Appreciating Indian Music using AI

A Project Report
Presented to
The Faculty of the College of
Engineering

San Jose State University
In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Software Engineering

By
Chetan Kulkarni, Lokesh Vadlamudi, Nupur Yadav
May 2021

APPROVED

Dr. Vishnu Pendyala, Project Advisor

Prof. Dan Harkey, Director, MS Software Engineering

Prof. Rod Fatoohi, Department Chair

ABSTRACT

Appreciating Indian Music using AI

By Chetan Kulkarni, Lokesh Vadlamudi, Nupur Yadav

Hindustani classical music is a multi-faceted and ancient form of musical art that has its roots in the Indian subcontinent. The main key element in Hindustani classical music is called raga (or "raag"). A raga is a musical theme or melodic framework. It can be thought of as a scale created by choosing a specific set of notes from within an octave where different combinations of notes evoke different moods and inspire different feelings. Hindustani classical music is gaining popularity all around the world, especially in the west. Many people want to learn to sing Hindustani classical music for personal passion or interest.

However, currently, we don't have enough resources to learn Hindustani classical music because it is very complicated. Many ragas can be based on the same scale with several improvisations which are very difficult to distinguish, and there are very few musicians who have mastered it. These improvisational patterns are not even concretely defined in any format and are just passed through a fading oral mentor-student tradition.

In this project, we aim to build a music website to enable real-time recording and uploading songs that can help the user in getting immediate feedback on the musical notes they are singing. We will be using deep learning to identify ragas using audio features such as scale, pitch, tone and provide feedback and suggestions to the users. The website can be accessed from anywhere, anytime, and free of cost to encourage people to follow their musical passion.

Acknowledgments

The authors are deeply indebted to Dr. Vishnu Pendyala for his invaluable comments and assistance in the preparation of this study. It was a pleasure working alongside him on this endeavor.

Table of Contents

List of Figures

List of Abbreviations

HCM: Hindustani Classical Music
ICM: Indian Classical Music
LSTM: Long Short Term Memory
BiLSTM: Bidirectional Long Short Term Memory
GRU: Gated Recurrent Units
RNN: Recurrent Neural Network
MFCCs: Mel-frequency cepstrum coefficients
CNN: Convolutional Neural Network
ML: Machine Learning
AI: Artificial Intelligence
MIR: Music Information Retrieval
AIR: Audio Information Retrieval
TFX: TensorFlow Extended
GCP: Google Cloud Platform

Chapter 1. Introduction

Hindustani Classical music is an ancient musical form predominant in northern parts of India, Pakistan and Bangladesh. It focuses mainly on melodic development. The key element in Hindustani Classical music is raga (also called "raag"). A raga is a musical theme, or a melodic framework created by choosing a specific set of notes (swaras), on a scale, ordered in melodies with musical motifs. It has characteristic intervals, rhythms, and embellishments which is used as a basis for improvisation. [1, 2] defines raga technically as a collection of melodic atoms and a technique to develop them. Hindustani classical music focuses more on the space between the notes than the notes themselves. A musician playing a raga may use the same notes but, may improvise or emphasize on certain degrees of the scale evoking a mood that is unique to each raga. The following Figure 1 shows how raga Bhairav scale looks like.

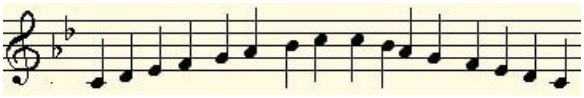

Figure 1: Raga Bhairav scale

As Hindustani Classical music is gaining popularity all around the world, more and more people want to learn this form of music. So, we have proposed a system for the identification of raga which will help users know the type of raga present in a song for them to practice and, also to check what they are practicing is a correct raga or not, with a provided confidence score. Hence, creating an ecosystem where Hindustani Classical music can thrive and generate interest globally which is our core objective behind designing this service.

1.1 Problem Statement

Each raga is constructed from five or more musical notes and can be written on a scale, but many ragas can be based on the same scale with several improvisations such as timing between notes; sustain, attack of each note. These improvisational patterns are very difficult to determine and are not properly documented in any form and also known to only few musicians who have mastered it. This melodic variation and inconsistent temporal spacing makes raga identification a very challenging problem.

1.2 Proposed Solution

In this project, we have proposed a deep learning solution for raga identification using a bidirectional LSTM based RNN architecture. Initially the system generates Mel-frequency cepstrum coefficients (MFCC) representations of the input audio data which are then fed to the BiLSTM RNN architecture for raga identification.

Further, the website enables users to either upload songs or do live recording and identify the type of raga present in the song along with the generated confidence score. The system also gives recommendations to users for the type of raga they are interested in.

