the cosine similarity scores between members of different clusters and different time groups. While performing cosine similarity analysis between different time group clusters we perform one-hot encoding (append 0s to the extra dimensions) to normalize the lengths. As expected the average similarity scores are very low in this case since the size of the vectors involved itself vary in great degree. For instance, in the case of Sequence to Sequence autoencoder trained by MFCC feature vectors, the final output vector in the case of time group 3 was double the length of output vector produced by time group 2. This alone can be a major factor for significantly lower similarity scores. In table 5.2 we show the average cross time group similarity scores. We see that the maximum cosine similarity is seen in the case of (TG1, TG2). Upon further inspection we see that these are fairly small cluster groups for which there was this high similarity score. However this time group has the highest range of inter-cluster similarities.

After performing cosine analysis we wish to compare the DTW (Dynamic Time Warping) scores between the ornaments. DTW is a good indicator of similarity between temporal sequences as discussed earlier. Here again we wish to verify that ornaments belonging to the same cluster have high similarity scores in their respective time groups. In table 5.3 we show the average DTW score between pairs of the same time group. This average value is the normalized average of average DTW scores computed for pairs of same clusters of a given time group. The highest DTW score average among all time groups is taken as 1, while the least is taken as 0. Range is [0,1], where 0 implies low DTW score and higher similarities. For instance if the highest average DTW score among all the time groups is of TG3 and it

Time groups	Number of Pairs	Average Similarity Score	Median Similarity	Range
TG0, TG1	13936500	0.021	0.059	[-0.2, 0.49]
TG0, TG2	3569700	0.17	0.18	[0.08, 0.33]
TG0, TG3	2004900	-0.14	-0.19	[-0.3, -0.05]
TG1, TG2	520125	0.43	0.46	[0.27, 0.65]
TG1, TG3	292125	0.02	0.03	[0.01, 0.55]
TG2, TG3	74825	0.02	0.02	[0.02, 0.03]

Table 5.2 Cross Cluster Cross Time Group Cosine Similarity Analysis

Time group	Average DTW Score (normalized)	Median DTW Score (normalized)	Variance
TG0	0.01	0.02	0.001
TG1	0.07	0.04	0.004
TG2	0.23	0.17	0.01
TG3	0.43	0.35	0.09

Table 5.3 Cross cluster, Same time group DTW Analysis

amounts to 500, then the average DTW scores of other time groups will be normalized by 500. Here we see that the scores are very low in the case of smaller time groups and increase as the number of time steps increase. The very low variance value of 0.001 for time group 0 (TG0) indicates that there isn't much difference between average normalized DTW scores between different clusters of that time group. Whereas the relatively higher variance value of 0.09 in case of time group 3 (TG3) shows a significant variation in average normalized DTW scores between different clusters of that time group.

Finally, we compute the average normalized DTW scores between members of different clusters and different time groups. This average value is normalized by the highest cross time group average DTW scores. For instance, if the highest average DTW value is between TG0 and TG2, we normalize all other average DTW scores with this value. In table 5.4 we show the normalized average DTW scores between members of different clusters and different time groups. We see that the average DTW scores (normalized) are much higher than that in the previous table 5.3. The average score is highest in the case of the pair of TG0, TG3. This is as expected since the number of time steps itself is significantly larger in this case leading to huge difference in DTW scores. The variance is significantly lower in this case as well, which indicates consistently high DTW scores. The least score is in the case of the pair of TG0, TG1.

5.7.3 Manual Analysis of Clusters

In this section we randomly pick a few samples from the clusters and see if they are musically similar. The samples studied are available at [CSD].

