Index

1. Abstract

Music information retrieval (MIR) is an interdisciplinary field bridging the domains of mathematics, statistics, signal processing, machine learning, computer science, musicology, biology, and more. Some of the most common MIR tasks include finding similar audio items, music classification and archival, playlist recommendation, source separation, etc. Raga is a quintessential aspect of Indian classical music. There are about two hundred ragas being currently performed in Carnatic music concerts and thousands are possible in theory. The Internet has brought a wealth of audio data and raga identification can provide high- level information about music data to become a basis for music search and archival. Ragas are primarily characterized by melodic time-FREQUENCY (T-F) trajectories. With hundreds of ragas being in regular use, performing raga identification as a machine-learning task is certainly not trivial. Ragas allow much scope for improvisation and elaboration within the bounds of certain specified norms, making the task of raga identification difficult for machine learning algorithms. There have been several computational approaches proposed to determine the identity of a raga, however, the techniques work only on a subset(s) of ragas and also perform poorly in terms of scalability.

The initial part of this work addresses the question of finding similar ragas required the verification framework and proposes to use Locality Sensitive Hashing (LSH) as a solution. The later part of the work includes the newly proposed method for raga identification using LSH, validating the approach, and demonstrating improved performance through experiments. Interestingly, LSH is close to how humans perceive ragas. Experiments with 3000 raga queries, were performed on a standard dataset consisting of 100 concerts, each with 8-10 ragas, leading to 927 items and 182 ragas, achieving an average accuracy of 71%. Most importantly LSH does not suffer from the scalability issues of the earlier approach.

2. Introduction

The legacy of Indian Carnatic Classical music has its roots back to 1500BC. Ragas form the backbone of Indian Classical music. Before the technological era, musicians used their hearing to recognize notes and ragas in a composition. Musicians then were able to understand the tonal differences even lesser than 0.2Hz. Carnatic music is composed of seven different keys called Swaras viz.

1. Sa (Shadja)
2. Ri (Rishabha)
3. Ga (Gandhara)
4. Ma (Madhyama)
5. Pa (Panchama)
6. Dha (Dhaivata)
7. Ni (Nishada)

The different combinations of Swaras with adherence to specific rules make up the seventy-two Melakarta ragas. Each raga is composed of defined notes and depicts a specific mood. There are attributes related to each raga that enhances its feel and emotions. Some of these attributes are Arohana and Avarohana (ascending and descending progressions), Gamaka (ornamentation of different notes in a pattern), Vadi (root swara of the raga and the most important note in a raga), Samavadi (the second most important note in a raga), Tala (rhythmic pattern) and Samay (specific time in which a raga shows its dominance and makes the recognition of that raga easy). The seventy-two Melakartha ragas constitute the Indian Carnatic music. These ragas are the parents of all other sub ragas in Carnatic music. Each raga is classified based on its key combination, which is unique. Ragas that are derived from Melakartha ragas are called Janya ragas. There are numerous Janya ragas for each Melakartha raga. The main difference between Melakartha ragas and Janya ragas lie with the number of notes present and their arohana and avarohana patterns. Janya ragas require should consist minimum of five notes from a Melakartha raga. Moreover, unlike

Their parent Melakartha ragas, arohana and avarohana patterns need not be the same for Janya ragas.

Carnatic raga recognition helps in identifying the ragas in a song, which would highly assist Musicologists and aspiring musicians. Raga recognition also helps in filtering songs according to their ragas.

Computational musicology is an emerging and trending research area. Many works have been carried out in raga identification in Indian Classical music. Different attributes that are used for raga recognition are the usage of notes, Archana and avarohana patterns, gamaka, pakad, vadi, and time. In1, the frequencies in the song were captured at specific intervals using Pitch Class Distribution methods [PCD]. The voice of the singer was isolated from the orchestration using the separation algorithm and segmented using the segmentation algorithm. Then the singer was identified to get their fundamental frequency, after which, string matching was used to identify the notes and thus the raga. In this paper, the frequency was considered relative to the fundamental frequency of the singer. The pitch distribution method was used method for raga identification. However, the problem with this method was that in most of the attempts, the voice of the singer had to be isolated from the song as the voice of the instruments created bafflements in the process. Though this process yielded reasonably good results, the process was error-prone. Vadi, Time and Pakad were used as parameters for raga identification in Hindustani music in2. The proposed method used fuzzy logic to find the interrelationship between the swarms. In this paper, importance is given to Samay (Time) when the raga is sung. The author claimed that the mood of each raga shows its prominence at a particular Samay which was used as the attribute for classification.

Vadi, Time and Pakad were used as parameters for raga identification in Hindustani music in2. The proposed method used fuzzy logic to find the interrelationship between the swaras. In this paper, importance is given to Samay (Time) when the raga is sung. The author claimed that the mood of each raga shows its prominence at a particular Samay which was used as the attribute for classification.

3. `Problem Statement

The use of statistical and probabilistic tools in musicology is not new. A strong theoretical grounding in Computational Musicology has sparked interest in the subject of Automatic Raga Recognition (ARR) in recent years. In a crude form, 'Raga Recognition' refers to techniques for identifying the raga in which an artist performs his rendition. Due to the complex nature of a raga, as well as nuanced differences between several ragas, this is not a trivial problem. Moreover, in Hindustani Music, the tonic is neverfixed, i.e. An artist can perform the same raga in different tonic scales on different occasions. For this reason, ARR is often also accompanied by,or preceded by tonic identification.