Chapter 2. Literature Survey

This chapter provides an overview of Music Information Retrieval (MIR), various pre-processing techniques for extracting audio features, deep learning techniques like convolutional and recurrent neural networks for MIR tasks and state-of-the-art raga analysis.

2.1 Music Information Retrieval

Music Information Retrieval (MIR) is an interdisciplinary research field which focuses on extracting information from music and its applications [3]. It has many real-world applications such as recommender systems, instrument recognition, automatic categorization, automatic music transcription, and music generation. Beside these potential applications extracting the raga from Hindustani Classical music is quite challenging because of the inconsistent temporal spacing between notes and the intense variation in its rhythmic patterns [4].

2.2 Audio Feature Extraction

Feature extraction is the most important part in the machine learning process. The performance of any machine learning model depends on the features it is trained and tested on. There are several techniques to extract features for audio data such as temporal domain, frequency domain, cepstral domain, wavelet domain and time-frequency domain features [5]. The important music related features that are often used for classification, prediction

and recommendation algorithms includes zero crossing rate, spectral centroid, spectral roll-off, Mel-Frequency Cepstral Coefficients (MFCCs) and chroma frequencies.

The zero-crossing rate defines the rate at which the signal changes from positive to negative or back. It has many applications such as music discrimination, music genre classification [6], etc. Spectral centroid indicates the location of center of mass of the spectrum. It is also called brightness feature of a sound as it describes the brightness of the sound. This feature is highly used as a measure of timbre of music, music classification and music mood classification [7]. Spectral roll-off is the measure of the shape of the signal. It represents the frequency such that 95% of the signal energy is contained below this frequency. Its applications include musical instrument classification, music classification [8], music genre classification [9,10,11,12], and audio-based surveillance systems [13]. The Mel frequency cepstral coefficients (MFCCs) of a signal are derived from the cepstral representation of an audio clip and is a key feature in any audio signal processing. They represent a small set of about 10-20 features which concisely describe the overall shape of a spectral envelope. MFCCs can mimic the human voice very closely as the frequency bands are equally spaced on mel-scale. MFCCs have wide variety of applications such as music information retrieval [14], speech enhancement [15], speech recognition [16], music genre classification [17], vowel detection [18] etc. Another important feature for representing music audio is chroma features or chromagrams. In chromagram the entire spectrum is projected into 12 bins which represents the 12 distinct chroma (or semitones) of the musical octave. We have also used MFCCs and chroma features for representing our music data.

2.3 Deep Learning for Raga Analysis

Many approaches have been used in the past for raga recognition. Most recent approaches involve extracting features such as spectrograms, MFCCs, chromograms from music data and then applying a clustering algorithm or a classification algorithm for extracting raga information [19, 20, 21, 22, 23, 24]. However, the existing systems cannot detect raga in real-time as they require the entire audio data to be processed first and then generate a raga prediction. Having a raga evaluation system that can predict raga in real-time can help users practice well because of live feedback. It can be more like a tutoring service.

The other approaches mentioned in [23, 25, 26] require manual extraction of features like pitch histograms that are unable to capture music features such as note transitions essential for raga analysis. Also, hand-crafting features can be time-consuming and tedious especially when we have huge number of classes as in our case. Also, the existing approaches for classifying Hindustani Classical music cannot be applied to other AIR tasks as they make assumptions about the note emphasis [20] and pitch distribution [23, 25, 26, 27] for music classification and make them limited in scope.

Some recent works like [28, 29] have used deep learning techniques like convolutional neural networks and recurrent neural networks for feature extraction and processing raga information from the music data. However, these works also require human intervention for extracting features along with some other additional inputs like composition annotation. Also, these cannot inspect music features like overall pitch distributions and note ornamentations separately which are essential in music data for raga classification.

Chapter 3. Methodology

This chapter discusses the steps followed to solve the problem of raga identification using an AI system. The first hurdle while solving this problem was to obtain a Hindustani Classical music dataset to train and test our AI model. We found few Indian Classical music (ICM) datasets but not many datasets were publicly available for Hindustani Classical music (HCM) with pre-defined raga labels, as this is still an on-going research problem. So, we created our own dataset in two ways. **Firstly,** by manually downloading songs from YouTube and secondly, by using PyCompmusic library which provides a wrapper to the Dunya API. Dunya comprises the music corpora and related software tools that have been developed as part of the CompMusic project. Figure 2 demonstrates the steps followed to download and prepare dataset to be further used by our AI model.