Time groups	Number of Pairs	Average DTW Score (normalized)	Median	Variance
TG0, TG1	13936500	0.24	0.27	0.08
TG0, TG2	3569700	0.53	0.62	0.12
TG0, TG3	2004900	0.80	0.81	0.007
TG1, TG2	520125	0.31	0.35	0.08
TG1, TG3	292125	0.39	0.49	0.11
TG2, TG3	74825	0.29	0.31	0.09

Table 5.4 Cross cluster, Cross time group DTW Analysis

Time groups	Cluster Id	Major Patterns
TG0	0	1.) Ascending and Descending group of notes with Krintan and meeṇd 2.) In the segments spanning the later half of the performance sequences of notes covering the entire octave 3.) Fast meeṇds spanning multiple notes 4.) Multiple note sequences with multiple meeṇds
TG0	1	1.) Long meeṇds 2.) Some sequences with both grace notes (krintan) and meeṇds. This is fewer in comparison with the 0th cluster. 3.) Segments corresponding to the later half of performance have some extremely fast moving ghaseets
TG0	2	1.) Rapid sequences of notes with fast meeṇds. 2.) Some samples of multi-note occurrences with long meeṇds
TG0	3	1.) Very diverse set of patterns. Very Slow meeṇds, Rapid meeṇds 2.) Variety of Murkis. Also occurence of combination of murkis and meeṇds.
TG0	4	1.) Consistent set of basic long meeṇds. They are usually in combination with Krintan and gamak. 2.) Presence of few rapid murkis and gamaks which are musical segments near the end of performances

TG0	5	1.) Most of them are multi note sequences with Krintans. That is all these notes are gracefully rendered continuously. The length varies from a few notes to the use of many grace notes.
TG0	6	1.) Multi note plain sequences. 2.) Multi note sequences with small murkis. 3.) Plain note sequences with some Kaṇ swar.
TG0	7	1.) Rapid murkis 2.) Few Slow meeṇds 3.) Multi note meeṇds 4.) Fast paced rapid murkis
TG0	8	1.) 1 note meeṇds 2.) 2 note meeṇds 3.) Multi note meeṇds 4.) Fast rapid meeṇds and murkis 5.) Multi note meeṇds, also few plain note sequences
TG0	9	1.) 3-4 note sequences with murkis and meeṇds 2.) Some end of performances segments with rapid meeṇds
TG0	10	Large note sequences ranging from 3 notes to a full octave. Some have meeṇds and murkis.
TG1	0	This mainly has rapid fretting of the string. It can be said that there are hints of emphasis on notes (maybe Kaṇ) but not a significant presence of alankār related to the note.
TG1	1	1.) Groups of varṇas used with Kaṇ and long meeṇds in a significant portion of the samples 2.) There is a good combination of tāla samparkit, and varṇatiriktaalankār in this group
TG1	2	1.) Large note sequences where notes are rendered with Kaṇ or krintan 2.) A note is approached in this group with emphasis in general. There is also presence of ascending and descending set of varṇas
TG2	0	1.) varṇa based alankāran: alankāran created due to the presence of certain varṇas. Groups of notes used in tandem

		2.) Few note sequences followed by a murki or meeṇd near the end. Few note sequences with Kaṇs and krintans to render notes. 3.) Presence of a few ghaseets to present the complete octave 4.) This group has alankāran related to varṇa and taala mainly.
TG2	1	1.) Note sequences covering complete octave 2.) Ascending and descending sequences of 6-7 notes 3.) Minor presence of gamaks
TG2	2	1.) Taala Samparkit alankār 2.) Rapid movement of notes in large note sequences 3.) Some note sequences with murkis and other gamak
TG3	-	1.) Large rhythmic sequences with tabla. Prominence of Tabla (accompanied instrument) in segments 2.) Large note sequences with murkis

