MACHINE LEARNING FOR RAGA IDENTIFICATION IN INDIAN CLASSICAL MUSIC

An Undergraduate Research Scholars Thesis

by

PRANATI CHINTHAPENTA

Submitted to the Undergraduate Research Scholars program at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by Research Advisor: Dr. Byung-Jun Yoon
Dr. Krishna Narayanan

May 2019

Major: Electrical Engineering

TABLE OF CONTENTS

ABSTRACT

Machine Learning for Raga Classification in Indian Classical Music

Pranati Chinthapenta
Department of Electrical and Computer Engineering
Texas A&M University

Research Advisor: Dr. Byung-Jun Yoon
Department of Electrical and Computer Engineering
Texas A&M University

Indian Classical music consists of "Ragas". Ragas are combinations of notes in a particular order. There are 72 combinations of these notes that form the 72 parent Ragas. Figure 1 shows a graph of the Ragas. The names of the Ragas are written around the circle while the defining notes of the Ragas are written inside the concentric circles. While Western music also has notes that go with the songs, it differs from Indian music in the rules that Ragas set to a specific song. For example, if someone was to perform a song in Raga #1, they must stick to the notes that are present in that Raga. Otherwise, it is no longer Raga #1. While imposing rules, Ragas also provide a lot of complexity and creativity to the artist. The artist needs to create impromptu variations while staying within the Raga's boundaries. This difficulty gets intensified when the notes are so close to each other that they can be easily confused. Recognizing a Raga is a whole other ordeal. They can be identified only by a well-trained artist. In addition to this, every single artist has their own style which can be shown while singing or playing instruments. People add their own nuances to make the song their own, while all the time obeying the demands of the Raga. People do not sing the notes of the song. Therefore, we cannot rely on speech to recognize the Raga. People also have different baseline frequencies - child v. adult, male v. female, healthy person v. one with a cold,

etc. All the complexity makes the problem of Raga identification challenging and something worth conquering. Ragas can be interpreted as specific frequency-time contours. Each peak in the frequency-time contour corresponds to a note. Each of them can be associated to a person's mood. This can be especially useful in music recommendation systems. It can also make a huge mark in copyright violation detection systems. Machine Learning and Signal Processing tools will prove to be very effective. Signal Processing will be used to obtain the feature vector for the Machine Learning algorithm while ML will be used to classify the songs into their respective Ragas.

The proposed methodology is to use Signal Processing techniques to obtain the feature vector for the ML algorithm and then use this to classify the Ragas. First, Discrete Fourier Transform (DFT) graphs are obtained. These graphs need to be normalized. The feature vectors are obtained, and these consist of smaller vectors. The vectors are the frequencies of the audio sample and its One-Hot encoded label. The Machine Learning algorithm then learns to recognize how close the vectors are to each other and classifies them into groups while outputting the name of the Raga. This method can be extended to any new Ragas that are developed.

ACKNOWLEDGEMENTS

I would like to thank my faculty advisors, Dr. Yoon and Dr. Narayanan, for their guidance and support throughout the course of this research. I would also like to thank the Undergraduate Research Scholar Program for giving me the opportunity and resources to write this thesis.

Special thanks to my friend Abhishek Joshi for helping me with implementing my first ML algorithm. Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience. I also want to extend my gratitude to my TA - Pranav Dhulipala, my professors - Stavros Kalafatis, Kevin Nowka and John Lusher and the entire Electrical Engineering department, including TAMU High Performance Research Computing for helping me through the Senior Design process.

NOMENCLATURE

HMM	Hidden Markov Models
ML	Machine Learning
DSP	Digital Signal Processing
AI	Artificial Intelligence
MATLAB	Matrix Laboratory
NN	Neural Networks
RNN	Recurrent Neural Networks
LSTM	Long Short-Term Memory

CHAPTER I

INTRODUCTION

Algo-Rhythm is a music recommendation system that tracks the types of songs people listen to and recommend similar or different songs depending on their mood. This is similar to what companies like Spotify are trying to do by combining Machine Learning and Signal Processing. It is based in Indian Classical Music since the concept of "Raga" demonstrates how similar mood is to a tune. The algorithm must be able to let users input voice or instrument recordings and classify them into "Ragas". Signal Processing techniques will be used to obtain the feature vector for the Machine Learning algorithm and the actual algorithm will be used to classify the recordings into Ragas. The algorithm proposed in this document is designed for music enthusiasts and engineers alike. It combines electrical engineering and programming with music in a novel idea. The algorithm will be able to help track mood changes of people throughout the day and can be particularly helpful for people with mental challenges. Since music is considered as a form of therapy in a lot of cases, this algorithm can be an extremely useful tool in ensuring that people are on the right path of progress.