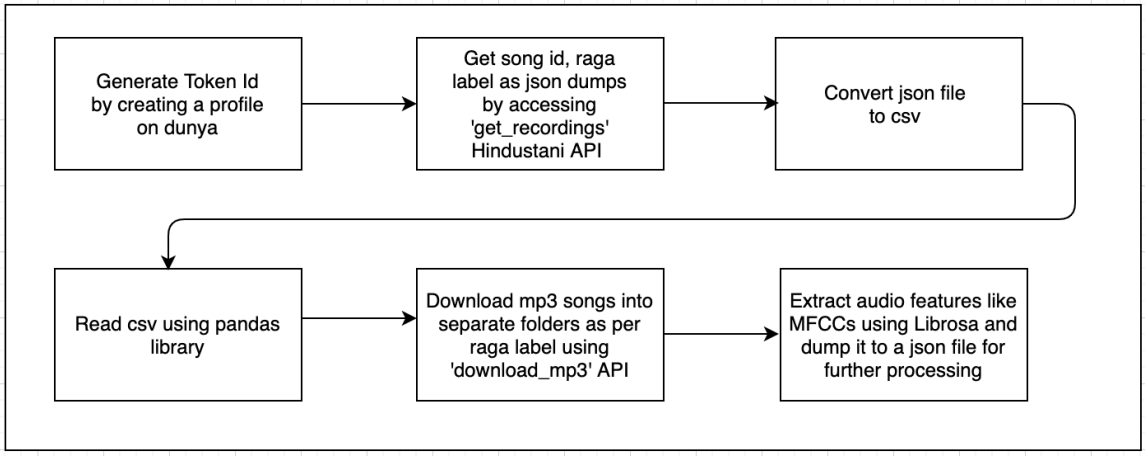

Figure 2: Steps to download and prepare HCM dataset

Once the audio features are generated, they are then fed to a ML model for raga identification.

3.1 Model Architecture Diagram

We have used a bidirectional LSTM RNN architecture for our raga identification problem which this is a very popular architecture used for sequence classification and sequence to sequence learning tasks. This neural network contains 3 LSTM layers followed by a simple RNN layer followed by dense layers. This model trains itself on MFCCs features to form an output in the form of a five-dimensional array representing the probability scores towards the five different classes we have chosen. The highest score representing the raga label for that song. We have used softmax activation function in the last layer to generate the probability scores and sparse_categorical_crossentropy as the loss function. The architectural block diagram of the model is shown in below Figure 3.

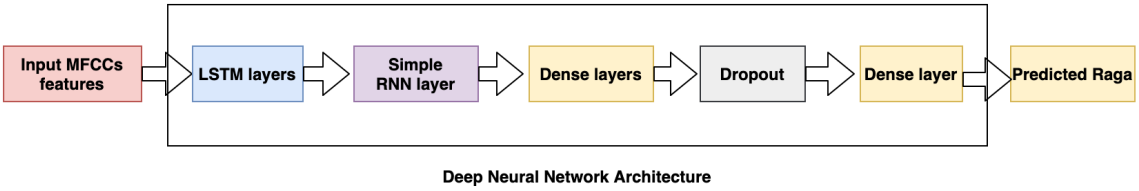

Figure 3: Architectural block diagram for deep neural network model

3.2 Loss Function and Optimizer

We have used sparse_categorical_crossentropy loss function which is defined as categorical crossentropy with integer targets, and we also have our targets in the integer form. This is called sparse since the target representation requires much less space than one-hot encoding. For example, a batch with b targets and k classes needs b * k space to be represented in one-hot, whereas a batch with b targets and k classes needs b space to be represented in integer form.

The optimizer used is adam. We experimented a bit with the optimizers and different learning rates and found adam optimizer with a learning rate of 0.001 worked the best. Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments. It combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems.

Chapter 4. Implementation Details

This chapter discussed the as tools and environments used for the project. It further elaborates the data collection process, data preprocessing techniques, model training and evaluation metrices used in developing the AI model for raga prediction.

4.1 Tools and Environments

We have used Python as the programming language to develop our project because of its simplicity, consistency, platform independence, and access to great libraries and frameworks for machine learning (ML) and artificial intelligence (AI). The version used is Python 3.7.10. We have used Tensorflow as the model backend and the version used is 2.4.1. The important machine learning libraries used are Keras, scikit-learn and Librosa. We have used Google Colab Pro with GPUs for building our AI model and Django a python-based open-source we framework for developing our website. The following figure shows the GPU configuration used in Colab Pro.

4.2 Dataset Collection

Dataset collection is the most crucial part for any machine learning task, but we could not find any relevant dataset for Hindustani Classical music available online, so we decided on creating our own dataset. The dataset collection was done in two parts.