Table 5.5: Manual Cluster Analysis

In table 5.5 we can explain in detail the properties observed in each of the clusters of different time groups. In time group 0 cluster 0, we notice that most of the samples are multiple note sequences with hints of Krintan and meeṇd used together. We found that most ornaments occur together in a segment. We also notice that segments in the back-end of the performance covering the entire octave are also a part of this group. In general lot of these samples at the back-end of the performance, spanning multiple notes were found in a lot of groups which is interesting. In time group 0 cluster 1, ghaseets and meeṇds were grouped together. This is musically correct since the temporal structure (in terms of spectral properties) are very similar between ghaseets and meeṇds. Time group 0 cluster 2 however is very different from the above two. It has sequences of notes with fast meeṇds. There is a very diverse set of patterns available in time group 3 cluster 0 and it is hard to find consistent musical similarity in this group. Time group 0 cluster 4, 7, 8, 9 are similar to cluster 0. There is a variety of combination of murki, Krintan, gamak and meeṇd used in all these clusters. Time group 0 cluster 6 has been observed to have plain note sequences with the presence of a few grace notes (Kaṇ). The last cluster in this time group mainly has large note sequences (greater than 3 notes) with the introduction of murkis and meeṇds on some notes.

In time group 1 cluster 0, samples have rapid fretting of the string. It can be said that there are hints of emphasis on notes (Kaṇ) but not a significant presence of ornaments related to the note. In time group 1 cluster 1, musical segments were found to have either taala-samparkit alankār or varṇa-alankār. Time

groups 2 and 3 consist of samples larger than 4 seconds. In time group 2 as well we find consistency in musical similarity across all the clusters.

In time group 3, we don't look at all clusters separately since its a small set of data samples. We randomly pick and observe a few data samples across clusters. We notice that most of the musical segments in this time group are large rhythmic sequences. There is a prominence of an instrument such as tabla in these sequences. Also, they are usually from the back-end of the performances. There are murkis along with some notes in some sequences but this is not consistent. Thus, we notice that there is some musical consistency across a majority of the clusters which is indicative of the semantic information captured by these clusters and in turn by our method. However this musical consistency doesn't imply that they all represent a particular group of ornaments. It implies:

- The clusters represent permutations/combinations of ornaments. Lot of ornaments such as murki, Kaṇ and gamak occur together in a musical segment. In live performances, performers use these ornaments in combination, which makes it hard to distinguish between them computationally (when they occur together). So a certain kind of combinations are grouped into the same category.

- Some ornaments have similar melodic properties as others which is why they are grouped into the same category. We have discovered some interesting groupings, such as: certain sequences of murkis were grouped along with meeṇds which indicates high similarity between these groups. Ghaseets were also grouped with murkis. This is because of the similar temporal change properties (in frequency domains) in these groups.

Since most of these ornaments occur together it might be more interesting to study how and when they occur together and its musical significance. We have also noticed that musical segments based on taala and rhythm form separate clusters (groupings) in all time groups. Which means that taala-samparkit alankār can very well be distinguished from the rest of the ornaments.

We have started this study to have a big data exploration of ornaments in Indian music but we discovered that ornaments are not present in an isolated form in most Indian music performances. Thus its important to study ornaments as a combination of multiple melodic structures.

5.7.4 Testing Phase

However we do wish to verify if this approach can distinguish between ornaments which have been recorded in isolation for the purpose of this study. Until now the experiment had no usage of annotated data. In this section we wish to explore the relevance of the learned representations in clustering similar ornaments (which do not occur together) into the same groups. However it is important to note that the size of this annotated dataset is just around 500 samples. Annotated dataset has 5 major classes: 1.) Āndolan, 2.) Gamak, 3.) meeṇd, 4.) Kaṇ, 5.) Ghaseet . These terms have been explained in earlier chapters. As discussed earlier there is a lot of commonality between these groups so we don't give all the annotated samples to the same clustering experiment. First we compare Murki with Kaṇ and then

Murki vs Kaṇ		**Truth** Murki	Kaṇ
Predicted	Murki	66	3
	Kaṇ	18	61

Table 5.6 Confusion Matrix for Murki vs Kaṇ

Murki vs Kaṇ	
Accuracy	0.86
Precision	0.78
Recall	0.95
F1 Score	0.86

Table 5.7 Confusion Matrix Analysis for Murki vs Kaṇ

we compare Āndolan vs Ghaseet. These groups are significantly different from each other and thus it is relevant to compare these instead of meeṇd and Ghaseet which are very similar musically.

