EXPLORING DEEP LEARNING BASED METHODS FOR INFORMATION RETRIEVAL IN INDIAN CLASSICAL MUSIC

BY

SATHWIK TEJASWI MADHUSUDHAN

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Industrial Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urbana, Illinois

Adviser:

Assistant Professor Girish Chowdhary
Associate Professor Ramavarapu Sreenivas

ABSTRACT

A vital aspect of Indian Classical Music (ICM) is *Raga*, which serves as a melodic framework for compositions and improvisations alike. Raga Recognition is an important music information retrieval task in ICM as it can aid numerous downstream applications ranging from music recommendations to organizing huge music collections. In this work, we propose a deep learning based approach to Raga recognition. Our approach employs efficient pre-possessing and learns temporal sequences in music data using Long Short Term Memory based Recurrent Neural Networks (LSTM-RNN). We train and test the network on smaller sequences sampled from the original audio while the final inference is performed on the audio as a whole. Our method achieves an accuracy of 88.1% and 97 % during inference on the *Comp Music Carnatic dataset* and its 10 Raga subset respectively making it the state-of-the-art for the Raga recognition task. Our approach also enables sequence ranking which aids us in retrieving melodic patterns from a given music data base that are closely related to the presented query sequence.

To my parents and my brother, for their love and support.

ACKNOWLEDGMENTS

I'd like to acknowledge everyone at Distributed Autonomous Systems Lab at the University of Illinois, Urbana Champaign for their help and thoughtful suggestions. I'd also like to thank my friends Kartik Hegde and Rahul Mahadev for their support and encouragement.

I would like to thank my Advisers Dr. Girish Chowdhary and Dr. Ramavarapu Sreenivas for constant feedback and suggestions during the course of my research. I would also like to thank Dr Sankalp Gulati and the Comp Music group for providing the Raga recognition Music Dataset to train and evaluate our models.

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION 1
1.1 Indian Classical Music 1
1.2 Raga in ICM 1
1.3 Motivation 3
1.4 Outcomes and Contributions 3

CHAPTER 2 TECHNICAL BACKGROUND AND RELATED WORK 5
2.1 Music Information Retrieval 5
2.2 Preprocessing Techniques 5
2.3 Related Works 8

CHAPTER 3 RAGA RECOGNITION AND SEQUENCE RANKING 12
3.1 Raga Recognition as a Sequence Classification Problem 12
3.2 Sequence Ranking and Retrieval 16
3.3 Learning to Classify Independent of the Tonic 18

CHAPTER 4 DATASET, INFERENCE AND EVALUATION METHODOLOGY . 21
4.1 Dataset 21
4.2 Inference 21
4.3 Evaluation Strategy 21

CHAPTER 5 RESULTS AND DISCUSSION 24

CHAPTER 6 CONCLUSION AND FUTURE WORK 27
6.1 Conclusion 27
6.2 Future Work 27

CHAPTER 7 REFERENCES 28

CHAPTER 1: INTRODUCTION

1.1 INDIAN CLASSICAL MUSIC

Indian Classical Music (ICM) is an ancient form classical music practiced in the Indian subcontinent. There are two major branches of ICM that are widely practised to date, namely, Carnatic music (CM) and Hindustani Music (HM) predominant in southern and northern regions of the country respectively. HM primarily focuses on improvisation with compositions being only a small portion of concerts. However, CM differs in this regard as it places a significant amount of importance on compositions despite improvisations being a key aspect. There are many theories that indicate that both CM and HM were indistinguishable prior to the 16th century. Incidentally both the art forms share a number of melodic and rhythmic concepts. These art forms, being spiritual and contemplative in nature, place a lot of importance on melodic development. Also, major portion of contemporary Indian Music including film, folk and other forms of music heavily draw inspiration from ICM [1, 2].

1.2 RAGA IN ICM

Raga, which governs various melodic aspects of ICM, serves as a framework for compositions and improvisations in both forms of music. [3, 4, 5] provide a wholesome technical definition of Raga as "Raga is a collection of melodic atoms and a technique for developing them. These melodic atoms are sequences of notes that are inflected with various micro pitch alterations and articulated with expressive sense of timing. Longer musical phrases are built by knitting these melodic atoms together". Over the last few decades, a popular technique for comprehending a Raga has been to separate it into it various melodic atoms like svara, scale, gamaka, and phrases [6]. We argue that understanding these melodic atoms, especially svara and gamaka, becomes very critical if one has to device a machine learning techniques based on Raga for music information retrieval.

Numerous melodic attributes make Ragas distinctive in nature, such as the *svara* (roughly, a musical note), the *gamaka* (for example oscillatory movement about a given note, slide from one note to another etc [7]), *arohana* and *avarohana* (upward and downward melodic movement) and melodic phrases/ motifs [2, 6]. Emotion is another attribute that makes Ragas distinctive. Balkwill et al[8] conducted studies on the perception of Ragas and observed that even an inexperienced listener is able to identify emotions portrayed by various Ragas.

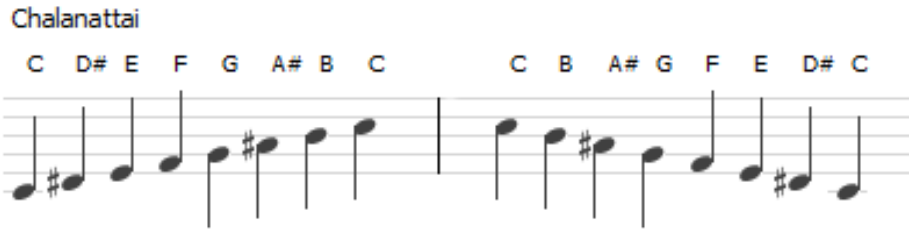

Figure 1.1: Figure depicts the Raga "Chalanattai" (used in CM) in the western music notation. Its not hard to observe that the left half of the notation shows the notes in an ascent direction while in the second half the notes are in descent. The ascent and descent are called Arohana and Avarohana respectively. Note the Tonic in this example is C

Few of the basic melodic attributes of the Raga have been delineated below.

Svara

As a manner of common parlance, a melodic note in the setting of Indian classical music can be loosely referred to as a svara. The major difference between a musical note (as in western music) and a svara (in ICM), is that the identity of a svara is dependent on another entity called the *Tonic*. A Tonic is also musical note, which acts as a datum using which the svaras are defined. Both HM and CM have 7 unique svaras namely, *Shadja*, *Rishaba*, *Gandhara*, *Madhyama*, *Panchama*, *Daivata* and *Nishada*. In Figure 1.1, the notes C, D#, E, F, G, A#, B and C correspond to these 7 notes, taken in order.

Gamaka

A gamaka can be understood as a type of melodic movement about one or more svaras using musical ornaments like trill, glissando, mordent, slide etc. Various ancient treatises on Carnatic music mention as many as 15 types of gamakas. Works like [6, 9] etc indicate that gamakas have undergone a lot of changes, as a result of which one would be able to spot 6-7 variations of gamakas as of today. Instances of different Ragas, which make use of similar svaras, having contrasting musical impact are in abundance. This can be largely attributed to the usage of different gamakas while maneuvering around the set of notes in different Ragas. Hence, It is safe to assume that Gamaka is a one of the major factors which help in identifying Ragas.