1. Manually downloading songs from YouTube.
2. Downloading songs from Dunya using PyCompmusic library

We wrote our own function to automate songs downloading from Dunya using PyCompmusic library which provides a wrapper to Dunya APIs. Initially we used compmusic.dunya.hindustani.**get_recordings**(*recording_detail=True*) API to generate a json file containing list of song Ids, title of songs and raga type. Then we converted the json file to csv for easy parsing. We then iterated through all the song ids in the csv file to download the 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.

4.3 Data Preprocessing and Feature Extraction

The HCM dataset obtained was loaded into a Google Colab Pro notebook for preprocessing and feature extraction. Initially the dataset was in the following format.

We then used librosa library to extract audio features which are understandable to a machine learning algorithm. We converted each mp3 audio data into 13 **Mel-frequency cepstral coefficients** (**MFCCs**) features and stored them into a json file along with their corresponding raga label. The MFCCs are a very powerful representation of an audio signal as it scales the frequency in order to match more closely what the human ear can hear. Figure 4 shows the sample of the extracted MFCCs features with their corresponding raga label.

mfcc1	mfcc2	mfcc3	mfcc4	mfcc5	mfcc6	mfcc7	mfcc8	mfcc9	mfcc10	mfcc11	mfcc12	mfcc13	label
-338.261033	113.617955	-14.052405	8.551622	-5.788107	-5.635894	-16.244627	-9.675543	-10.974425	1.525304	-2.864719	7.664231	-0.295875	madhuvanti
-272.303322	119.241958	-11.244106	11.319374	-2.549124	-5.570148	-12.734917	-7.217122	-8.655407	1.841118	-0.864794	7.683995	1.280020	madhuvanti
-509.803522	73.650876	-8.511494	10.525128	-0.658952	-0.146639	-6.690942	-1.944142	-4.960757	2.031218	-2.907506	6.473613	-0.126429	madhuvanti
-260.629208	121.702862	-10.383890	10.917586	-1.730320	-4.178613	-13.270461	-6.671383	-9.754154	0.945042	-2.934038	6.622784	3.343194	madhuvanti
-695.559207	4.576745	-0.495340	0.662960	-0.007068	0.012834	-0.409880	-0.082544	-0.259814	0.133248	-0.168076	0.420270	-0.027064	madhuvanti
...	...	...	...	...	...	...	...	...	...	...	...	...	...
-198.717986	129.919135	-15.070532	15.536915	-15.608483	0.439417	0.440207	-9.561070	-7.697309	3.332878	-15.943490	-2.292459	-12.214308	multani
-227.857793	120.694875	-18.167212	16.042029	-16.922651	-7.386678	-4.247232	-12.899577	-10.193587	1.571845	-19.803650	-0.746796	-14.709724	multani
-187.687285	109.692590	-35.086028	6.101116	-9.711220	-13.607133	-10.112420	-7.792900	-2.879457	1.073179	-7.411411	2.969039	-1.413216	multani
-192.349212	123.895564	-16.105062	19.597401	-12.457828	-4.877896	-3.930122	-10.114332	-9.244178	-0.755781	-14.367605	-2.424859	-12.324200	multani
-162.931756	114.775892	-40.825779	-0.814991	-9.189457	-10.207901	-8.421390	-5.949722	-2.424465	1.347141	-9.525109	-1.475691	1.210340	multani

Figure 4: Sample dataset after MFCC feature extraction from audio data

The following Figure 5 shows the MFCC representation of an audio clip in the form of a heatmap.

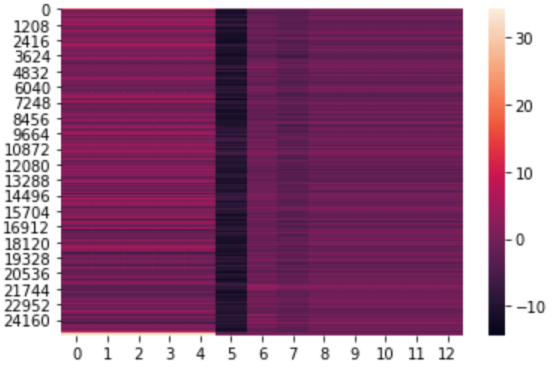

Figure 5: MFCC representation of Raga Ahir Bhairav audio clip

Further, we also extracted features like spectrogram and chromagram images to see if these features can help in building a better model for raga identification problem. Figure 6 and Figure 7 shows the spectrogram and chromagram images for raga bhairav sample audio file respectively.