First we compare Murki with Kaṇ. We take the annotated samples and pass it through the sequence to sequence autoencoder framework previously trained to learn representations of these samples. We then make a mapping between the annotated sample and the new learned representation. Now these samples are given to a K-Means clustering procedure where K=2. We compare if the samples belonging to one class are grouped into the same category or not based on the newly learned representation. Table 5.6 provides the confusion matrix for Murki vs Kaṇ. This dataset was filtered i.e, if it consists of combination of ornaments or low amplitudes they were excluded from the experiment. We see that 66 of the 84 Murki are grouped correctly and 61 of the 64 Kaṇ are grouped correctly. Table 5.8 gives the confusion matrix for Āndolan vs Ghaseet. We see that 71 of the 86 Āndolan were grouped correctly and 35 of the 45 Ghaseet were grouped correctly. Thus we have achieved accuracies of 0.86 and 0.81 in distinguishing between Murki vs Kaṇ and Āndolan vs Ghaseet. This implies that the learned representations are useful in distinguishing between different classes of ornaments.

5.8 Further Work and Conclusions

We developed a robust unsupervised method to learn fixed length representations of ornaments. Existing research has shown that it is possible to transform audio segments into fixed dimensional vectors. In this thesis we extended the same model for learning fixed length features using sequence to sequence autoencoder to represent audio segments of ornaments and flows in musical data without human annotation (Curated dataset of 50 hours consisting both sitar and sarod performances). We have started this study to have a big data exploration of ornaments in Indian music but we discovered that orna-

Āndolan vs Ghaseet		**Truth** Āndolan	Ghaseet
Predicted	Āndolan	71	10
	Ghaseet	15	35

Table 5.8 Confusion Matrix for Āndolan vs Ghaseet

Āndolan vs Ghaseet	
Accuracy	0.81
Precision	0.82
Recall	0.88
F1 Score	0.85

Table 5.9 Confusion Matrix Analysis for Āndolan vs Ghaseet

ments are not present in an isolated form in most Indian music performances. Thus its important to study ornaments as a combination of multiple melodic structures. We have also discovered that some ornaments have similar musical properties as some others which is why they are grouped into the same category. For instance, certain sequences of murkis were grouped along with meeṇds which indicates high similarity between these groups. This might be because of the partially similar temporal change properties (in frequency domains) in both these groups. Further, we have also noticed that musical segments based on taala and rhythm form separate clusters (groupings) in all time groups. Which means that taala-samparkit alankār can very well be distinguished from the rest of the ornaments.

We have thus seen that these learned audio representations capture essential sequential structural information and are useful in distinguishing between different types of ornaments (recorded in isolation) in the vector space. We have verified this using cosine analysis, DTW analysis and k-NN Classifier on a testing set.

However the real application of this approach can be utilized in a real time ornament tagging system. Since the ornaments won't be present in isolation we could detect segments and classify them as belonging to a certain combination of ornaments instead of one specific ornament class. In the rest of the explanation wherever we mention building an ornament tagging system we mean that the tagging is a combination of ornaments.

To build a scalable real time ornament tagging system is difficult. Firstly we need to devise a feedback loop to regularly annotate the cluster groups derived in this approach. Once this annotation has been done one can devise a semi supervised learning approach which uses previously tagged clusters in the training phase. A portion of this annotated set can also be used for testing phase. An attempt can be made to progress from feature learning to direct classification of ornaments while taking the advantages

of feature learning. Once such a model is trained which can differentiate between ornaments at a great hierarchical level an online ornament tagging system can be built.

An online ornament tagging system can be built in two phases:

- Offline Phase: In this phase we are interested in training a model which can differentiate between ornaments as discussed before. We should also store a mapping between the learned feature representation and the ornament class.
- Online Phase:
  - Now given an audio file as input, it can be segmented in the same way as described in this chapter. We can then process the returned list of intervals individually.
  - An ornament interval is given to the trained sequence to sequence autoencoder which returns a compressed representation.
  - This compressed representation can then be compared with the centroids of tagged alankār groups and then be matched with the one with the highest cosine similarity.
  - Instead of matching with the vector that gives highest cosine similarity one can even return the ranked list of closest matching centroids.
  - This reduces the time complexity significantly compared to iterative template matching since the representation of ornaments are much more concise than pitch representations.