Arohana and Avarohana

Arohana and arohana literally translate to ascent of notes and descent of notes. Arohana and avarohana of a Raga provide the legal combination of notes that can be used in a composition/improvisation during ascent or descent of notes respectively.

1.3 MOTIVATION

A bulk of Indian film music compositions use Raga (either one Raga or a combination of Ragas) as a melodic framework[1]. Many of the popular western music scales are widely used in ICM as well (for instance, major scale in western music is called shankarabharana in CM). Hence, we strongly believe *Automatic Raga Recognition* has tremendous potential in empowering content based recommendation systems pertaining ICM and contemporary Indian music. However, Raga recognition is not a trivial problem. There are numerous examples where two or more Ragas have the same or a similar set of notes but are worlds apart in the musical effect they produce due factors like the gamaka, temporal sequencing (which has to abide by the constraints presented in the arohana and avarohana), as well as places of svara emphasis and rest.

As such, automatic Raga classification methods have been widely studied. However, existing methods based on Pitch Class Distributions (PCD) disregard the temporal information and are highly error prone, while other methods (reviewed in detail in Chapter 2) are highly dependent on preprocessing, Tonic of the audio and hand made features, which limit the performance of such methods. Recent success of deep learning methods in various tasks like sequence classification, neural machine translation, neural image captioning etc. demonstrate deep learning methods' flexibility and ability in handling unstructured data. In this work, we address Raga recognition and Raga content based retrieval using deep learning.

1.4 OUTCOMES AND CONTRIBUTIONS

This work deals with the problem of Automatic Raga Recognition and Raga content based recommendation systems. First, we reformulate the problem of Raga recognition as a sequence classification task performed using an LSTM RNN based architecture with attention. Next, we explore solutions for the same problem using convolutional neural network based approaches. We then show that by employing a simple data augmentation technique, the Raga Recognition model can be made to function independent of the Tonic of the audio ex-

cerpt. Finally, we show that the LSTM RNN network, with minimal fine tuning based on the triplet margin loss [10], can be used for sequence ranking. We introduce sequence ranking as a new sub-task of automatic Raga recognition. In this configuration, the model can be used to perform Raga content based retrieval. A user provides the model with a query sequence and the model returns sequences that are very closely related to the presented query. This can aid in downstream applications like recommendation systems and so on.

To summarize, the main contributions of this work is as follows :

- We present a new approach to Raga recognition using deep learning using LSTM based RNN architecture to address problem of Raga recognition.
- With our approach we obtain 97.1 % on the 10 Raga classification task and 88.1 % accuracy on the 40 Raga classification task on the Comp Music Carnatic Music Dataset (CMD), hence improving the state-of-the-art on the same.
- We present a novel data augmentation technique which enables the model to perform Raga Classification irrespective of the Tonic.
- We introduce sequence ranking as a new sub task of Raga recognition, which can be used in creating Raga content based recommendation systems.

Note that we refer to our Raga recognition (i.e sequence classification) model as SRGM1 and the sequence ranking model as SRGM2 throughout the paper.

CHAPTER 2: TECHNICAL BACKGROUND AND RELATED WORK

This chapter presents a review on music information retrieval, various preprocessing techniques as applicable to the problem of Raga recognition and a summary of previous works on Raga recognition.

2.1 MUSIC INFORMATION RETRIEVAL

Music information retrieval (or MIR for short), is an interdisciplinary field that combines concepts from varied fields like music theory, signal processing, machine learning, information retrieval etc. Most popular application of music information retrieval is undoubtedly recommendation systems. Numerous companies like Spotify, Pandora and Google Play Music have their business built around recommendation systems. Apart from recommendation systems, MIR can be used across a host of music related tasks like categorizing and organizing music libraries, music tagging, music generation and manipulation, genre recognition and so on. In recent times, with the advent of machine learning and deep learning, MIR tasks like music generation, automatic music transcription, audio source separation and related tasks are garnering a lot of attention.

Music recommendation systems can be coarsely classified into two variants, meta data based recommendation systems and content based music recommendation systems. Meta data based recommendation systems work based off of meta data like genre, artist, instrumental / non-instrumental, year of creation, the lyrics of the song, user rating etc. A major drawback of such methods is that they ignore the music content of a song. Content based music recommendation systems analyze the musical content i.e, melody, rhythm etc and hence are naturally much more effective in making recommendations. In this work, we explore ways to extract melodic information from ICM audio excerpts which will can help in various downstream applications like automatic music tagging and content based music recommendation.

2.2 PREPROCESSING TECHNIQUES

In this Section, we discuss the importance of preprocessing techniques such as the fundamental frequency estimation and issues related to the same in brief. Later, we go on to highlight the significance of audio source separation and the benefits of using audio source

separation in the preprocessing pipeline. We then describe a neural network based audio source separation technique called MaDTwinNet [11] which we have used as part of our model.

2.2.1 Fundamental Frequency Estimation

A critical concept that characterizes the tonal aspect of sound is the frequency, often perceived as pitch. It is one among many basic dimensions of sound like loudness, timbre etc. Majority of the audio wave forms can be represented as a superposition of complex sinusoids, where each of these sinusoids are called as partials or harmonics. They are called so, since they are is a harmonic relationship with the fundamental frequency. Since pitch of an audio is related to the fundamental frequency, the problem of fundamental frequency estimation might often be interchangeably referred to as pitch estimation.

Fundamental frequency estimation (FFE) is an important step in analyzing audio or speech data. Consequently, there have been numerous methods that focus on FFE. The algorithms for FFE can be broadly classified as time domain, frequency domain and a combination of both time and frequency domain approaches. We see that most of the methods for FFE fall into the time domain (lag-based) based approaches. [12] describes an auto-correlation function (ACF) based approach for pitch detection. [13] proposed numerous modifications to the ACF based method making it robust to white noise. [14] address issues related to the resolution of the pitch estimate arising from finite sampling rate and analysis window size. One of the most popular methods for FFE is the YIN algorithm[15] which significantly reduces the error rates in ACF based methods. There are numerous works based on spectrum auto-correlation, subharmonic summation and harmonic matching as well.

Although some of these methods may be applicable under polyphonic settings (an audio containing more than one source of sound), a large number of these methods work well only under monophonic settings. Hence, these methods are not suitable for obtaining the fundamental frequency from music since music is mostly heterophonic. Works like [16, 17, 18] provide solutions for FFE under polyphoic audio settings. In this work, we first use audio source separation techniques [19] to obtain audio that only contains vocals, following which we perform FFE. Since we use audio source separation as a preprocessing step, the estimated fundamental frequency is less erroneous.