Background

Indian Classical music consists of "Ragas". Ragas are combinations of notes in a particular order. There are 72 combinations of these notes that form the 72 parent Ragas. Figure 1 shows a graph of the Ragas. The names of the Ragas are written around the circle while the defining notes of the Ragas are written inside the concentric circles. While Western music also has notes that go with the songs, it differs from Indian music in the rules that Ragas set to a specific song. For example, if someone was to perform a song in Raga #1, they must to stick to the notes that are

present in that Raga. Otherwise, it is no longer Raga #1. While imposing rules, Ragas also provide a lot of complexity and creativity to the artist. The artist needs to now show how many impromptu variations they can do while staying within the Raga's boundaries. This difficulty gets intensified when the notes are so close to each other that they can be easily confused.

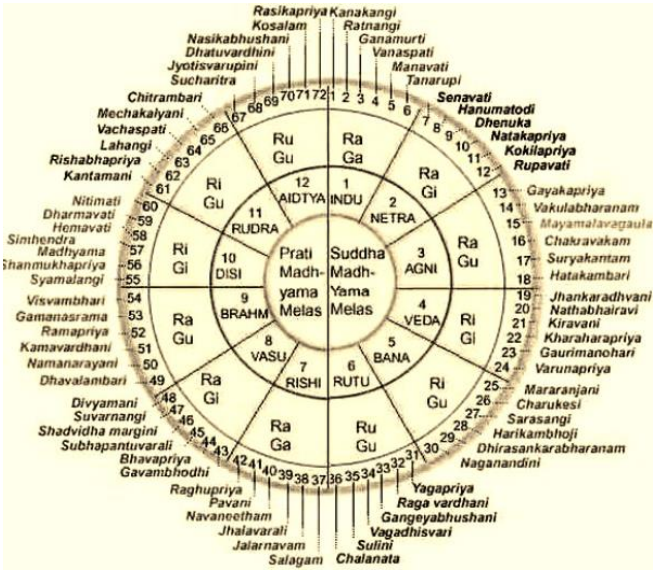

Figure 1: 72 Parent Ragas

Recognizing a Raga can only be done by a well-trained artist. In addition to this, every single artist has their own style which can be shown while singing or playing instruments. People add their own nuances to make the song their own, while all the time keeping obeying the demands of the Raga. People do not sing the notes of the song. Therefore, we cannot rely on speech to recognize the Raga. People also have different baseline frequencies - child v. adult, male v. female, healthy person v. one with a cold, etc. All the complexity makes the problem of Raga identification challenging and something worth conquering. Ragas can be interpreted as specific frequency-time contours. Each peak in the frequency-time contour corresponds to a note. Each of them can be associated to a person's mood. This can be especially useful in music recommendation systems. It

can also make a huge mark in copyright violation detection systems. Machine Learning and Signal Processing tools will prove to be very effective. Signal Processing will be used to obtain the feature vector for the Machine Learning algorithm while Machine Learning will be used to classify the songs into their respective Ragas.

Overview

In the initial version of the algorithm, the user can input an audio file recorded by a string instrument in a noise free environment. The output to the algorithm will be the list of ragas that are most similar to the one that is played. In addition to the raga classification.

A more advanced version of the algorithm is when the user can use it in any setting and not be restricted to string instruments. The user should be able to sing a song and the algorithm needs to identify the base pitch since this various from person to person. In addition to recognizing the raga, the algorithm will also be able to recognize the mood of a person.

CHAPTER II

METHODS

Algo-Rhythm combines Digital Signal Processing and Machine Learning to recognize Ragas. It uses DSP for pre-processing data and Machine Learning to classify this data into Ragas. Figure 2 shows an overview of the project and the process that DSP and ML performs to obtain the output.