Chapter 6

Conclusions

Ornamentation is an indispensable part of improvisation. The performer tries to evoke the mood of the raga (melodic-mode) with the delicate exploration of flourishes and ornamentation. Indian classical music is cognitively associated with these decorations instead of the actual underlying notes. In this thesis we make an attempt to study these ornaments computationally. We begin the study with the objective to computationally detect these ornaments in Dhrupad performances on rudra veena. We were successfully able to deploy the dynamic time warping technique to detect ornaments to a large extent. However, we saw that this method of iterative template matching is naive and has drawbacks, some of which are:

- Restrictiveness of the experiment to a set of non-exhaustive list of ornament templates.
- Computational overheads of comparing huge pitch vectors using dynamic time warping.
- Dynamic time warping is still very rigid in terms of the constraints. No elements can be omitted and no replications are allowed in the alignment.

While DTW matching was an effective method of detecting these ornaments, we didnt use any domain specific knowledge to detect/identify these ornaments computationally. We have proposed that the rate of change in frequency and amplitude domains together constitute the fundamental nature of ornamentation. We thus modeled ornamentation as first and second derivative of frequency and amplitude information to show that the property of rate of change is fundamental to the idea of flows. We were able to achieve very high clustering accuracies by using this representation and our custom defined distance measure. We achieved an accuracy of 81% in distinguishing between Murki vs Kan and 84.5% in Andolan vs Ghaseet.

In all of our previous experiments we had small annotated datasets which did not exhaustively capture patterns underlying ornaments. Thus we attempted to have a big data driven exhaustive exploration of these melodic structures. To avoid the shortcomings of non-exhaustive small datasets and hand engineered representations, we developed a robust unsupervised method to learn fixed length representations of ornaments. Drawing from existing research we have seen that it is possible to transform audio segments into fixed dimensional vectors. In this thesis we extended the same model for learning fixed

length features using sequence to sequence autoencoder to represent audio segments of ornaments and flows in musical data without human annotation. We had started this study to have a big data exploration of ornaments in Indian music but we discovered that ornaments are not present in an isolated form in most Indian music performances. Thus its important to study ornaments as a combination of multiple melodic structures. We have also discovered that some ornaments have similar musical properties as some others which is why they are grouped into the same category. For instance, certain sequences of murkis were grouped along with meends which indicates high similarity between these groups. This might be because of the partially similar temporal change properties (in frequency domains) in both these groups. Further, we have also noticed that musical segments based on taala and rhythm form separate clusters (groupings) in all time groups. Which means that taala-samparkit alankaar can very well be distinguished from the rest of the ornaments.

We have shown that these learned audio representations capture essential sequential structural information and are useful in distinguishing between different types of ornaments (recorded in isolation) in the vector space. We also finally discussed about how this model can be used for real time ornament transcription systems, ornament retrieval and tagging. This further has great applications in Indian music pedagogy, raga classification, music search, retrieval and recommendation.

Chapter 7

Glossary

- Amplitude: The maximum extent of a vibration or oscillation, measured from the position of equilibrium.
- Autoencoder: Its an artificial neural network used for unsupervised learning of efficient codings.
- Alap: Alap is the opening section of a typical North Indian classical performance which introduces the raga.
- Andolan(a): The Andolan alankar is a gentle swing or oscillation that starts from a fixed note and touches the periphery of an adjacent note.
- Aroh(a): Sequence of the notes in ascending order.
- Avaroh(a): Sequence of the notes in descending order.
- Bandish: Its a fixed, melodic composition in Hindustani vocal or instrumental music.
- Back-propagation: Its an algorithm for supervised learning of artificial neural networks using gradient descent
- Dhrupad: Its one of the oldest genres in Hindustani classical music.
- Equi Tempered scale: It is a musical tuning in which every pair of adjacent pitches is separated by the same interval.
- Euclidean Distance: It is the straight-line distance between two points in Euclidean space.
- False Positive: These are the negative data samples that were incorrectly labeled as positive by a classifier.
- False Negative: These are the positive data samples that were incorrectly labeled as negative by a classifier.