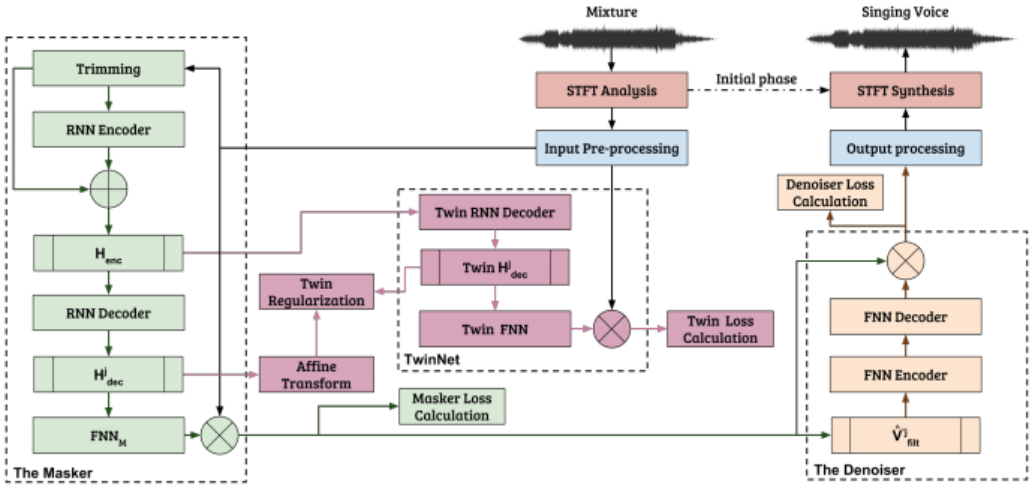

Figure 2.1

2.2.2 Audio Source Separation using MaDTwinNet

MadTwinNet [11] (which stands for Masker-Denoiser Architecture with Twin Networks for Monaural Sound Source Separation) is a deep learning based approach for audio source separation. It takes the raw audio signal of a mixture of sources as an input and outputs the raw audio signal of the target source (i.e. the vocals). The first part, called as Masker, takes the magnitude spectrogram of the audio as an input, estimates the target source magnitude spectrogram from the mixture by predicting a time frequency filter and applying it to the input, and outputs the estimated magnitude spectrogram of the target source. The second part, called as Denoiser, takes the output of the first part as an input, predicts and applies a denoising mask, and outputs a filtered version of its input.

Since the input to the first part is the mixture signal and the output is an estimate of the signal of the target source, the time-frequency filter of the first part is framed as a time-frequency mask. Similarly, since the input to the second part is a representation of the target source and its output is the same representation of the same source, the time-frequency filter of the second part is framed as a denoising filter. A detailed block diagram of the network is shown as in Figure 2.1. We use the pretrained weights for the model, released by authors (https://github.com/dr-costas/mad-twinnet). This is means that we freeze the model weights while training, although the audio source separation model can be retrained for better results.

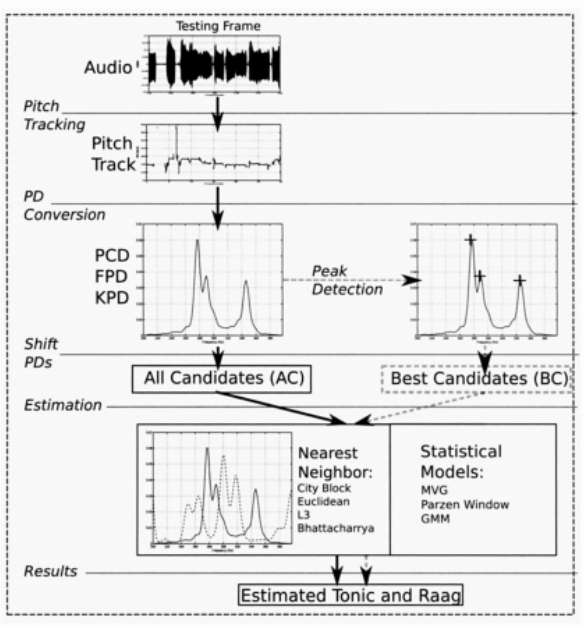

Figure 2.2: Block diagram for joint recognition of Tonic and Raga [21]

2.3 RELATED WORKS

Automatic recognition has been widely studied in the past using various approaches, of which a vast majority is based on machine learning. Broadly speaking, ML methods for Raga classification can either be Pitch class distribution based (PCD) or sequence based. PCD methods would be an intuitive method for Raga recognition since a PCD is a 12 dimensional vector (also referred to as bins) with values indicating the amount of energy present in the audio corresponding to each of the 12 semitones [20]. Pitch-class profiles (PCP), Harmonic pitch-class profiles (HPCP) are sometimes used interchangeably with PCD, although there are a few differences. Sequence based methods consider an audio as a time varying function and hence try to extract various patterns from the audio excerpts through which an inference on the Raga can be drawn.

Works like [22, 23, 24] use PCD based methods to perform Raga recognition with Chordia et al [21], currently holding the best performing results for PCD based methods with an accuracy of 91%. The main objective of their work is to to automatically determine the Raga and the Tonic of a given audio sample. Their experiments invlove audio excerpts of length 30 seconds. Figure 2.2 a) outlines the approach followed [21]. As a first step, they pitch track the given audio excerpt resulting in an array pitch values as would be observed in the audio. They array of pitch values is then condensed to a single octave and is summarized by

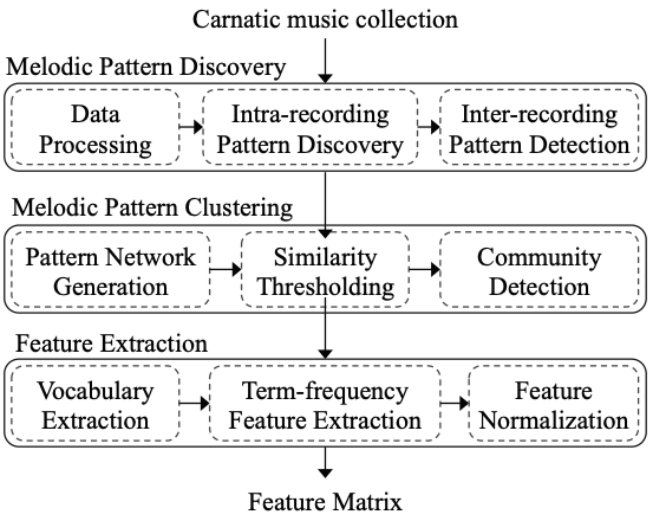

Figure 2.3: Outline of phrase based vector space modelling for Raga recognition [25]

a pitch distribution. This is used a feature vector to perform Raga and Tonic classification. Methods like nearest neighbour etc are then used to compare the feature vector with the feature vectors of the training set yeilding the Tonic and Raga.

One of the main drawbacks of this work is that they assume the audio to be monophonic, i.e, void of any accompaniments. This assumption limits the applicability of the method in real world scenarios. [7] presents an in depth study on various distribution based methods, their limitations and a comparison to investigate how well each of these fair on the same dataset . Although an intuitive and an effective approach, the major shortcoming of PCD based methods is that it disregards temporal information and hence is error prone. To overcome the limitations of PCD based methods, previous works have used Hidden Markov Models based approach[26], n-gram based approach [27], arohana avarohana based approach [28] among various other methods.