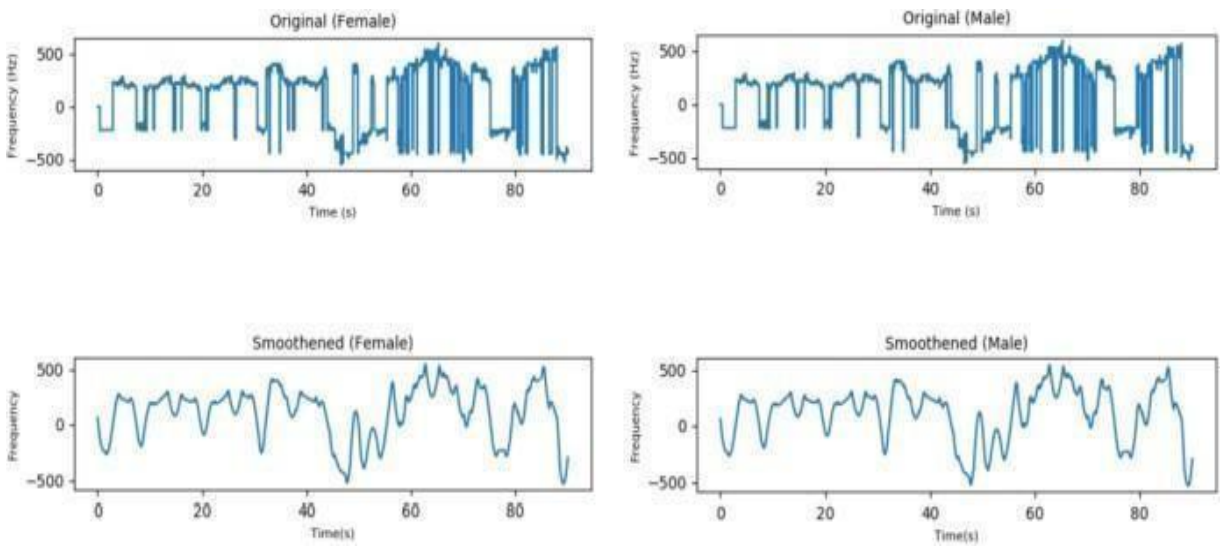
4. Method

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 shows an overview of the project and the process that DSP and ML performs to obtain the output.

Steps 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 was used. In addition to this, a tutorial 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 filter. Figure 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.



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:

$<f1>, \ldots, <fn>$

Step 3: 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 andsoon 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 inthis project as well. Large amounts of data are fed into the algorithm so that it starts recognizing different patternsand 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 inTable 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.