- Feedforward Neural Network: Its the simplest form of artificial neural networks wherein connections between the units do not form a cycle.
- Frequency: The rate at which something occurs over a particular period of time or in a given sample.
- Gamak: Gamak is a certain swaying or oscillation of the pitch of a note so that it goes slightly off its true pitch for a moment and returns to its original pitch immediately.
- Ghaseet: This is a kind of meend specific to stringed instruments. While its literal meaning translates to pull, it usually refers to a fast glide.
- GRU: Gated recurrent units (GRUs) are a gating mechanism in recurrent neural networks.
- Harmony: A combination of simultaneously sounded musical notes to produce a pleasing effect.
- Hindustani Music: A major tradition in Indian music that emerged out of the Northern part of India.
- Interval: In music theory, an interval is the difference between two pitches.
- Just Intonated scale: It is a musical tuning in which the frequencies of notes are related by ratios of small whole numbers.
- Jor: Part of a performance in Dhrupad which follows the Alap and has a regular rhythmic pulse.
- Jhala: Part of a performance in Dhrupad which follows the Jor and features very rapid rhythmic patterns.
- Kan: Kan are the linking or grace notes. They are played in a very subtle manner.
- Krintan: Grace note.
- MAP: Mean Average Precision (MAP) is a very popular performance measure in information retrieval.
- Monophony: Monophony is the simplest of musical textures, consisting of a melody, typically sung by a single singer or played by a single instrument player.
- Meend: Meend refers to the glissando or a glide in the wave from one note to another.
- Melody: The aspect of musical composition concerned with the arrangement of single notes to form a satisfying sequence.
- Melodic motif: A dominant or recurring pattern.
- Metre: rhythmic structure.

- Microtone: An interval smaller than a semitone.
- Neural Networks: A computer system modelled on the human brain and nervous system.
- Octave: A series of eight notes occupying the interval between (and including) two notes, one having twice or half the frequency of vibration of the other.
- Onset: Onset refers to the beginning of a musical note or other sound.
- Ornament: Ornaments are musical flourishes that enhance the beauty of a performance.
- Pitch: The degree of highness or lowness of a tone.
- Raga: Raga is akin to a melodic mode in Indian classical music.
- Raga-roopa: Characteristics/Personality of the raga.
- Rasa: Rasa is the aesthetic flavor of any visual, literary or musical work, that evokes an emotion or feeling in the audience, but it cannot be described.
- RNN: A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed cycle.
- Rudra-veena: Its a large plucked string instrument used in Hindustani classical music.
- Samavadi: Samavadi is the second most important note in the raga.
- Sargam: Sargam refers to singing the notes instead of the words of a composition
- Sarod: Its a stringed instrument of India, used mainly in Indian classical music.
- Segmentation: Process of splitting a sound signal into intervals of interest.
- Semitone: The smallest interval used in classical Western music, equal to a twelfth of an octave or half a tone.
- Shruti: The smallest interval of pitch that the human ear can detect.
- Taala: Taala is the metrical structure of a musical performance.
- True Positive: These refer to the positive data samples that were correctly labeled by the classifier.
- True Negative: These are the negative data samples that were correctly labeled by the classifier.
- Vadi: Vadi is the tonic (root) swara (musical note) of a given raga .
- Varna: A varna is a short group of notes that gets its meaning in the context of a musical phrase.