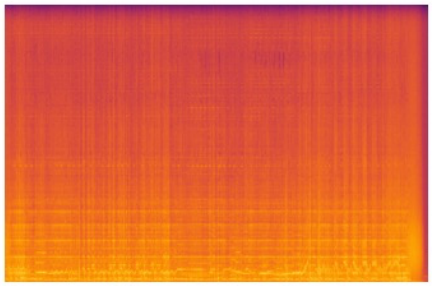

Figure 6: Spectrogram representation of Raga Bhairav audio clip

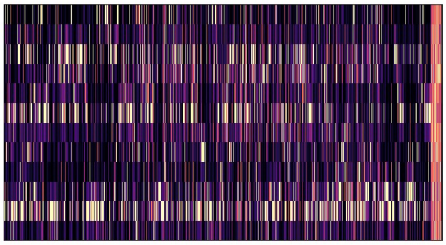

Figure 7: Chromagram representation of Raga Bhairav audio clip

4.4 Model Training

As the next step we trained our model on this pre-processed dataset. We used MFCCs features to train our bidirectional LSTM RNN model which is a very popular choice for sequential classification and sequence to sequence tasks. The train_test_split method from Scikit-learn library was used to divide the dataset into train, test and validation set. We allocated 25% of the dataset to test split and 20% of the train data to validation split. The BiLSTM RNN model was then trained on 75% of the dataset for 500 epochs. We used Adam optimizer with 0.001 learning rate and sparse_categorical_crossentropy as the loss function while training our model.

We also trained deep Covolutional Neural Network model and pretrained ResNet model on spectrogram and chromagram images extracted from the mp3 audio data. We thought of using pretrained ResNet model because in image classification tasks pretrained models have yielded better results in some cases. While training the ResNet model we only trained top layers so that we can use pre trained weights for our advantage.

4.5 Model Evaluation

We have evaluated our model on the test data and considered Accuracy as our key evaluation metrics for performance comparison because we had a balanced dataset. We also considered train and validation accuracies. We have used TensorBoard integration for observing the training process and tracking metrices like accuracy and loss.

Chapter 5. Experiments and Results

This chapter provides an overview of all the experiments performed as part of this work. Experiments were performed to come up with the best LSTM RNN model configurations for raga identification task using the MFCCs features as input. Later we also ran experiments with other audio features such as spectrograms and chromagrams. Since these were image features, we used CNN and pretrained models such as ResNet for training. The number of raga classes remained fixed to five across all experiments. We have evaluated model performance in terms of test accuracy achieved. Figure 8 shows the experiments performed and accuracies achieved.

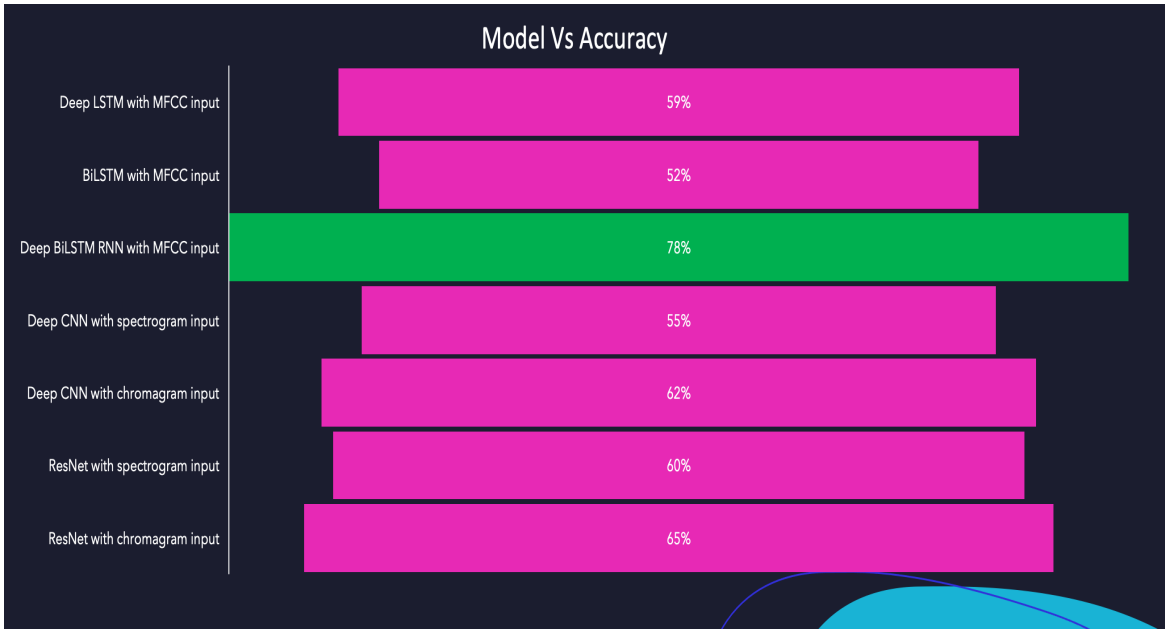

Figure 8 Graph representing Model Vs Accuracy

As the above experiments show Deep BiLSTM RNN architecture achieved the best accuracy of 78%. Hence, we have used this as our final model for raga prediction and further deployment.