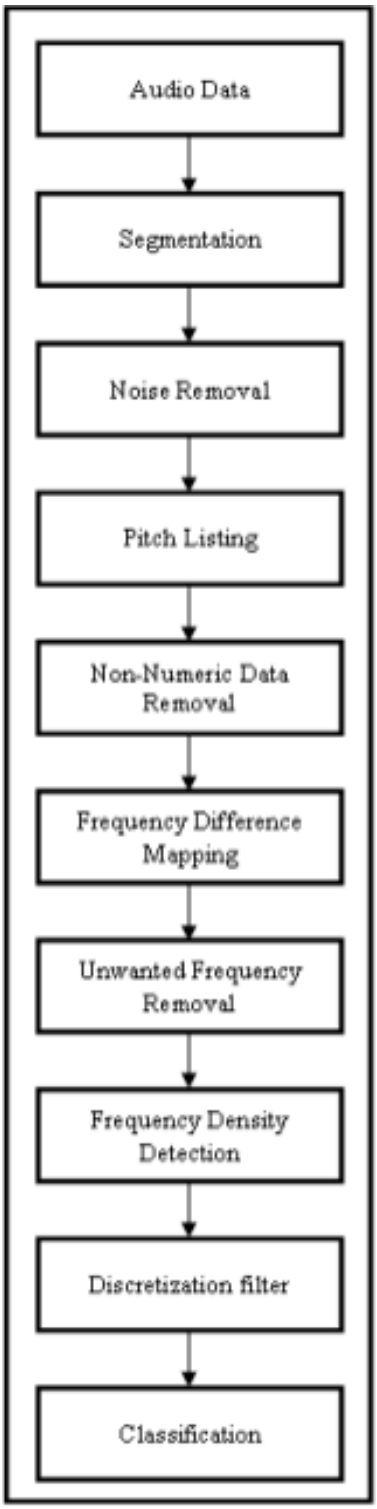
Table 1: One-Hot encoding example

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

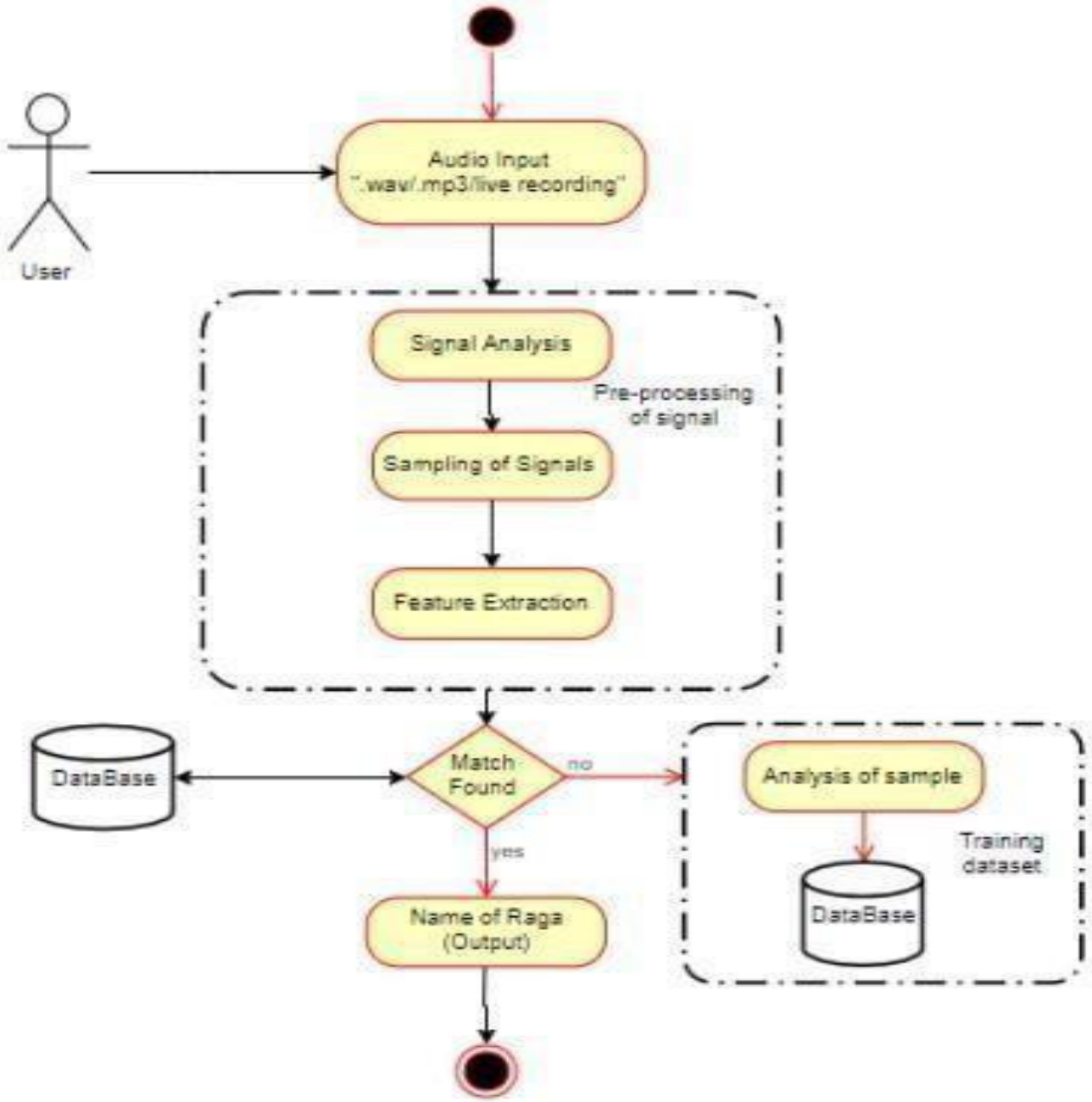
There are several layers in a neural network that contribute in streamlining the classification process. These layers are made up on nodes. The nodes have weights that enhance or dampen the input data. This helps the network know how important a particular input is and helps find the most efficient way 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.

5. Flow diagram



6. Use-case diagram



7. Methodology

Teaching Indian classical music is an excellent example of similarity-based learning. The teacher renders a melody, and the students imitate the teacher to be as like the teacher's rendering as possible. The similarity is a key element of machine learning. Recent studies proved that every machine learning algorithm that uses gradient descent and that includes deep learning, is essentially a kernel machine, where the kernel function measures similarity between data points. It is quite appropriate that deep learning is used to induce similarity-based music learning. At the crux of the methodology is therefore deep learning and specifically LSTM, as detailed.

8. Literature survey

In existing systems, work to date has been done on the basis of a static level using the characteristics of raga. Raga is identified on the basis of its fundamental characteristics: Arohana-Avarohana, Pakad, Gamakas, Swara Choice, Vadi, Time and Season, etc. These characteristics help to analyze and frame different computational techniques for the identification of raga. Pandey et.a.. It provides approximately 70 percent-80 percent accuracy with respect to the ML algorithm data set. Also, these methods are not real time methods.

A brief introduction to raga is being discussed. Characteristics of the raga that make them sufficient to be distinguishable from each other. Different techniques are better than other techniques depending on the input parameters and the restriction on the input and the method. Limited database containing a limited number of ragas. Incorrect extraction of the pitch. Manual detection of tonic. The assumption has been made for different algorithm parameters. Various restrictions on inputs, such as a limitation on singers, number of swaras, length of time, and monophonic type. No machine-learning techniques have been used. Note this Prefer method, which is independent of input parameters. Provide a large amount of dataset to obtain a better and more accurate result.

Try to combine 2 techniques, n-gram & pitch-class profiles in order to get a better result than other classical methods. Comp-Music dataset and our own dataset show that the combination of pitch-class profiles and n-gram histograms actually improves performance. The best accuracy of our approach is 83.39 percent. It is limited to only 4 to 5 ragas. There are problems with the classifiers and the pitch profile class. The combination of the male and female voices as well as many other instruments may affect the accuracy of the result. Remark on this paper uses combinations of techniques that provide better results and accuracy than conventional methods. The combination of different voices and instruments may lead to a wrong result.