Chapter 8

Related Publications

[1] Full Paper Title (Accepted and Published): DETECTION OF MICRO-TONAL ORNAMENTATIONS IN DHRUPAD USING A DYNAMIC PROGRAMMING APPROACH
Conference: 9th Conference on Interdisciplinary Musicology CIM14. Berlin, Germany 2014
Authors: Achyuth Narayan, Navjyoti Singh
Link: https://www.sim.spk-berlin.de/cim14_919.html

[2] Full Paper Title (Accepted and Published): DEMYSTIFYING FLOWS BASED ON TRANSITIONAL PROPERTIES OF IMPROVISATION IN HINDUSTANI MUSIC
Workshop: 7th International Workshop on Folk Music Analysis. Malaga, Spain 2017
Authors: Achyuth Narayan, Navjyoti Singh
Link: http://fma2017.uma.es/programfma2017.html

[3] Full Paper Title (Submitted, Under Review): Sequence to Sequence Autoencoder for Learning Representations of Ornaments
Conference: The 24th International Conference on Multimedia Modeling. Bangkok, Thailand 2018
Authors: Achyuth Narayan, Navjyoti Singh
Link: http://mmm2018.chula.ac.th

Bibliography

[1] E. e. a. Azarov. Instantaneous pitch estimation based on rapt framework. In *Proceedings of the 20th European Signal Processing Conference (EUSIPCO'2012)*, pages 2787–2791, Bucharest, Romania, 2012.

[2] S. Bagchee. Nad: understanding raga music. *Eeshwar - illustrated edition*, 1998.

[3] P. Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Unsupervised and Transfer Learning Challenges in Machine Learning*, volume Volume 7, page 43, 2012.

[4] S. Bengio and G. Heigold. Word embeddings for speech recognition. In *INTERSPEECH*, 2014.

[5] V. N. Bhatkhande. Hindusthani sangeet paddhati. *Sangeet Karyalaya*, 1934.

[6] C. C. K. B. Y. Chung, Junyoung; Gulcehre. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *ArXiv:1412.355*, 2014.

[7] M. K. Doetsch and H. Ney. Fast and robust training of recurrent neural networks for offline handwriting recognition. In *ICFHR*, 2014.

[8] B. J. F. D. Dueck. Clustering by passing messages between data points. *Science*, 315:972976, 2007.

[9] S. Dutta and H. A. Murthy. Discovering typical motifs of a raga from one- liners of songs in carnatic music. In *Proc. of Int. Soc. for Music Information Retrieval (ISMIR)*, page 397402, 2014.

[10] S. Dutta and H. A. Murthy. A modified rough longest common subsequence algorithm for motif spotting in an alapana of carnatic music. In *Proc. of Twentieth National Conference on Communications (NCC)*, page 16, 2014.

[11] C. D. et al. Complex domain onset detection for musical signals. In *Proc. Int. Conf. Digital Audio Effects*, London, U.K, 2003.

[12] J. Flanagan. Speech analysis, synthesis and perception. *Springer- Verlag*, 1972.

[13] E. B. Fowkles and C. L. Mallows. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association.*, 1983.

[14] C. P. G. Chen and T. N. Sainath. Query-by-examplekeyword spotting using long short-term memory networks. In *Proceedings of ICASSP*, 2015.

[15] R. A. P. V. K. P. . R. P. Ganguli, K. K. Efficient melodic query based audio search for hindustani vocal compositions. In *Proc. of Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, page 591597, 2015.

[16] T. Giannakopoulos. pyaudioanalysis: An open-source python library for audio signal analysis. *PloS one*, 10(12), 2015.

[17] S. Gulati. *Computational Approaches for Melodic Description in Indian Art Music Corpora*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2016.

[18] S. Gulati, J. Serrà, V. Ishwar, and X. Serra. Discovering rāga motifs by characterizing communities in networks of melodic patterns. In *41st IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2016)*, pages 286–290, Shanghai, China, 20/3/2016 2016. IEEE, IEEE.

[19] C. Gupta and P. Rao. Objective assessment of ornamentation in indian classical singing. In *CMMR'11 Proceedings of the 8th international conference on Speech, Sound and Music Processing: embracing research in India*, pages 1–25, Berlin, Heidelberg, 2012. Springer-Verlag.

[20] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. In *Science*, volume Volume 313, no. 5786, page 504507, 2006.