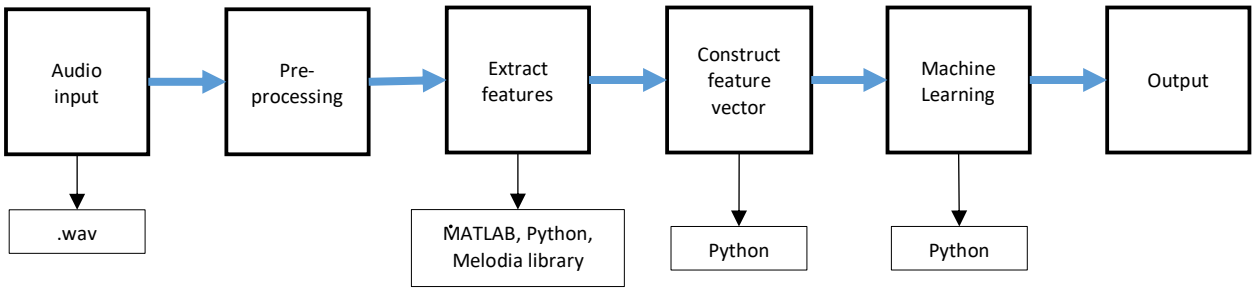

Figure 2: Flow diagram of the algorithm

Step 1: Pre-processing data

The data was obtained using datasets online. The first step for in the algorithm is to process the data. This includes removing noise, considering different baseline frequencies and computing the Discrete Fourier Transform of the data so that it is easily accessible. In addition, the data has to be structured in a way that can be easily passed through the Machine Learning algorithm.

To compute the DFT plots, Melodia [1] was used. In addition to this, a tutorial [2] by Justin Salamon was also utilized. Following obtaining the DFT plots, there were seen to be negative frequencies in the graphs. This was said to be what the Melodia library does when there is “no melody” in the recording. However, there is no definition of what melody is defined as. After this,

the next step was to smoothen out the very highly fluctuating notes, also called vibrato. This was done using a Savitzky – Golay filer. Figure 3 shows how the filter was applied to smoothen out high frequency components. It also shows how the frequencies were normalized so different baseline frequencies were accounted for.

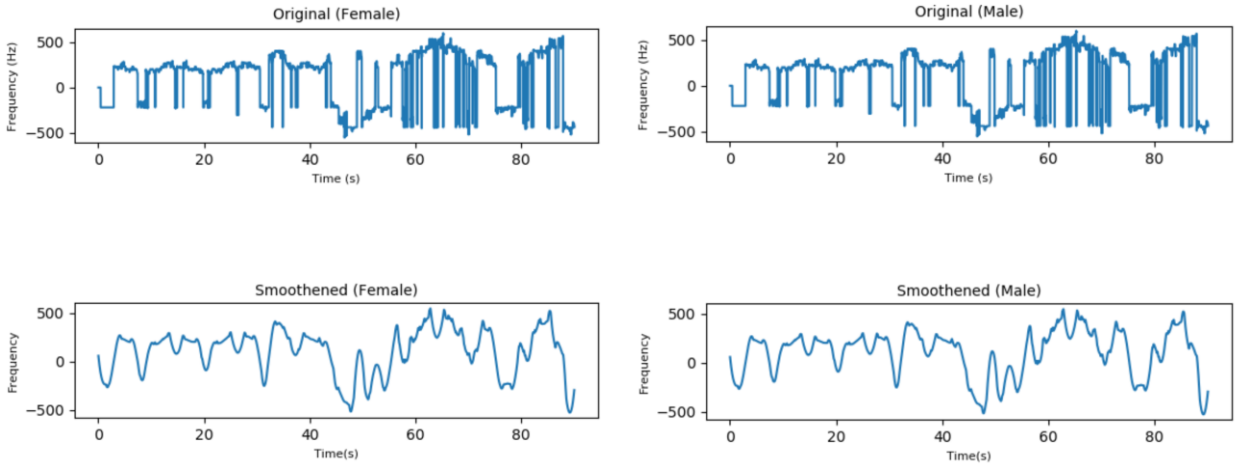

Figure 3: DFT graphs of the same audio clip by male and female artists

Step 2: Feature Vector Extraction

The feature vector is the data that becomes the input to the machine learning algorithm. This must be in a simplified way so that it is easy for the algorithm to process. Frequencies of the notes are form the most important part of the feature vector. The structure of the feature vector looks like the following:

$$< f_1 >, < f_2 >, \dots , < f_n >$$

Step 3: Machine Learning

Machine Learning is an extremely robust technique. As the name suggests, in this method, we are trying to train the machine to learn certain qualities of our data. Imagine this to be like a

child learning to recognize a cat versus a dog. Multiple pictures of both animals are shown along with their labels and soon the child begins to learn to recognize and differentiate the two animals. This, in particular is called a classification problem. It is exactly what happens in this project as well. Large amounts of data are fed into the algorithm so that it starts recognizing different patterns and classifies the Ragas into the labels that are given. This project is an example of a classification problem since we are trying to classify various data points into their respective categories. Specifically, Recurrent Neural Networks are used. Keras and TensorFlow are Python libraries that help with creating machine learning algorithms.

One-Hot Encoding for Labels

Labels are the titles we give to various groups of data so that the algorithm can now classify it. The labels in this case are the names of the Ragas. However, the machine can not understand these labels. So, to make it easier, we perform One-Hot encoding on them. One-Hot encoding is a way of converting categorical data into a form that Machine Learning algorithms can understand. ML algorithms require label data to be numeric. This is a constraint for effectively implementing ML algorithms rather than hard implementations themselves. One-Hot encoding removes the integer encoded variable and replaces it with a binary variable that is unique for each label. For example, if we have three colors, blue, red and green, there are three categories and three values. These will be encoded as shown in Table 1.

Neural Networks

Neural Networks are a class of algorithms that are modeled after the human brain and designed to identify patterns. They interpret data by perception, labeling and classifying the input. The inputs of these algorithms are vectors that are known as feature vectors. All real-world data,

whether it is audio, video, sensory, etc. data must be put into these feature vectors. Neural networks have the capacity to classify unlabeled data.

Table 1: One-Hot encoding example

Blue	Red	Green
1	0	0
0	1	0
0	0	1

There are several layers in a neural network that contribute in streamlining the classification process. These layers are made up on nodes. The nodes have weights that enhance or dampen the input data. This helps the network know how important a particular input is and helps find the most efficient way to classify the data without error.

These weights are then summed and passed through the algorithm's activation function as shown in Figure 4. This determines to what extent the signal should move further to affect the outcome. There are several types of neural networks that are best suited for various applications. We will look further at RNNs in the next section.