Incorrect extraction of pitch. Manual detection of tonic. Assumption has been made for different algorithm parameters. Various restrictions on inputs, such as limitation on singers, number of swaras, length of time, monophonic type. There are no machine learning techniques used. Only the static work is done through identification[3]. Remark on this paper is the lack of a database and the incorrect extraction of features results in an incorrect result. Machine learning is going to provide a better result. Assumption could go wrong.

Using a classifier, we can identify raga, so that we can correlate this raga with its respective rasa to identify emotions in music. For a better result, more comprehensive dataset is needed for the K-NN classifier. We can only get approximate values by using soft computing techniques such as fuzzy logic. The SVM classifier is difficult to handle scale and multiple instruments. K-NN may cause problems with gammakas and pitch extraction. Classifier of Naïve Bayes, the identification of raga is very difficult and gives less accuracy. The remark in this paper is that soft computing techniques are used to archive results. The classifiers used have limitations on the results and the extracting features.

The algorithms used to give the most accurate result for two types of ragas. The system needs to be improved with the Hidden Markov model. A based approach where we combine the features of low levels. And HMM for better and more accurate identification. Singers and audios are only specific and limited. Limited test data may lead to a misleading decision. This one. The basic disadvantage of the system is the assumption of the fundamental frequency and therefore the determination of the fundamental frequency is our next task. A review of this paper is a new "Hidden Markov" method used to improve the outcome. The result is limited by different singers and styles. Fundamental frequencies are considered as one of the disadvantages due to inaccuracy.

9. Hardware and software specification

Hardware and Software	Characteristics
Memory	8GB
Processor	Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz
Graphics	NVIDIA GeForce RTX 2060 4GB GDDR6
Operating System	Windows 11 home 64

10. Implementation

Implementation Details:

Python was used as the programming language to develop this project because of its simplicity, consistency, platform independence, and access to great libraries for machine learning(ML). The version used is Python 3.7.10. TensorFlow was used as the model backend. The TensorFlow Extended (TFX) is an end-to-end platform for deploying production machine learning pipelines. In this project, the pusher component of the TensorFlow Extended platform known as the "TensorFlow serving" was used to serve the machine learning model. By default, TensorFlow serving provides the REST endpoint for users to get their predictions. The important machine-learning libraries used are JSON and Librosa.

A function was written to automate the process of downloading songs from Dunya using the PyCompmusic library which provides a wrapper to Dunya APIs.

Initially, compmusic.Dunya.Hindustani.get_recordings(recording_detail=True) API was used to generate a JSON file containing a list of song Ids, the title of songs, and raga type. This JSON file was then converted to CSV for easy parsing. All the song ids in the CSV file are iterated through to download the corresponding mp3 songs into separate raga folders as per the raga type associated with it using the compmusic.Dunya.Hindustani.download_mp3(recordingid, location) API. The final dataset contains 25 full-length recordings per raga and features 5 different ragas, hence 125 recordings.

The HCM dataset obtained was loaded into a Google Colab Pro notebook for preprocessing and feature extraction. The librosa library was used to extract audio features which are understandable to a machine learning algorithm. Each mp3 audio data was converted into 13 Mel-frequency cepstral coefficients (MFCCs) features and stored into a JSON file along with their corresponding raga label. The MFCCs are a powerful representation of an audio signal as it scales the frequency to match more closely what the human ear can hear.

11. Result

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 dataas required to run a Neural Network algorithm. The Ragas used were as follows:

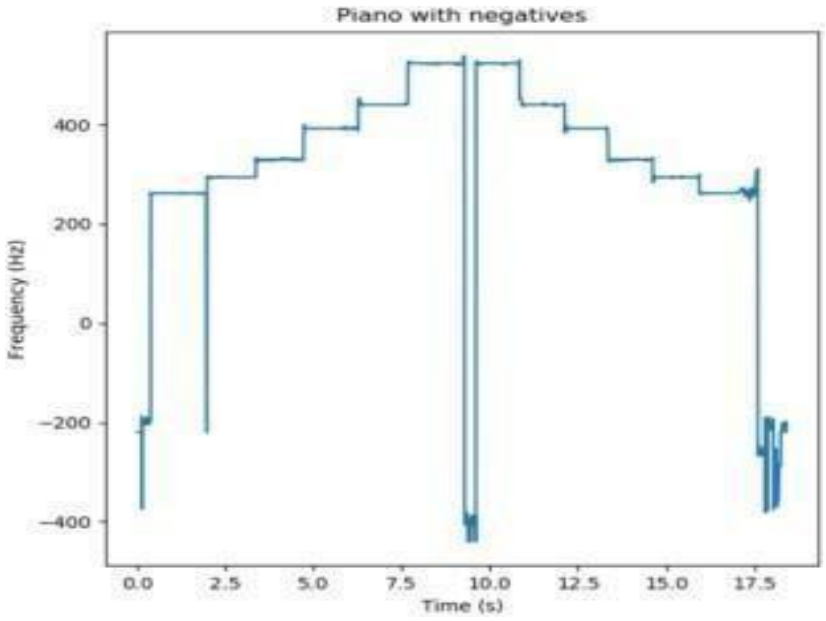
1. Mayamalavagowla
2. Abhogi
3. Hindolam
4. Kalyani
5. Madhaymavti
6. Mohanam
7. Panthuvarali
8. Sankarabharanam
9. Shanmukhapriya
10. Vasantha

These Ragas all belong to Carnatic music.