[29] use a novel longest common segment set (LCSS) based approach to Raga classification. The work tries to mimic the way in which a listener identifies a Raga. Their model, reffered to as the verification system, assumes that a Raga is claimed based on similarity of movements and motivic patterns. For every Raga claim made, the model chooses a set of cohorts. The system then represents every Raga using the note transcriptions of compositions from that particular Raga, following which LCSS method is used. Although an interesting approach, this method requires accurate note transcriptions of compositions. The method also requires human intervention in choosing the cohorts and since the human has

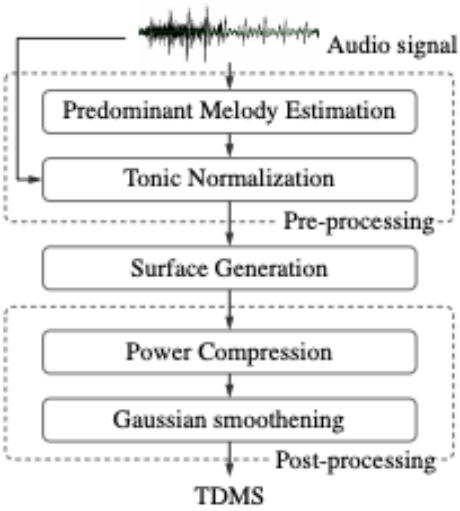

Figure 2.4: Block diagram for Time Delayed Melodic Surface (TDMS) based approach to Raga Recognition [2]

to be a skilled musician, this could prove to be a huge disadvantage when the model is to be extended to new Ragas.

[25] use an interesting phrase based approach inspired by the way in which seasoned listeners identify a Raga, with the first step being, the unsupervised extraction of melodic patterns from a library of ICM audio. Leveraging complex concepts, authors then group and cluster similar patterns from the set of extracted melodic patterns. Using a similar concept to topic modelling in NLP literature, the melodic patterns are then transformed into a vector space representation. The authors their method on the Comp Music, Carnatic Music dataset (CMD) [25]. This method is currently the state-of-the-art on a 10 Raga subset of CMD. A block diagram of this method is shown in Figure 2.3. The entire method can be broken down into 3 main parts, namely melodic pattern discovery, melodic pattern clustering and feature extraction which results in the creation of a feature matrix. Methods like Support vector machines, random forests etc can later be used to perform classification.

[2] introduces Time Delayed Melodic Surface(TDMS) which is feature based on delay coordinates that captures the melodic outline of a given audio excerpt. By using the predominant pitch of an audio, it is capable of describing both tonal and temporal characteristics of the melody. TDMS is a rich representation of the melody of the given audio and hence, a simple classifier such as the K nearest neighbour classifier is able to achieve state-of-the-art results on the CMD and Comp Music's Hindustani Music Dataset (HMD). It acheives an

accuracy of 98% on the HMD dataset and an accuracy of 87% on the CMD dataset. The authors cite that the TDMS features are simple and easy to compute. The block diagram that indicates various steps for creating a TDMS is as in Figure 2.4

While the PCD based methods ignore the temporal information, many works like assume that the audio is monophonic. Works like [26, 28] attempt to take advantage of the temporal information by extracting arohana and avarohana. Although arohana and avarohana are critical components to identifying a Raga, they do not contain as much information as the audio excerpt itself. Most previous works heavily rely on prepossessing and feature extraction, most of which are hand crafted, which can prove to be a limitation as it only gets harder to device ways to extract complex and rich features.

CHAPTER 3: RAGA RECOGNITION AND SEQUENCE RANKING

In this chapter we present our approach to the problem of Raga Recognition. First, we describe a novel approach to Raga recognition by framing it as a sequence classification problem. We then outline the procedure to convert the sequence classification-Raga recognition model as a sequence ranking model by introducing the triplet sampling sampling layer and subsequently fine tuning the network based on the triplet margin loss. Finally, we introduce a novel data augmentation technique to enable the Raga recognition model to perform Raga recognition independent of the Tonic of the Raga. (Note that we refer to the Raga recognition model, sequence raking model and Tonic independent Raga recognition model as SRGM1, SRGM2 and SRGM3 repectively.)

3.1 RAGA RECOGNITION AS A SEQUENCE CLASSIFICATION PROBLEM

Sequence classification is a predictive modelling problem where an input, a sequence over space or time, is to be classified as one among several categories. In modern NLP literature, sentiment classification is one such example, where models predict sentiments of sentences. We argue that the Raga recognition is closely related to the task of sequence classification, since any audio excerpt in ICM involving vocals/melody instruments, can be broken down as a sequence of notes spread over time. To reformulate Raga recognition as a sequence classification task, we treat the notes obtained via predominant melody estimation as words, the set of all words form the vocabulary and each Raga as a class. We detail our approach to Raga recognition as below.

3.1.1 Preprocessing

Deep Learning based audio source separation

Audio recordings available as part of CMD accurately represent live ICM concerts with vocalists accompanied by various string, wind and percussion instruments. However, for the purposes of Raga recognition, analyzing the vocals is sufficient. We believe that the presence of other elements in the audio interferes with the performance of the classifier and hence we include audio source separation as a preprocessing step. We use Mad-TwinNet [11], a deep neural network based approach capable of recovering vocals from a single channel track comprised of vocals and instruments. It uses a Masker Denoiser architecture combined with

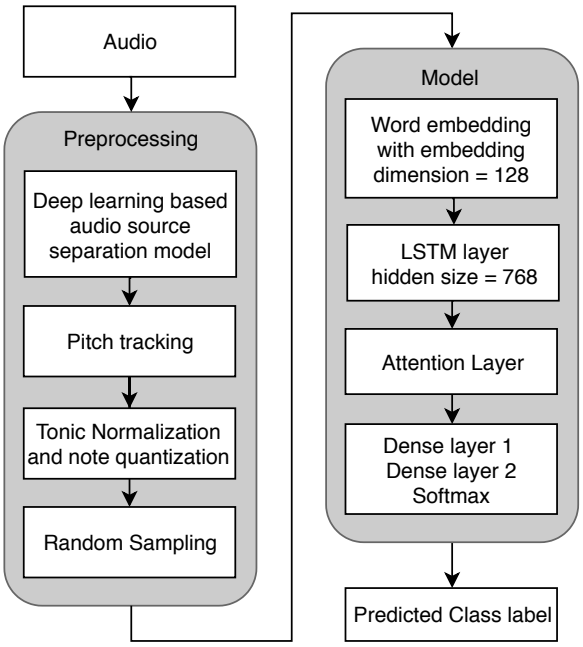

Figure 3.1: Figure shows various preprocessing steps and model architecture for SRGM1 (refer Section 3)

twin networks (a technique used to regularize recurrent generative networks). We use the pretrained model made publicly available by the authors.

Pitch Tracking

Our model analyzes the melody content of audio excerpts to perform Raga recognition, hence pitch tracking is an essential preprocessing step. To perform pitch tracking, we use the python api [30] for praat [31] which is an open source software for the analysis of speech and sound.

Tonic Normalization and Note Quantization

ICM utilizes a relative scale that is based on the Tonic note of the performer. Since the Tonic can vary in different renditions of the same Raga, recognizing and accounting for the Tonic note is extremely important. [25] have made numerous features, including Tonic of audio excerpts, available as part of CMD. We normalize the pitch tracked array of every audio in CMD by using the following:

$$f_n = 1200 * log_2(f/T) \tag{3.1}$$

where f is the frequency after pitch tracking (in Hz), T is the frequency of the Tonic (in Hz) and f_n is the Tonic normalized frequency. The above expression results in 100 cents in a half step, for instance E and F. Empirically we observe that it is sufficient to have just 5 levels in a half step. However, we leave this as a hyper parameter on the model. Hence the expression for obtaining a Tonic normalized note is given by:

$$f_p = \text{round}(1200 * log_2(f/T) * (k/100)) \tag{3.2}$$

where k is number of desired levels in a half step. For instance, $k = 5$ when number of desired levels between two consecutive notes are 5.