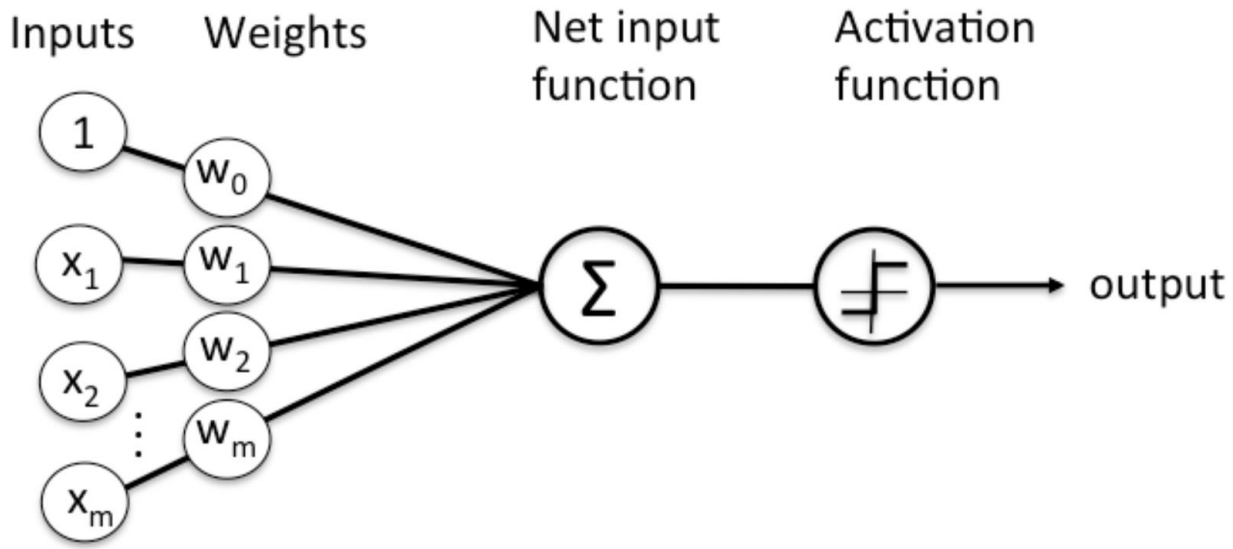

Figure 4: Obtained from https://skymind.ai/wiki/neural-network

Recurrent Neural Networks

A type of neural networks, Recurrent Neural Networks (RNN), are extremely efficient in dealing with sequence prediction data. In Ragas, the order of the notes plays a significant role in defining the raga itself. Therefore, this type of neural network was automatically shortlisted as the winner. Humans do not start their thinking from scratch every time. This is how RNNs behave as well. They have loops. They remember past information to help in the future. An RNN can be thought of as multiple copies of the same neural network. Recurrent Neural Networks were originally difficult to train. However, Long Short-Term Memory (LSTM) overcomes these problems.

LSTMs were first introduced by Hochreiter & Schmidhuber in 1997. Their work was later refined and popularized by many enthusiasts. LSTMs are designed to eliminate the long-term dependency issue. By default, they are made to store information and data for long periods of time. LSTMs also have the chain-like structure that most neural networks have. However, instead of

having a single layer, they have multiple ones interacting in a special way. Figure 5 shows how the LSTM module interacts. The cell state, which is the line at the top is a key ingredient of LSTMs. Only minor changes are made to it and it makes it convenient for information to flow through. Gates are present that make it easy for the LSTM to add or remove information to and from the cell state. These gates are comprised of a sigmoid later and a point-wise multiplication operator. LSTMs have three gates to control the flow of information to and from the cell state.

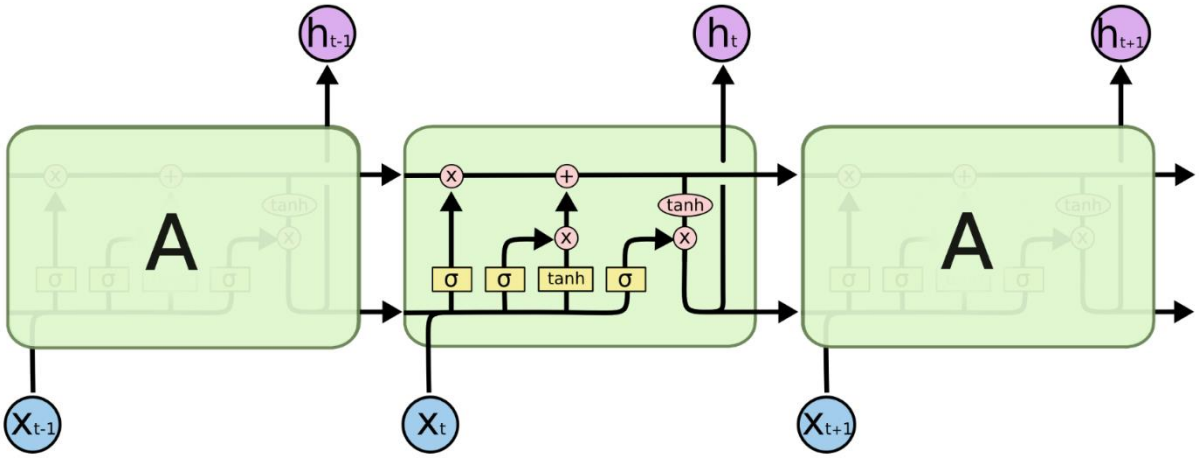

Figure 5: LSTM Structure from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

The structure of the model is a Sequential one. Using Keras which uses a TensorFlow backend, allows the use of a Sequential model which is a linear stack of layers. In addition to the LSTM layers, multiple Dense layers are also added before and after the LSTM layer.

CHAPTER III

RESULTS

The success of a Machine Learning algorithm is heavily dependent on the amount of data given to it. It has to be at least on the order of 10,000 for each label that is present.