[21] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2: 193, 1985.

[22] O. V. I. Sutskever and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

[23] B. A. . M. H. A. Ishwar, V. Motivic analysis and its relevance to raga identification in carnatic music. In *Proceedings of the 2nd CompMusic Workshop*, page 153157, 2012.

[24] D. S. B. A. . M. H. Ishwar, V. Motif spotting in an alapana in carnatic music. In *Proc. of Int. Conf. on Music Information Retrieval (ISMIR)*, pages 499–504, 2013.

[25] V. T. Joe Cheri Ross and P. Rao. Detecting melodic motifs from audio for hindustani classical music. In *Proc. 13th International Society for Music Information Retrieval Conference*, Porto, 2012.

[26] L. Kaufman and P. Rousseeuw. Clustering by means of medoids. *Statistical Data Analysis Based on the L 1 Norm and Related Methods*, page 405416, 1987.

[27] Q. V. Le and T. Mikolov. Distributed representations of sentences and documents. In *arXiv preprint*, volume arXiv:1405.4053, 2014.

[28] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

[29] B. Mukerji. *An analytical study of improvisation in hindustani classical music*. PhD thesis, University of Delhi, November 2014.

[30] M. Mller. *Information Retrieval for Music and Motion*. Springer, Berlin, Heidelberg, 2007.

[31] A. Narayan and N. Singh. Detection of micro-tonal ornamentations in dhrupad using a dynamic programming approach. In *Proc. 9th Conference on Interdisciplinary Musicology CIM14*, Berlin, 2014.

[32] A. Narayan and N. Singh. Consonance of micro-tonal ornamentation in melodic contexts. In *Proc. of the 11th International Symposium on CMMR*, Plymouth, UK, 2015.

[33] A. Narayan and N. Singh. Demystifying flows based on transitional properties of improvisation in hindustani music. In *Proceedings of The 7th International Workshop on Folk Music Analysis*, Malaga, Spain, 2017.

[34] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS*, pages 849–856. MIT Press, 2001.

[35] I. L. Y. B. P. Vincent, H. Larochelle and P.-A. Man-zagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. In *JMLR*, volume 11, pages 3371–3408, 2010.

[36] Pratyush. Analysis and classification of ornaments in north indian (hindustani) classical music. Master's thesis, Universitat Pompeu Fabra, 2010.

[37] R.-J. Rao, P. Classification of melodic motifs in raga music with time-series matching. In *Journal of New Music Research*, volume 43(1), page 115131, 2014.

[38] L. Rokach and O. Maimon. Clustering methods. In *Data mining and knowledge discovery handbook*, pages 321–352. Springer US, 2005.

[39] A. Rosenberg and J. Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[40] J. C. Ross and P. Rao. Detection of raga-characteristic phrases from hindustani classical music audio. In *Proc. of 2nd CompMusic Workshop*, page 133138, 2012.

[41] P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Computational and Applied Mathematics*, 20:53–65, 1987.

[42] R. B. S. S. Miryala, K. Bali and M. Choudhury. Automatically identifying vocal expressions for music transcription. In *Proc. ISMIR*, pages 239–244, Brazil, 2013.

[43] Sanyal and Widdess. *Dhrupad: Tradition and Performance in Indian Music*. Aldershot, Ashgate, 2004.

[44] scikit-learn developers. Adjustment for chance in clustering performance evaluation.

[45] A. Singhal. Modern information retrieval: A brief overview. In *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, volume 24 (4), page 3543, 2001.

[46] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. In *Journal of molecular biology*, page 195197, 1981.

[47] A. Strehl and J. Ghosh. Cluster ensembles a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

[48] G. C. T. Mikolov, K. Chen and J. Dean. Efficient estimation of word representations in vector space. In *arXiv preprint*, volume arXiv:1301.3781, 2013.

[49] C.-H. S. H.-Y. L. L.-S. L. Yu-An Chung, Chao-Chung Wu. Audio word2vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder. In *arXiv preprint*, volume arXiv:1603.00982, 2016.