Random Sampling

The pitch tracked sequences of audio excerpts from CMD have length of at least $5 * 10^5$ steps. It is impractical to train a Recurrent neural network on sequences with such length as it increases training time considerably. Also the network would struggle to retain information over such large sequences. Hence, we train the network on *subsequences*, which are smaller sequences randomly sampled from the pitch tracked audio excerpt. It is random in that the start index is randomly selected. We sample N_r number of times from every audio excerpt. Length of the subsequences are chosen carefully as it impacts the models training considerably. We observe that smaller sequences tend to confuse the model. An ideal value for the length of the subsequence is 4000-5000 steps.

Choosing an appropriate value for N_r

Eq (3), determined empirically, gives the appropriate value for N_r, based on the length of the subsequence L_r and maximum of the set of lengths of all pitch tracked audio excerpts L_{max}.

$$N_r = 2.2 * L_{max}/L_r \tag{3.3}$$

3.1.2 Model Architecture

At the core of our model (refer Figure 1) is the LSTM [32, 33] based recurrent neural network, which is a popular choice for sequence classification and sequence to sequence learning tasks. We experiment with various configurations for the LSTM block and find that a single LSTM layer with hidden size of 768 works the best. The word embedding layer, which appears prior to the LSTM layer, converts each note that appears in the subsequence into a 128 dimensional vector. The LSTM block is followed by an attention layer. Our implementation closely follows soft alignment as described in [34]. The attention layer is followed by two densely connected layers, first of which has 384 hidden units and the second one has hidden units equal to the number of classes represented in the training dataset. This is followed by a softmax layer, which converts the output of the final dense layer into a probability distribution over the given number of classes i.e, a vector of length equal to number of classes and sums to one.

3.1.3 Model Training and Optimization

We train the model for over 50 epochs, with a batch size of 40 (i.e, 40 subsequences are fed to model at each training step) and an initial learning rate of 0.0001. We employ dropout [35] in the dense layer and batch normalization [36] after the LSTM layer to reduce to reduce over fitting.

Loss Function

Since we are dealing with a classification problem with number of classes greater than 2, the categorical cross entropy is the best suited loss function. The categorical cross entropy loss (CCE) is computed as follows:

$$\text{CCE} = -\sum_{i=1}^{N} y_{i,j} * log(p_i) \tag{3.4}$$

Where N is the number of classes, $y_{i,j}$ is an indicator function which is 1 only when $i = j$, j is the training label for the given subsequence, p_i is the i^{th} entry of the probability vector (which is the output of the model).

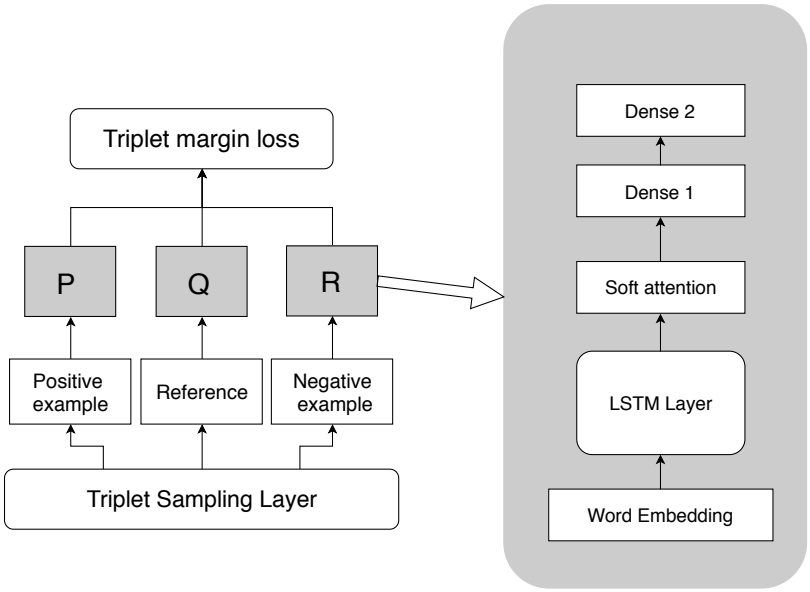

Figure 3.2: Schematic diagram for the sequence ranking algorithm. P, Q and R are the copies of the same model and hence have the same architecture.

Optimization

Random sampling dramatically increases the size of the data set (the size would be $480 * N_r$, where 480 is the number of recordings in CMD). This makes it hard to fit all these samples on a single computer. Hence we use the Distributed asynchronous stochastic gradient descent training algorithm [37]. We use the Adam [38] optimizer to update the gradients during training.

3.2 SEQUENCE RANKING AND RETRIEVAL

A fully trained sequence ranking model will be able to create feature embeddings such that the L2 distance between the feature embeddings of a pair of sequences is directly proportional to how similar the sequences are. In this work we fine tune a pre-trained Raga recognition model (SRGM1, from the previous Section) using the triplet margin loss and evaluate its ability in retrieving subsequences similar to query subsequence. The demonstrated ability to retrieve similar sounding musical pieces is a unique and highly useful feature of our method.

3.2.1 Preprocessing

The preprocessing steps used in the training of SRGM1 apply to sequence ranking. We start we audio source separation followed by pitch tracking, Tonic normalization and finally random sampling.

3.2.2 Model Architecture

The network architecture of the sequence ranking model is shown in **??**. P, Q and R are modified copies of pre-trained SRGM1 network (Note: the diagram shows P,Q and R as different blocks. However, in practice, the triplets are passed through same network one after the other). The modifications being, 1) absence of softmax activation at the end of the network 2) Dense 2 is altered to have 600 hidden units in place of 40 (or 10 depending on the dataset it was trained on)

3.2.3 Triplet Sampling

The triplet sampling layer supplies the model with triplets such that one pair is similar to each other and the other pair is dissimilar. We consider two sequences to be similar if they are from the same Raga and are dissimilar if they are from different Ragas. The procedure for sampling triplets is as follows:

- From the collection of all subsequences, sample a subsequence P_{ref} randomly (uniform random).
- Let P_{ref} belong to Raga R_i
- Select another phrase P_+ from the same Raga R_i randomly (uniform random) such that $P_+ ! = P_{ref}$
- Randomly (uniform random) choose another Raga R_j such that $R_j ! = R_i$ from the set of all Ragas R
- Sample P_- from Raga R_j randomly (uniform random).