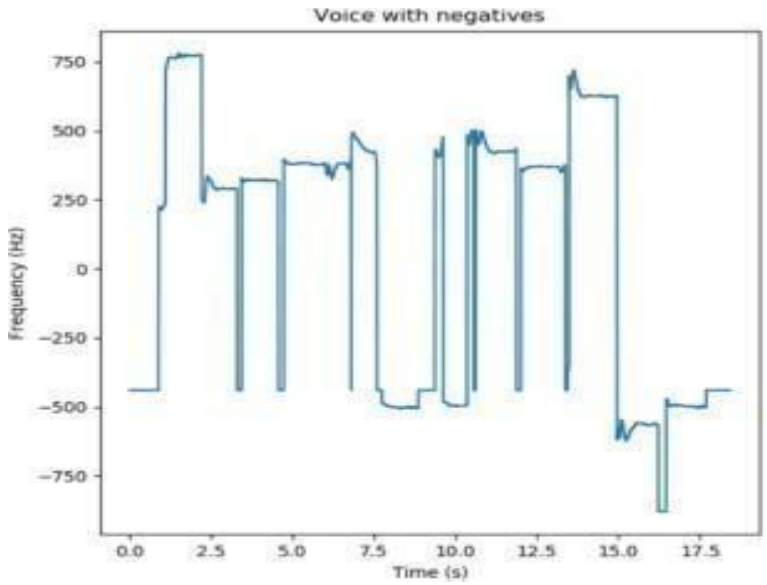
Pre-Processing :-

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 timelengths 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 thatthis 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.



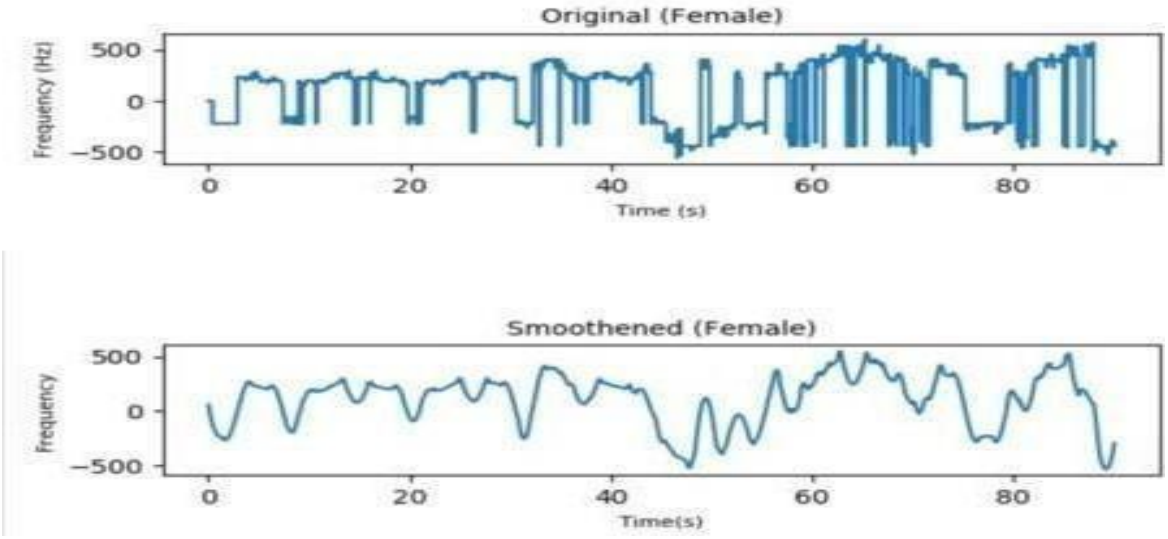
DFT graph of Piano sample of Mohana Raga



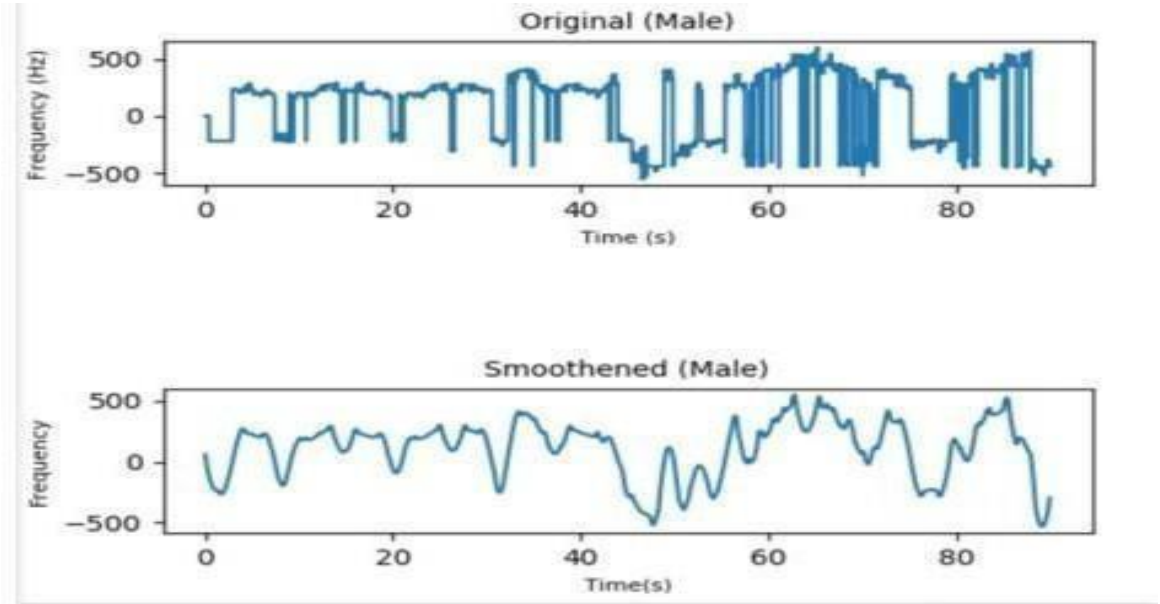
DFT graph of vocal sample of Mohana Raga

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 ofthe data in the feature vector.