Data

There were multiple ways to obtain data for the algorithm. Audio platforms like YouTube, Spotify, Saavn and Pandora provide a plethora of data samples. However, one big problem with these is that there was a lot of noise associated with the recordings. In any DSP project, removing noise and obtaining a clean recording makes the difference between success and failure. As a result of this, it was best to use existing datasets. However, there were more datasets for Hindustani (North Indian) music as compared to Carnatic (South Indian) music. Initially, recordings from both types were obtained. As time went on and problems started appearing in the Neural Network stage as shown in the following sections, the project had to be scaled down to differentiate only between five Ragas. Despite cutting the data down, there were still problems with the accuracy since there was not as much data as required to run a Neural Network algorithm. The Ragas used were as follows:

- Abhogi
- Kalyani
- Mohana
- Sahana
- Saveri

These Ragas all belong to Carnatic music.

Pre-Processing

Before sending the data through the Neural Network, it is imperative that all the data is in the right format. It has to be understandable by the RNN algorithm. The data was obtained in the form of .mp3 files. These were converted to .wav format using online resources since it was easier to edit this format of files. After converting, the files were then split into smaller lengths. Multiple lengths were testing out to see how they compare with each other. The time lengths that were tested were – 20 seconds, 35 seconds and 90 seconds. This was done so that there would be more data to work with for the RNN algorithm.

After obtaining data, the DFT was calculated using Melodia. However, it was seen that this gave rise to negative frequency values as shown in Figure 6. This was seen in pianos and vocal audio samples as seen in Figure 7. The reason for these negative frequencies is unclear due to how the Melodia computes these values. Melodia documentation says that the negative frequencies are areas where there is "no melody" in the audio sample. However, what melody is defined as is unclear since the usual definition that there is no audio at that time is not true.

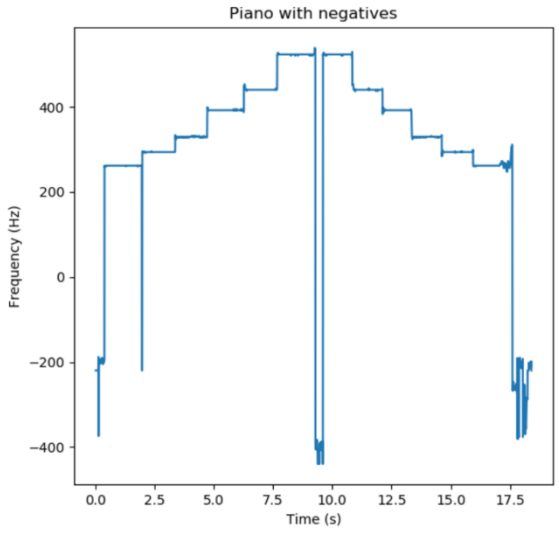

Figure 6: DFT graph of Piano sample of Mohana Raga

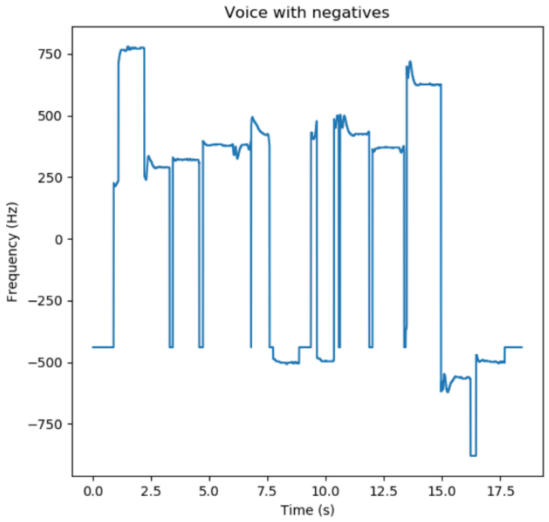

Figure 7: DFT graph of vocal sample of Mohana Raga

Despite obtaining data from datasets that are relatively cleaner and low noise, there was significant discrepancies in the data. In addition to noise, ‘Gamakas’ (known as ‘vibrato’ in Western music) is a very significant component of Indian music. Gamakas can be thought of as two or three notes fluctuating between each other at a high speed. These make it hard to capture individual notes. Therefore, it was important to perform filtering on the data to clean the signals. A Savitzky-Golay filter was used. The results of this filter are shown in Figure 8 and Figure 9. These two images also compare the difference between a male voice and a female voice when they are singing the same audio sample.