Chapter 6. Model Deployment

Model deployment is one of the crucial stages in machine learning application. In order to serve thousands of users, we need our model to be highly scalable and optimized. This chapter provides an overview of model deployment architecture and infrastructure used to deploy the model and its services.

6.1 Deployment Architecture

In order to handle hundreds of requests at the same time, we needed to have a good cloud infrastructure. We have used Docker to create two separate images one for our Django app and another for our model which is being served using TensorFlow serving in TFX. Both the images are then deployed on a Kubernetes engine of GCP. Figure 8 shows the model and web deployment architecture. It shows the user accessing web application service which in turn uses model service to fetch predictions.

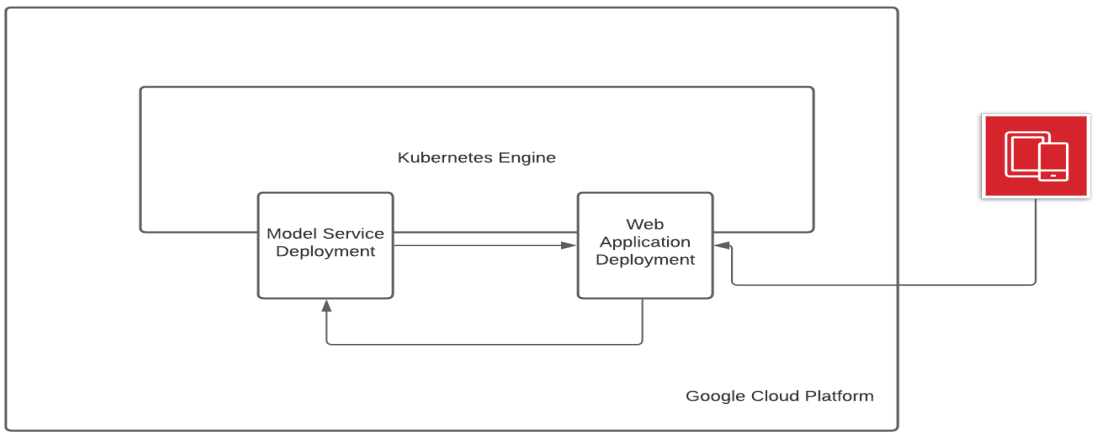

Figure 9: Model and web application deployment architecture

6.2 TFX

TensorFlow Extended (TFX) is an end-to-end platform for deploying production ML pipelines. In this project we have used the pusher component of TensorFlow extended platform known as TensorFlow serving to serve our ML model. By default, TensorFlow serving provides the rest endpoint for users to get their predictions.

6.3 Docker

Docker is a set of platform as a service products that use OS-level virtualization to deliver software in packages called containers. Docker enables developers to easily pack, ship, and run any application as a lightweight, portable, self-sufficient container, which can run virtually anywhere. Docker containers are easy to deploy in a cloud. The two main docker containers in this project are

1) Django web application Container

2) Raga Prediction Model Container.

`The Django application image was created with the help of a base python image. The docker compose file was configured including steps for copying Django content into root directory of base python image and starting the server at a specific port.

The model container was created using the base tensorflow/serving image. Figure 9 below shows the steps to create a custom TensorFlow serving image for the model.