Before and after filtering of female vocal sample



Before and after filtering of male vocal sample

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.

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:-

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.

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

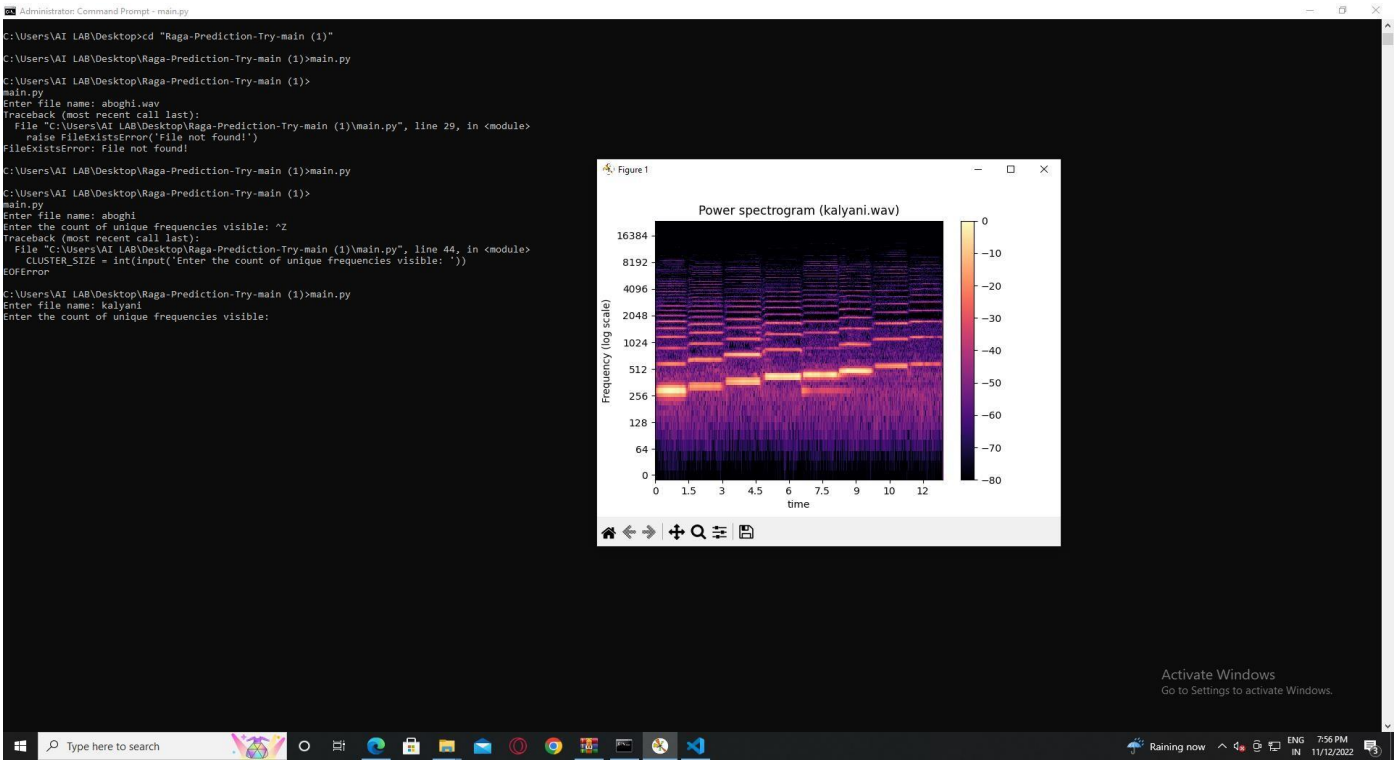
Test Case 3 :-

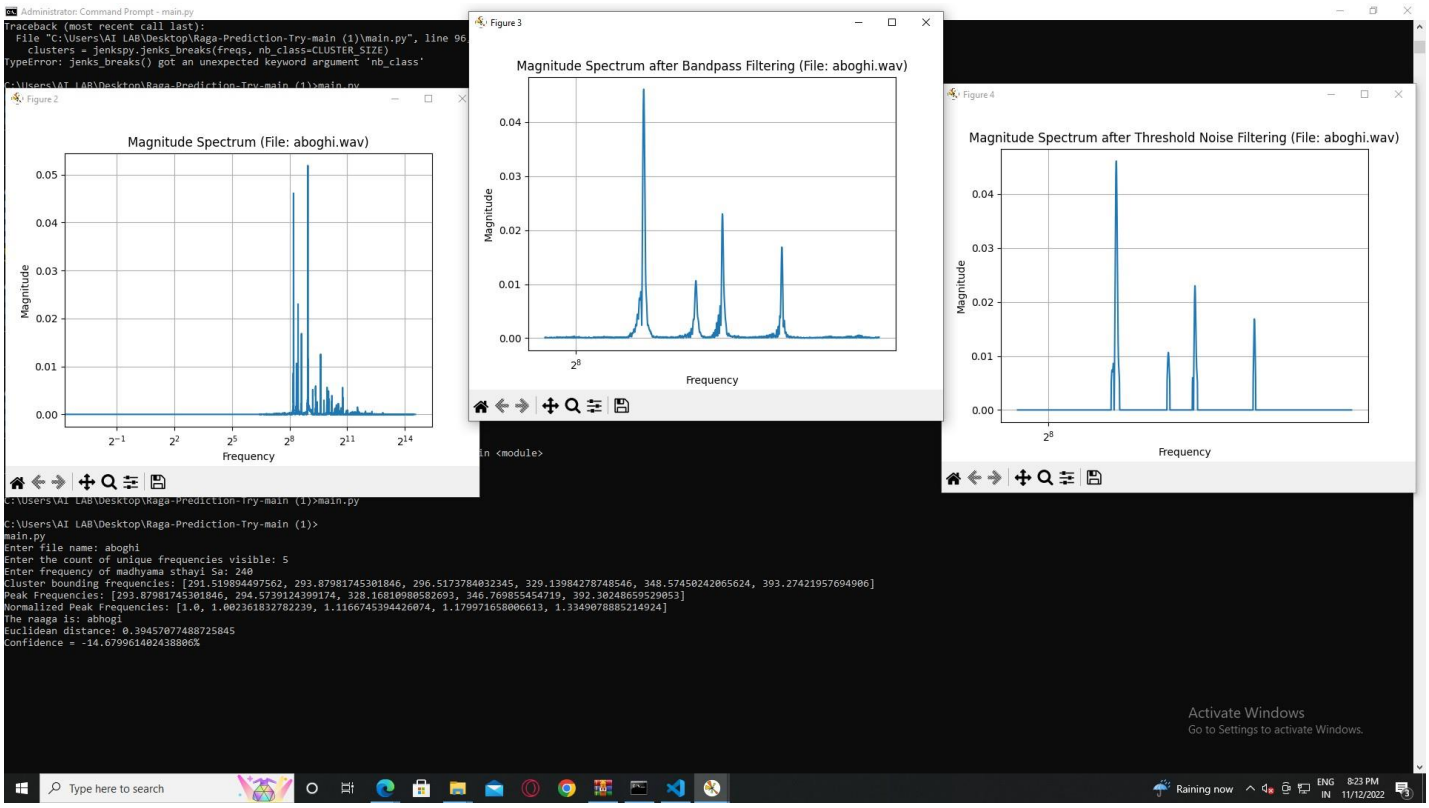
In the third test case, the length of the samples was increased to 35 seconds. This is because 20 seconds became too small to train the algorithm. More Dense layers were also added before and after the LSTM layer. Table 4 shows a summary of these results.