The frequencies obtained after filtering are put into an array. This frequency array and the One-Hot encoded labels are appended to another array. Figure 10 shows the format of the data in the feature vector.

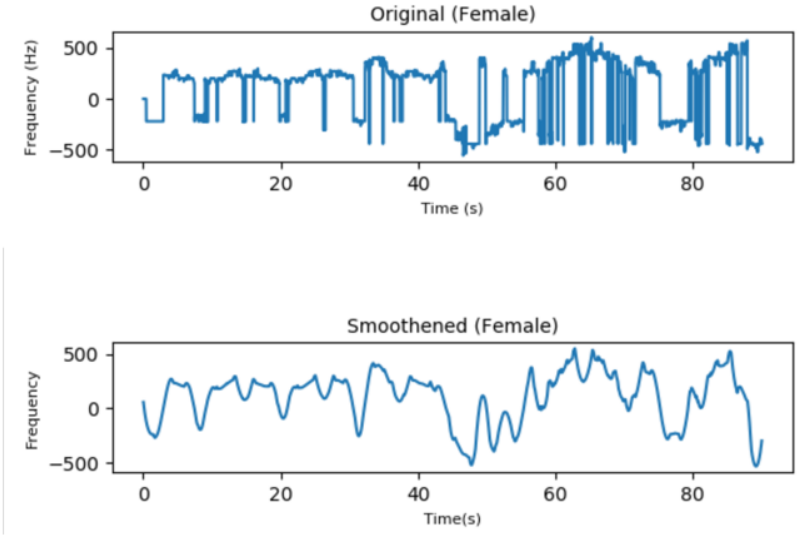

Figure 8: Before and after filtering of female vocal sample

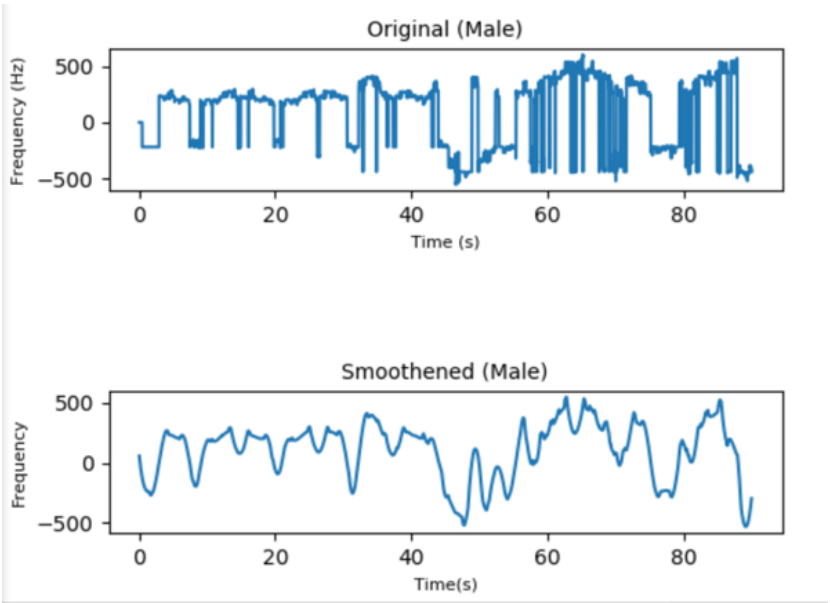

Figure 9: Before and after filtering of male vocal sample

$$[\,[list\ of\ frequencies], [corresponding\ label]\,]$$

Figure 10: Data Array format

The maximum length of each data sample is padded or truncated to 7000. This was done after a lot of trial and error. Initially a value of 400 was selected but this cut off a lot of high frequency components since the sample itself was much longer. The data is then split randomly into test and train buckets. 20% of the available data goes into testing the algorithm.

Neural Network

The presence of Gamakas is also why Machine Learning would be the easiest technique to use for this application. After splitting the dataset into train and test sets, the Neural Network is built.