3.2.4 Model Training and Optimization

Similar to the training of SRGM1, we employ dropout on the dense layers of the model. Since we are fine tuning SRGM1, we do not have to train the word embeddings and the

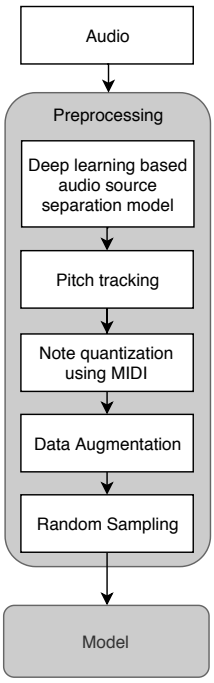

Figure 3.3: Preprocessing steps for the Tonic Raga recognition model

LSTM layers. Triplet sampling can be offline (before training) or online (during training). Although it reduces training time, offline sampling is extremely inefficient as it requires a lot of memory (RAM). Hence we sample triplets as and when required. The sequence ranking model is optimized based on the triplet margin loss, given by:

$$\mathcal{L} = min(D(\mathcal{P}_+, \mathcal{P}_{ref}) - D(\mathcal{P}_-, \mathcal{P}_{ref}) + M, 0) \tag{3.5}$$

where P is the feature embedding obtained for any input P using the model. D(.) is a distance function (most common choice is the euclidean distance) and M is the margin (common choice 1). Similar to SRGM1, we use distributed asynchronous stochastic gradient descent for training the model with Adam optimizer.

3.3 LEARNING TO CLASSIFY INDEPENDENT OF THE TONIC

3.3.1 Preprocessing

We follow a similar preprocessing pipeline as detailed in 3.1, with the differences being, the introduction of the data augmentation technique and a new approach to note quantization using MIDI which stands for Musical Instrument Digital Interface. This enables the model

to learn the structure inherent in the melody while generalizing it to multiple Tonics at the same time. (Figure 3.3 outlines various preprocessing steps involved).

Note Quantization using MIDI

Since Indian Classical Music predominantly characterized by acoustic instruments/ vocals, the audio hence is a continuous waveform. To be able to effectively analyze a sequence of frequencies, the audio has to be discretized. For this we could convert a given frequency into the corresponding Musical Instrument Digital Interface (MIDI) Note by using the formula,

$$F_{MIDI} = 69 + 12 \times log_2(f/440). \tag{3.6}$$

The issue with this approach is that frequencies in the range of 21 Hz to 4186 Hz is represented by 88 discreet levels (i.e MIDI note 21 to 108) which leads to a severe loss of information. Hence we define α additional levels (which are technically called cents) between two MIDI notes, thus resulting in a total of $88 \times \alpha$ possible levels (we choose to be 5).

Data Augmentation

Although the process mentioned above helps our classifier in efficiently learning the nuances of music, it necessitates that we feed the classifier with data in every possible Tonic, (mostly not available), in order to make the classifier independent of the Tonic. To address this problem, we propose a novel data augmentation technique, detailed as an algorithm (Algorithm 1). In short, the idea is to linearly transpose MIDI audio excerpts to represent them across multiple Tonics.

3.3.2 Model Architecture, Training and Optimization

Since the proposed solution to make the Raga recognition model independent of Tonic is a data augmentation technique, it can be used with almost any suitable deep learning method. In our work we use this method in conjunction with the sequence classification model described in Section 3.1. The model is trained using the asynchronous stochastic gradient descent algorithm, optimized on the categorical cross entropy loss (procedure adopted from Section 3.1)

Algorithm 1 Data Augmentation Technique to make the Classifier Tonic Independent

Require: $\mathcal{A} = A_1, A_2, .., A_T$, a collection of all audio excerpts
Require: S is a an empty set which will be populated with audio samples as and when they are transformed
Require: X is a an empty set which will be populated with the data augmented audio samples
Require: δ is a hyperparameter (we set it to 1).
  for $A_i \in \mathcal{A}$ **do**
    $T = \{\}$
    $A_i \longleftarrow FFE(A_i)$
    for all p_i in A_i **do**
      $p_{i_{MIDI}} = F_{MIDI}(p_i)$
      $T \longleftarrow T \cup p_{i_{MIDI}}$
    end for
    $S \longleftarrow S \cup T$
    $minv \longleftarrow min(T)$
    $maxv \longleftarrow max(T)$
  end for
  for all $S_i \in S$ **do**
    $S_{i_{copy}} \longleftarrow S_i$
    while $minv_i > 10$ **do**
      $S_{temp} \longleftarrow S_{i_{copy}} - \delta$
      $X \longleftarrow X \cup S_{temp}$
      $S_{i_{copy}} \longleftarrow S_{temp}$
    end while
    $S_{i_{copy}} \longleftarrow S_i$
    while $maxv_i < 108$ **do**
      $S_{temp} \longleftarrow S_{i_{copy}} + \delta$
      $X \longleftarrow X \cup S_{temp}$
      $S_{i_{copy}} \longleftarrow S_{temp}$
    end while
  end for

CHAPTER 4: DATASET, INFERENCE AND EVALUATION METHODOLOGY

4.1 DATASET

In this work, we use the Carnatic Music Dataset [25] made available by the Comp Music group. The Comp Music data set consists of high quality recording of live ICM concerts. This dataset comprises of 124 hours of 480 commercially available recordings that are stored in the mp3 format. The dataset contains 12 full length recordings per Raga and features 40 different Ragas (hence 480 recordings). These recordings feature numerous artists, wide variety of compositions and a diverse set of Ragas. The Ragas are diverse in terms of their melodic attributes (refer to Section 1.2). Therefore, we believe that training and evaluating the model on this dataset would allow us to assess how well the model would perform in near real world scenarios.

4.2 INFERENCE

4.2.1 SRGM1

Although the model is trained on subsequences, during inference, we are interested in determining the Raga of the audio as a whole. Thus, follow the procedure as shown in Figure 4.1. First, we split the audio excerpt into numerous subsequences and obtain the predictions for the each of these. We then perform a voting to determine the majority class. Note, that if the majority class has less than 60% of the votes we label the classification as incorrect.

4.2.2 SRGM2

For SRGM2, we are interested mostly in the sequences the model retrieves rather than inference on the audio as a whole. However, if one desires to make inference on the audio as a whole, we can adopt the procedure described in Section 3.3.1.

4.3 EVALUATION STRATEGY

We compare our results with three prior works, [25, 2, 21] of which [2] is the current state of the art for the Raga recognition task on the CMD dataset and [25] is the current state of

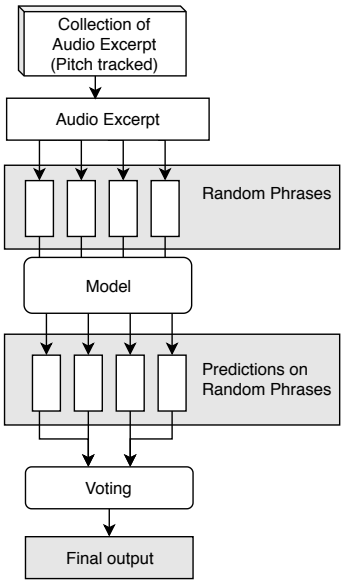

Figure 4.1: Figure gives an overview of the inference process for SRGM1 as described in Section 5.3.1

the art for Raga recognition on the 10 Raga subset of CMD dataset. Gulati et al. [2] provide a summary of the performance of all of these the three methods on the CMD dataset. To ensure a fair comparison, we follow the evaluation strategy as provided in [2] (described in 3.3.1). The evaluation strategy is discussed in detail below.

4.3.1 Evaluation Strategy for SRGM1

CMD consists of 12 recordings per Raga. Since we train the model on subsequences, we have a total of $12*N_r$ subsequences in every class. Therefore, this is a balanced classification problem and accuracy is a suitable metric for evaluation. Authors in [2] use leave-one-out cross validation strategy for evaluation where one of the 12 recordings for every Raga is used as test data and the remaining 11 are used as the training data. We adopt the same approach to evaluate our model as it enables us to make fair comparision of our results with previous works. We also present a confusion matrix for the observed results to further investigate the performance of SRGM1.