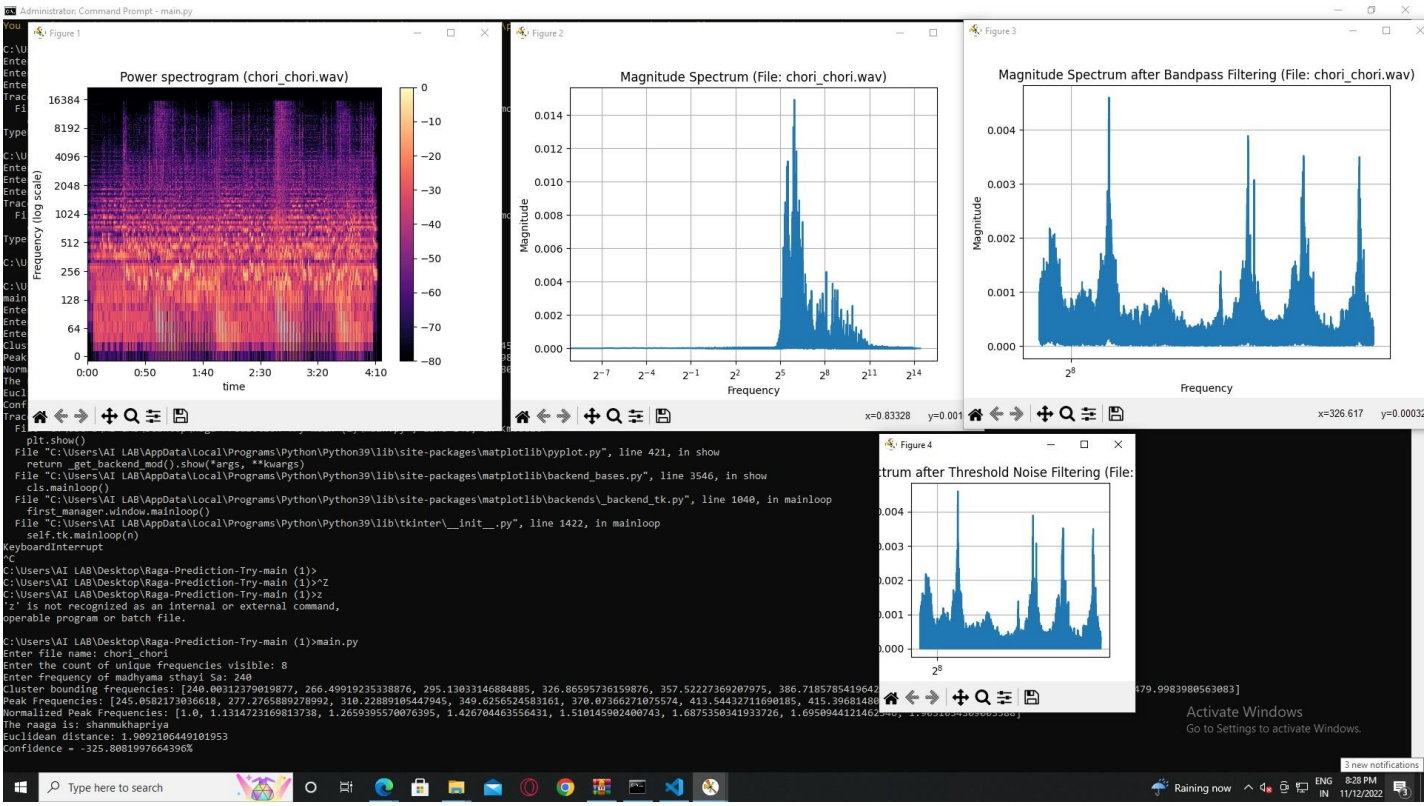
Summary of the test case 3:-

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

Input and Output




Traceback (most recent call last):
File "C:\Users\AI LAB\Desktop\Raga-Prediction-Try-main (1)\main.py", line 96,
clusters = jenkspy.jenks_breaks(freqs, nb_class=CLUSTER_SIZE)
TypeError: jenks_breaks() got an unexpected keyword argument 'nb_class'
Figure 2
Magnitude Spectrum (File: aboghi.wav)
Magnitude
Frequency
Figure 3
Magnitude Spectrum after Bandpass Filtering (File: aboghi.wav)
Magnitude
Frequency
Figure 4
Magnitude Spectrum after Threshold Noise Filtering (File: aboghi.wav)
Magnitude
Frequency
C:\Users\AI LAB\Desktop\Raga-Prediction-Try-main (1)>main.py
C:\Users\AI LAB\Desktop\Raga-Prediction-Try-main (1)>
main.py
Enter file name: aboghi
Enter the count of unique frequencies visible: 5
Enter frequency of madhyama sthayi Sa: 240
Cluster bounding frequencies: [291.519894497562, 293.87981745301846, 296.5173784032345, 329.13984278748546, 348.57450242065624, 393.27421957694906]
Peak Frequencies: [293.87981745301846, 294.5739124399174, 328.16810980582693, 346.769855454719, 392.30248659529053]
Normalized Peak Frequencies: [1.0, 1.002361832782239, 1.1166745394426074, 1.179971658006613, 1.3349078885214924]
The raaga is: abhogi
Euclidean distance: 0.39457077488725845
Confidence = -14.679961402438806%
Activate Windows
Go to Settings to activate Windows.
Type here to search
Raining now
ENG
IN
8:23 PM
11/12/2022


Power spectrogram (chori_chori.wav)
Frequency (log scale)
time
Magnitude Spectrum (File: chori_chori.wav)
Magnitude
Frequency
Magnitude Spectrum after Bandpass Filtering (File: chori_chori.wav)
Magnitude
Frequency
Figure 4
trum after Threshold Noise Filtering (File:
plt.show()
File "C:\Users\AI LAB\AppData\Local\Programs\Python\Python39\lib\site-packages\matplotlib\pyplot.py", line 421, in show
return _get_backend_mod().show(*args, **kwargs)
File "C:\Users\AI LAB\AppData\Local\Programs\Python\Python39\lib\site-packages\matplotlib\backend_bases.py", line 3546, in show
cls.mainloop()
File "C:\Users\AI LAB\AppData\Local\Programs\Python\Python39\lib\site-packages\matplotlib\backends_backend_tk.py", line 1040, in mainloop
first_manager.window.mainloop()
File "C:\Users\AI LAB\AppData\Local\Programs\Python\Python39\lib\tkinter__init__.py", line 1422, in mainloop
self.tk.mainloop(n)
KeyboardInterrupt
^C
C:\Users\AI LAB\Desktop\Raga-Prediction-Try-main (1)>
C:\Users\AI LAB\Desktop\Raga-Prediction-Try-main (1)>^Z
C:\Users\AI LAB\Desktop\Raga-Prediction-Try-main (1)>z
'z' is not recognized as an internal or external command,
operable program or batch file.
C:\Users\AI LAB\Desktop\Raga-Prediction-Try-main (1)>main.py
Enter file name: chori_chori
Enter the count of unique frequencies visible: 8
Enter frequency of madhyama sthayi Sa: 240
Cluster bounding frequencies: [240.00312379019877, 266.49919235338876, 295.13033146884885, 326.86595736159876, 357.52227369207975, 386.71857854196427
Peak Frequencies: [245.0582173036618, 277.2765889278992, 310.22889105447945, 349.6256524583161, 370.07366271075574, 413.54432711690185, 415.39681480
Normalized Peak Frequencies: [1.0, 1.1314723169813738, 1.2659395570076395, 1.426704463556431, 1.510145902400743, 1.6875350341933726, 1.69509441214625
The raaga is: shanmukhapriya
Euclidean distance: 1.909210644910195
Confidence - -325.8081997664396%
Activate Windows
Go to Settings to activate Windows.
Type here to search
Raining now
ENG
IN
8:28 PM
11/12/2022
3 new notifications

12. 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 eventhe most complicated forms of art can be combined with engineering. It has to be extendedto 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.

13. Reference

[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- encodingwhy- 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/.