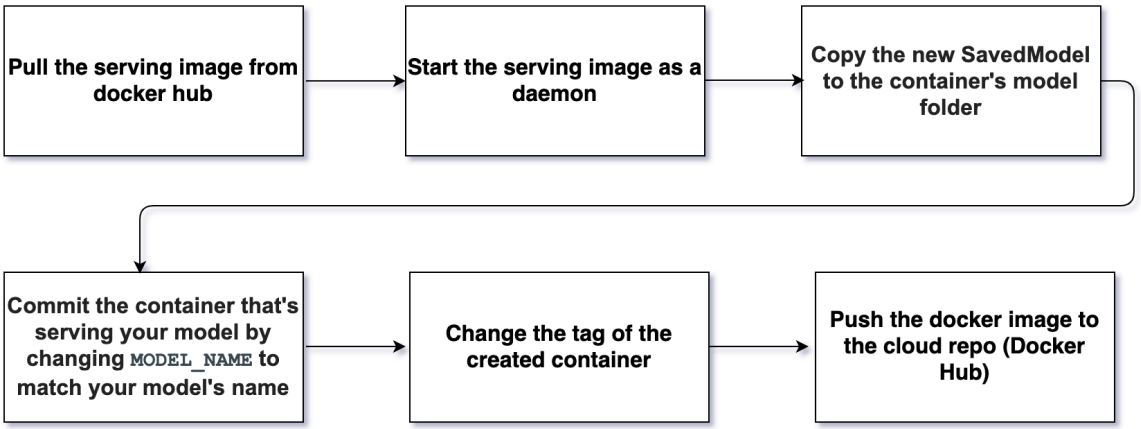

Figure 10: Steps to create a custom tensorflow serving image

The model protocol buffer file from our generated model are copied into the serve image. Later we push our custom model docker to cloud storage as discussed in the TFX section above. The cloud storage for our dockers is Docker Hub. GCP pulls the images from this storage for creating deployments on cluster.

6.4 Kubernetes

Kubernetes is an open-source container orchestration platform that enables the operation of an elastic web server framework for cloud applications. It offers **portability**, and faster, simpler deployment times. We have deployed both Django web application Container and Raga Prediction Model Container on the Kubernetes cluster of GCP. Figure 10 below shows the internals of virtual node machines in Kubernetes cluster.

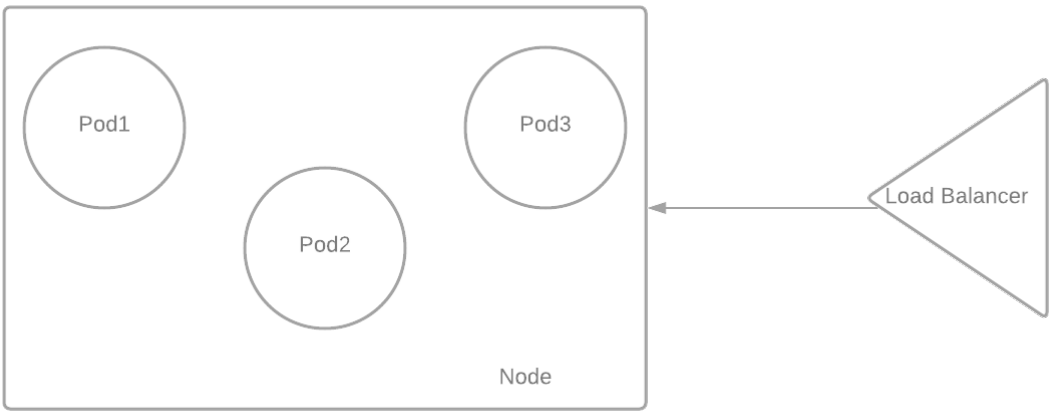

Figure 11: Internals of virtual node machines in Kubernetes cluster

Each node can contain many pods. And each pod can contain one or few containers. Generally, it is advisable to run a single container inside a pod for better stability. We have three identical pods in total, and load is distributed between the three with the help of a load balancer service.

Chapter 7. Conclusions and Future Work

In this work we have proposed a deep learning solution for the problem of raga identification. Through various experiments and validations, we found that the best deep learning solution uses 3 LSTM layers (2 BiLSTM and 1 LSTM) followed by a simple RNN layer which is further followed by 3 dense layers. The system was able to distinguish between five ragas with an accuracy of 78%. The system is adaptable and can be used for other AIR tasks as well.

The work is only tested on Hindustani Classical music dataset. As part of future work, it can be used to analyze different styles of music and instruments. Also, the HCM dataset we had was very limited so maybe we can gather more data in future to achieve better results.

References

[1] Krishnaswamy, A. (2004, October). Melodic Atoms for Transcribing Carnatic Music. In *ISMIR* (Vol. 2004, p. 5th).

[2] Krishnaswamy, A. (2004, August). Multi-dimensional musical atoms in South-Indian classical music. In *Proc. of International Conference on Music Perception and Cognition*.

[3] Downie, J. S. (2003). Music information retrieval. *Annual review of information science and technology*, *37*(1), 295-340.

[4] Gottlieb, R. S. (1977). *The major traditions of North Indian tabla drumming: Supplement*. Musikverlag Emil Katzbichler.

[5] Sharma, G., Umapathy, K., & Krishnan, S. (2020). Trends in audio signal feature extraction methods. *Applied Acoustics*, *158*, 107020.