Gulati et al. [25] use stratified 12 fold cross validation. To be able to compare our results to theirs on the 10 Raga subset of CMD, we adopt the same strategy. As an alternate evaluation strategy we hold out 5 of the 12 recordings from every class as the test set and train the model on the rest.

4.3.2 Evaluation Strategy for SRGM2

Precision at top-k

A suitable evaluation metric for SRGM2, a sequence ranking model, is the "precision at top-k" measure which is popular score used to evaluate information retrieval systems. Precision at top k is defined as the proportion of retrieved items in the top-k set that are relevant to the query. We consider a retrieved subsequence to be relevant when it belongs to the same Raga as the query subsequence. The expression for precision at top-k hence becomes :

$$P_k = \frac{\sum_{i=1}^{k} I_{y_i,c}}{k} \tag{4.1}$$

$$P_{avg} = \frac{\sum_{i=1}^{n} P_{k_i}}{n} \tag{4.2}$$

Where P_k is precision at top k evaluated on one query sample, I_{y_i} is an indicator function which is 1 only when y_i which is the Raga of the i^{th} retrieved sample is same as c which is the Raga of the query sample. P_{avg} is the top-k precision averaged over the complete dataset.

Mean Average Precision (mAP)

Although a good metric for evaluation, the major drawback of the precision at top-k metric is that it only cares about the top-k items that are retrieved and not the order in which they are retrieved. The order becomes essential since a user would expect the items at the top of the list (of retrieved items) to be more relevant than the ones at the bottom. Hence, to obtain a wholesome evaluation of our model we use the mean average precision, defined as below:

$$mAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} \tag{4.3}$$

where Q is the total number of queries and AveP is the average precision at top-k for a query, given by:

$$AveP = \frac{1}{GTP} \times \sum_{i=1}^{k} \frac{TP_{seen}}{i} \tag{4.4}$$

where, GTP is ground truth positives and TP_{seen} is the number of true positives seen/observed.

CHAPTER 5: RESULTS AND DISCUSSION

We present a summary of results in table 5.1 where we compare the performance of our model with [2], [25] and [21]. $\mathcal{M}_F$, $\mathcal{M}_{KL}$ and $\mathcal{M}_B$ represent 3 variations of TDMS [2] which uses euclidean distance, KL divergence and Bhattacharya distance respectively. $\mathcal{E}_{VSM}$ represents vector space model [25] for Raga recognition while $\mathcal{E}_{PCD}$ represents pitch class distribution based method presented in [21].

We observe that our method performs better than the current state-of-the-art on both 10 Raga subset of CMD and the CMD as a whole. We obtain an improvement of 6 % on the 10 Raga subset of CMD and an improvement of 2% on CMD using the SRGM1-Ensemble model. The Ensemble model was created using 4 sets of weights for the same model architecture. For each test sample we obtain output from each of the 4 models and then combine them by summing the log of the outputs of the model to obtain the final prediction vector for that sample.

The confusion matrix for SRGM1 trained on CMD (40 Ragas) is as shown in Figure 5.1. Note that the confusion matrix is presented for the results obtained on a subsequence level and not on the audio excerpt level i.e, these are results before performing inference (refer to Section 4.3.1). There are no patterns apparent from the confusion matrix. However, on closely observing the samples on which the model performs poorly, we see that these samples have very little information, which can be attributed to events like to long pauses during the performance, sustaining the a particular note for a prolonged interval etc.

Method	**CMD-10 Ragas**	**CMD-40 Ragas**
SRGM1	95.6%	84.6%
SRGM1 Ensemble	**97.1**%	**88.1**%
$\mathcal{M}_F$	-	81.5 %
$\mathcal{M}_{KL}$	-	86.7 %
$\mathcal{M}_B$	-	86.7 %
$\mathcal{E}_{VSM}$	91.7 %	68.1 %
$\mathcal{E}_{PCD}$	82.2 %	73.1 %

Table 5.1: Table summarizes performances of various models on the CMD dataset and its 10 Raga subset.

Figure 5.1: Figure captions should be placed below the figure.

Subsequence length	**Num epochs to converge**	**Time per epoch in sec (Wall Time)**	**Hold out test set accuracy**
500	12	32	88.86
1500	6	58.4	95.5
3000	5	120.1	95.63
6000	3	241.5	97.34

Table 5.2: Summary of our findings for the study on variation of model training and performance based on subsequence length.

Metric	Precision at Top-30	Precision at Top-10	Top-30 Mean Average Precision	Top-10 Mean Average Precision
Score	81.83%	81.68%	85.23%	85.97%

Table 5.3: Summary of performance of sequence ranking on the top-10 and top-30 precision metrics.

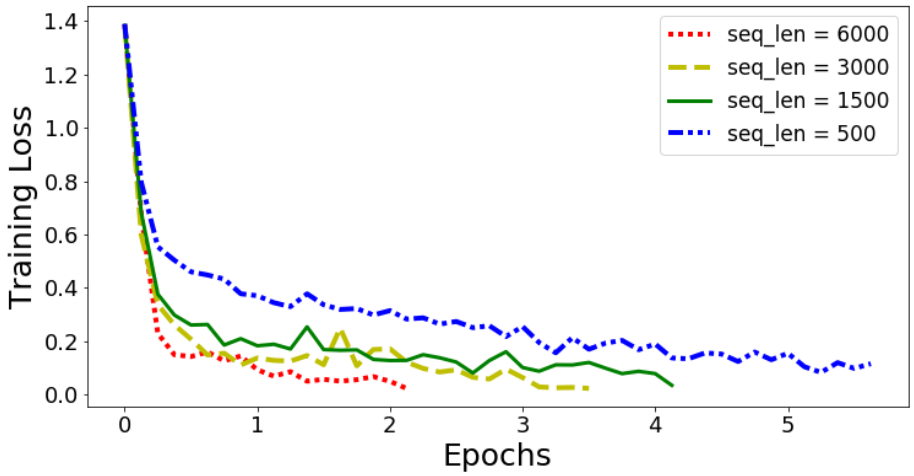

Figure 5.2: Figure captions should be placed below the figure.

To further investigate the effect of subsequence length on the training of a model, we device an experiment by using a 4 Raga subset of CMD. We use 36 recordings as training set and 12 recordings as the test set. We train the network until the categorical cross entropy loss reduces to 0.02. Figure 5.2 shows a plot of the training loss vs number of epochs. It is clearly visible that a model that is trained on subsequences of length 6000 converges in lesser number of epochs and has a smooth descent while the using subsequences of length 500 makes the training very noisy and the network takes as long as 12 epochs to attain the same value of loss. A summary of our findings has been tabulated in table 2.