Test Case 1

First, a vanilla model was built where the inputs were only two Ragas – Mohana and Abhogi. These Ragas are fairly different from each other and therefore, made a good test. Table 2 shows a summary of these results.

Table 2: Summary of Test Case 1

Layers	- 2 Dense Layers - LSTM layer
Train Accuracy	~ 94 %
Test Accuracy	~ 60 %
Length of Array	400
Length of audio sample	90 seconds

Test Case 2

In the second test case, there were a total of 7 Ragas:

- Abhogi
- Kalyani
- Mohana
- Saveri
- Sahana
- Begada
- Sri

The length of the audio samples was reduced to 20 seconds to give rise to more data samples since there were a limited number that were obtained through online datasets. The length of the array was also increased to 7000 since limiting it to 400 was cutting off a lot of high frequency components. Table 3 shows a summary of these results.

Table 3: Summary of Test Case 2

Layers	- 2 Dense Layers - LSTM layer
Train Accuracy	~ 50 %
Test Accuracy	~ 25 %
Length of Array	7000
Length of audio sample	20 seconds

Test Case 3

In the third test case, the length of the samples was increased to 35 seconds. This is because 20 seconds became too small to train the algorithm. More Dense layers were also added before and after the LSTM layer. Table 4 shows a summary of these results.

Table 4: Summary of Test Case 3

Layers	- 8 Dense Layers - LSTM layer
Train Accuracy	~ 87 %
Test Accuracy	~ 30 %
Length of Array	7000
Length of audio sample	35 seconds

CHAPTER V

CONCLUSION

The work in this project is the first of many steps that will help to bring the creative arts and engineering together. It shows that technology has been reaching new heights that even the most complicated forms of art can be combined with engineering. It has to be extended to other Ragas. The principal challenge would be to get enough data samples so that the algorithm can train properly. That was the limitation of this project. Even though the goal was to identify all 72 of the parent Ragas, there were not enough datasets available for this task. Therefore, the project had to be scaled down.

REFERENCES

[1] “MTG - Music Technology Group.” Cerca, www.upf.edu/web/mtg/melodia.

[2] “Melody Extraction.” Justin Salamon, www.justinsalamon.com/melody-extraction.html

[3] 983218471799356. “What Is One Hot Encoding? Why And When Do You Have to Use It?” Hacker Noon, Hacker Noon, 3 Aug. 2017, hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f.

[4] “Why One-Hot Encode Data in Machine Learning?” Machine Learning Mastery, 19 May 2018, machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/.

[5] “A Beginner's Guide to Neural Networks and Deep Learning.” Skymind, skymind.ai/wiki/neural-network.

[6] “When to Use MLP, CNN, and RNN Neural Networks.” Machine Learning Mastery, 25 Apr. 2018, machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/.

[7] Mahendra, Siddarth & Patil, Hemant & Shukla, Narendra. (2010). Pitch Estimation of Notes in Indian Classical Music. 1 - 4. 10.1109/INDCON.2009.5409364.

[8] “Notes, Octaves and Scale.” Kaminimusic.com, 7 Apr. 2017, kaminimusic.com/notes-octaves-scale/.

[9] Ray, Sunil, and Business Analytics. “Essentials of Machine Learning Algorithms (with Python and R Codes).” Analytics Vidhya, 7 Mar. 2019, www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/.

[10] “Indian Art Music Raga Recognition Dataset.” CompMusic, compmusic.upf.edu/node/328.

[11] “Carnatic Varnam Dataset.” CompMusic, compmusic.upf.edu/carnatic-varnam-dataset.

[12] “Datasets.” CompMusic, compmusic.upf.edu/datasets#indian.

[13] Albon, Chris. “LSTM Recurrent Neural Network.” Chris Albon, 20 Dec. 2017, chrisalbon.com/deep_learning/keras/lstm_recurrent_neural_network/.

[14] “Sequence Classification with LSTM Recurrent Neural Networks in Python with Keras.” Machine Learning Mastery, 6 May 2018, machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/.

[15] “Understanding LSTM Networks.” Understanding LSTM Networks -- Colah's Blog, colah.github.io/posts/2015-08-Understanding-LSTMs/.

[16] The Unreasonable Effectiveness of Recurrent Neural Networks, karpathy.github.io/2015/05/21/rnn-effectiveness/.