[6] Ahrendt, P., Meng, A., & Larsen, J. (2004, September). Decision time horizon for music genre classification using short time features. In *2004 12th European Signal Processing Conference* (pp. 1293-1296). IEEE.

[7] Agostini, G., Longari, M., & Pollastri, E. (2003). Musical instrument timbres classification with spectral features. *EURASIP Journal on Advances in Signal Processing*, *2003*(1), 1-10.

[8] Al-Shoshan, A. I. (2006). Speech and music classification and separation: a review. *Journal of King Saud University-Engineering Sciences*, *19*(1), 95-132.

[9] Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, *10*(5), 293-302.

[10] Lu, L., Liu, D., & Zhang, H. J. (2005). Automatic mood detection and tracking of music audio signals. *IEEE Transactions on audio, speech, and language processing*, *14*(1), 5-18.

[11] Bergstra, J., Casagrande, N., Erhan, D., Eck, D., & Kégl, B. (2006). Aggregate features and a da b oost for music classification. *Machine learning*, *65*(2-3), 473-484.

[12] Li, T., Ogihara, M., & Li, Q. (2003, July). A comparative study on content-based music genre classification. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval* (pp. 282-289).

[13] Rabaoui, A., Davy, M., Rossignol, S., & Ellouze, N. (2008). Using one-class SVMs and wavelets for audio surveillance. *IEEE Transactions on information forensics and security*, *3*(4), 763-775.

[14] Hu, N., Dannenberg, R. B., & Tzanetakis, G. (2003, October). Polyphonic audio matching and alignment for music retrieval. In *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No. 03TH8684)* (pp. 185-188). IEEE.

[15] Krueger, A., & Haeb-Umbach, R. (2010). Model-based feature enhancement for reverberant speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, *18*(7), 1692-1707.

[16] Davis, S., & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, *28*(4), 357-366.

[17] Müller, M. (2007). *Information retrieval for music and motion* (Vol. 2, p. 59). Heidelberg: Springer.

[18] Korkmaz, Y., Boyacı, A., & Tuncer, T. (2019). Turkish vowel classification based on acoustical and decompositional features optimized by Genetic Algorithm. *Applied Acoustics*, *154*, 28-35.

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

[20] Kirthika, P., & Chattamvelli, R. (2012, July). A review of raga based music classification and music information retrieval (MIR). In *2012 IEEE International Conference on Engineering Education: Innovative Practices and Future Trends (AICERA)* (pp. 1-5). IEEE.

[21] Pandey, G., Mishra, C., & Ipe, P. (2003, December). TANSEN: A System for Automatic Raga Identification. In *IICAI* (pp. 1350-1363).

[22] Kumar, V., Pandya, H., & Jawahar, C. V. (2014, August). Identifying ragas in indian music. In *2014 22nd International Conference on Pattern Recognition* (pp. 767-772). IEEE.

[23] Salamon, J., Gulati, S., & Serra, X. (2012). A multipitch approach to tonic identification in indian classical music. In *Gouyon F, Herrera P, Martins LG, Müller M. ISMIR 2012: Proceedings of the 13th International Society for Music Information*

Retrieval Conference; 2012 Oct 8-12; Porto, Portugal. Porto: FEUP Ediçoes; 2012.. International Society for Music Information Retrieval (ISMIR).

[24] Shetty, S., & Achary, K. K. (2009). Raga mining of Indian music by extracting arohana-avarohana pattern. *International Journal of Recent Trends in Engineering*, *1*(1), 362.

[25] Joseph, R., & Vinod, S. (2017). Carnatic raga recognition. *Indian Journal of Science and Technology*, *10*(13), 0974-6846.

[26] Samsekai Manjabhat, S., Koolagudi, S. G., Rao, K. S., & Ramteke, P. B. (2017). Raga and tonic identification in carnatic music. *Journal of New Music Research*, *46*(3), 229-245.

[27] Shetty, S., & Achary, K. K. (2009). Raga mining of Indian music by extracting arohana-avarohana pattern. *International Journal of Recent Trends in Engineering*, *1*(1), 362.

[28] Degaonkar, V. N., & Kulkarni, A. V. (2018). Automatic raga identification in Indian classical music using the Convolutional Neural Network. *Journal of Engineering Technology*, *6*(2), 564-576.

[29] Ross, J. C., Mishra, A., Ganguli, K. K., Bhattacharyya, P., & Rao, P. (2017, October). Identifying Raga Similarity Through Embeddings Learned from Compositions' Notation. In *ISMIR* (pp. 515-522).

Appendices

Appendix A. Description of Implementation Repository

The entire project code can be found on the below GitHub repository

https://github.com/nupursjsu/Masters-Project