We evaluate the sequence ranking model on the metrics described in Section 5.4.2, namely top-10 precision and top-30 precision. Our findings have been summarized in table 3. SRGM2 obtains a top-30 precision of 81.83 % and top-10 precision of 81.68 %. We also conduct a qualitative analysis of the retrieved samples by inspecting various aspects like the range of svaras (notes) observed in the query to range of svaras observed in the retrieved sample, checking for similar note progressions etc. We observe that the sequences retrieved by the model are similar to the ones in the query.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

In this work, we proposed a novel method to address the problem of Raga recognition. Through various experiments and validation we are able to demonstrate the capability of our approach in tackling the problem of Raga recognition. Our model achieves 88% accuracy on CMD and 97% accuracy on it's 10 Raga subset, hence improving the state-of-the-art performance. We then introduce sequence ranking as a new sub-task of Raga recognition which obtains a top-10 precision of 81% and a top-10 mean average precision of 85%. Having conducted numerous qualitative analysis on the performance of the sequence classification model, we observe that the model is able to retrieve meaningful and musically proximate sequences to the query sequence. We also introduce a new algorithm that enables deep learning based Raga recognition models to be Tonic independent.

6.2 FUTURE WORK

We believe that deep learning based approaches have tremendous scope in MIR pertinent to ICM. As part of future work, we would like to further explore the idea of sequence ranking as we feel it can be used for tasks like joint recognition of tonic and Raga. We would also like to pursue better triplet sampling strategies and evaluation metrics for the same. An exciting future research direction would be to explore the possibility of using deep learning for generative modelling tasks in ICM. It would definitely be interesting to see if deep learning models can replicate the intricate improvisation aspects of ICM.

CHAPTER 7: REFERENCES

[1] T. Ganti, *Bollywood: A guidebook to popular Hindi cinema.* Routledge, 2013.

[2] S. Gulati, J. Serrà Julià, K. K. Ganguli, S. Sentürk, and X. Serra, "Time-delayed melody surfaces for rāga recognition," in *Devaney J, Mandel MI, Turnbull D, Tzanetakis G, editors. ISMIR 2016. Proceedings of the 17th International Society for Music Information Retrieval Conference; 2016 Aug 7-11; New York City (NY).[Canada]: ISMIR; 2016. p. 751-7.* International Society for Music Information Retrieval (ISMIR), 2016.

[3] P. Chordia and A. Rae, "Raag recognition using pitch-class and pitch-class dyad distributions." in *ISMIR*. Citeseer, 2007, pp. 431–436.

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

[5] A. Krishnaswamy, "Multi-dimensional musical atoms in south-indian classical music," in *Proc. of International Conference on Music Perception and Cognition*, 2004.

[6] T. Krishna and V. Ishwar, "Carnatic music: Svara, gamaka, motif and raga identity," in *Serra X, Rao P, Murthy H, Bozkurt B, editors. Proceedings of the 2nd CompMusic Workshop; 2012 Jul 12-13; Istanbul, Turkey. Barcelona: Universitat Pompeu Fabra; 2012.* Universitat Pompeu Fabra, 2012.

[7] G. K. Koduri, S. Gulati, P. Rao, and X. Serra, "Rāga recognition based on pitch distribution methods," *Journal of New Music Research*, vol. 41, no. 4, pp. 337–350, 2012.

[8] L.-L. Balkwill and W. F. Thompson, "A cross-cultural investigation of the perception of emotion in music: Psychophysical and cultural cues," *Music perception: an interdisciplinary journal*, vol. 17, no. 1, pp. 43–64, 1999.

[9] D. R. Shringy and D. P. L. Sharma, "Sangita ratnakara of sarngadeva, text and translation."

[10] X. Wang and Y. Wang, "Improving content-based and hybrid music recommendation using deep learning," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 627–636.

[11] K. Drossos, S. I. Mimilakis, D. Serdyuk, G. Schuller, T. Virtanen, and Y. Bengio, "Mad twinnet: Masker-denoiser architecture with twin networks for monaural sound source separation," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[12] L. Rabiner, "On the use of autocorrelation analysis for pitch detection," *IEEE transactions on acoustics, speech, and signal processing*, vol. 25, no. 1, pp. 24–33, 1977.

[13] P. Boersma, "Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound," in *Proceedings of the institute of phonetic sciences*, vol. 17, no. 1193. Amsterdam, 1993, pp. 97–110.

[14] Y. Medan, E. Yair, and D. Chazan, "Super resolution pitch determination of speech signals," *IEEE transactions on signal processing*, vol. 39, no. 1, pp. 40–48, 1991.

[15] A. De Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[16] V. Rao and P. Rao, "Vocal melody extraction in the presence of pitched accompaniment in polyphonic music," *IEEE transactions on audio, speech, and language processing*, vol. 18, no. 8, pp. 2145–2154, 2010.

[17] A. P. Klapuri, "Multiple fundamental frequency estimation based on harmonicity and spectral smoothness," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 6, pp. 804–816, 2003.

[18] J. Salamon and E. Gómez, "Melody extraction from polyphonic music signals using pitch contour characteristics," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 6, pp. 1759–1770, 2012.

[19] L. Benaroya, F. Bimbot, and R. Gribonval, "Audio source separation with a single sensor," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 191–199, 2006.

[20] S. Gulati, "A tonic identification approach for indian art music," 2012.

[21] P. Chordia and S. Şentürk, "Joint recognition of raag and tonic in north indian music," *Computer Music Journal*, vol. 37, no. 3, pp. 82–98, 2013.

[22] G. K. Koduri, V. Ishwar, J. Serrà, and X. Serra, "Intonation analysis of rāgas in carnatic music," *Journal of New Music Research*, vol. 43, no. 1, pp. 72–93, 2014.

[23] P. Dighe, H. Karnick, and B. Raj, "Swara histogram based structural analysis and identification of indian classical ragas." in *ISMIR*, 2013, pp. 35–40.

[24] P. Dighe, P. Agrawal, H. Karnick, S. Thota, and B. Raj, "Scale independent raga identification using chromagram patterns and swara based features," in *2013 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*. IEEE, 2013, pp. 1–4.

[25] S. Gulati, J. Serra, V. Ishwar, S. Sentürk, and X. Serra, "Phrase-based rāga recognition using vector space modeling," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 66–70.

[26] A. S. Krishna, P. Rajkumar, K. Saishankar, and M. John, "Identification of carnatic raagas using hidden markov models," in *2011 IEEE 9th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2011, pp. 107–110.

[27] V. Kumar, H. Pandya, and C. Jawahar, "Identifying ragas in indian music," in *2014 22nd International Conference on Pattern Recognition*. IEEE, 2014, pp. 767–772.

[28] S. Shetty and K. Achary, "Raga mining of indian music by extracting arohana-avarohana pattern," *International Journal of Recent Trends in Engineering*, vol. 1, no. 1, p. 362, 2009.

[29] S. Dutta and H. A. Murthy, "Raga verification in carnatic music using longest common segment set." in *ISMIR*, vol. 1. Malaga, Spain, 2015, pp. 605–611.

[30] Y. Jadoul, B. Thompson, and B. De Boer, "Introducing parselmouth: A python interface to praat," *Journal of Phonetics*, vol. 71, pp. 1–15, 2018.

[31] P. Boersma et al., "Praat, a system for doing phonetics by computer," *Glot international*, vol. 5, 2002.

[32] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[33] C. Olah, "Understanding lstm networks," 2015. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[34] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[35] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[36] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[37] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le et al., "Large scale distributed deep networks," in *Advances in neural information processing systems*, 2012, pp. 1223–1231.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.