Table of Contents

[1 Introduction 1](#__RefHeading___Toc2094_1662942055)

[2 Data 1](#__RefHeading___Toc2096_1662942055)

[2.1 Alap Dataset 2](#__RefHeading___Toc2098_1662942055)

[2.2 Pakad Dataset 2](#__RefHeading___Toc2100_1662942055)

[3 Data Preprocessing 3](#__RefHeading___Toc2102_1662942055)

[3.1 Source separation and Pitch contour extraction 3](#__RefHeading___Toc2104_1662942055)

[3.2 Tonic extraction and Normalisation 3](#__RefHeading___Toc2106_1662942055)

[3.2.1 Tonic extraction 3](#__RefHeading___Toc2108_1662942055)

[3.2.2 Pitch Contour Normalisation 4](#__RefHeading___Toc2110_1662942055)

[3.3 Sequence Extraction 4](#__RefHeading___Toc2112_1662942055)

[4 Train-test splits 5](#__RefHeading___Toc2114_1662942055)

[4.1 Easy split – 1 [split 1 – singer separate] 5](#__RefHeading___Toc2116_1662942055)

[4.1.1 Easy split – 1 - AG 5](#__RefHeading___Toc2118_1662942055)

[4.1.2 Easy split – 1 – CC 6](#__RefHeading___Toc2120_1662942055)

[4.1.3 Easy split – 1 – SCh 6](#__RefHeading___Toc2122_1662942055)

[4.2 Easy split – 5 [split 5 – random solo] [TO BE UPDATED] 7](#__RefHeading___Toc2124_1662942055)

[4.3 Hard split [split 2 – unseen singer] [TO BE UPDATED] 7](#__RefHeading___Toc2126_1662942055)

[4.3.1 AG split 7](#__RefHeading___Toc2128_1662942055)

[4.3.2 CC split 8](#__RefHeading___Toc2130_1662942055)

[4.3.3 SCh split 8](#__RefHeading___Toc2132_1662942055)

[5 Literature I have gone through 9](#__RefHeading___Toc2134_1662942055)

# Introduction

This project approaches the problem of raga detection as a time series classification problem using a CNN architecture.

A raga can be defined as a melodic form in Indian classical music consisting of particular scale notes, having identifiable melodic shapes and with an associated mood [Sandeep Bagchee]. Most ragas have a characteristic phrase which frequently occurs, referred to as pakad or catch phrase. By it’s repetitiveness it makes the uniqueness of the raga evident and aids in raga recognition [Sandeep Bagchee – Chapter 2]. The idea here is to extract phrases and its associated characteristics from raga performance and use its uniqueness to identify raga. In order to do this we have used time-series classification using CNN architectures.

Phrases can be described with time series data, depicting fluctuating such as pitch and energy at fixed time intervals. Hence, this becomes a time series classification problem. A similar problem is Human Activity Recognition (HAR). Sensor based HAR, uses time series motion data from sensors such as accelerometers, gyroscope etc. To predict the activity being performed by the human being. Convolutional Neural Networks (CNN) have proven to perform very well on HAR tasks because of 2 factors: local dependency and scale invariance [Deep learning for sensor based activity – a survey].

# Data

The collected audio clips were performed by 3 professional musicians – Apoorva Gokhale (AG), Chiranjeeb Chakraborty (CC) and Sudokshina Chatterjee (SCh).

9 Ragas included in this data set are - Bageshree, Bahar, Bilaskhani Todi, Jaunpuri, Kedar, Marwa, Miyan ki Malhar, Nand and Shree.

2 sets of data have been used – Alap dataset and the Pakad dataset. Both datasets have videos recorded in 25 fps, and 48 kHz stereo with 128 kbps bit rate audio. All files are available in mp4 format.

## Alap Dataset

The singers here were asked to sing unaccompanied alap for 3 minutes. The duration of the videos range from 165-221 seconds, with a median duration of 187 s. In most cases 2 takes of each raga alap were recorded. There are a total of 55 aalaps. Below is a distribution of the alap raga-wise and singer-wise:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Raga/Singer | SCh | CC | AG | Sum |
| Bageshree | 2 | 2 | 2 | 6 |
| Marwa | 2 | 2 | 2 | 6 |
| Bahar | 2 | 2 | 2 | 6 |
| Kedar | 3 | 2 | 1 | 6 |
| Shree | 2 | 2 | 2 | 6 |
| Nand | 2 | 2 | 2 | 6 |
| Miyan ki Malhar | 3 | 2 | 2 | 7 |
| Jaunpuri | 2 | 2 | 2 | 6 |
| Bilaskhani Todi | 2 | 2 | 2 | 6 |
| **Sum** | **20** | **18** | **17** | **55** |

Table 1: Distribution of Alap Files among singers and ragas

## Pakad Dataset

These are clips of ‘catch’ phrases (pakad) of the raag. The duration ranges from 9-96 seconds, with a median duration of 19 s and mean duration of 30.1 s. There are 37 such clips (27 are reported in the pose estimation paper, not sure why). Below is a distribution of the pakad raga-wise and singer-wise.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Raga/Singer | SCh | CC | AG | Sum |
| Bageshree | 2 (1) | 1 | 1 | 4 (3) |
| Marwa | 2 | 1 | 1 | 4 |
| Bahar | 2 (1) | 1 | 1 (0) | 4 (2) |
| Kedar | 2 | 1 | 1 | 4 |
| Shree | 2 | 1 | 1 (0) | 4 (3) |
| Nand | 2 (1) | 1 | 1 (0) | 4 (2) |
| Miyan ki Malhar | 2 | 1 | 1 | 4 |
| Jaunpuri | 2 | 1 | 1 | 4 |
| Bilaskhani Todi | 2 | 2 | 1 | 5 |
| **Sum** | **18 (15)** | **10** | **9 (6)** | **37 (31)** |

Table 2: Distribution of Pakad songs among singers and raga. The number in the brackets depict the number of files actually used in the rest of this project. Some of these songs were discarded because they were less than 12 s.

For all experiments explained in the later sections, both the Alap and Pakad datasets were used together unless otherwise mentioned giving rise to a dataset comprising of 86 songs sung by 3 singers and comprising of 9 different ragas.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Raga/Singer | SCh | CC | AG | Sum |
| Bageshree | 3 | 3 | 3 | 9 |
| Marwa | 4 | 3 | 3 | 10 |
| Bahar | 3 | 3 | 2 | 8 |
| Kedar | 5 | 3 | 2 | 10 |
| Shree | 4 | 3 | 2 | 9 |
| Nand | 3 | 3 | 2 | 8 |
| Miyan ki Malhar | 5 | 3 | 3 | 11 |
| Jaunpuri | 4 | 3 | 3 | 10 |
| Bilaskhani Todi | 4 | 4 | 3 | 11 |
| **Sum** | **35** | **28** | **23** | **86** |

Table 3: Distribution of songs used (excluding discarded songs from Table 2) from both datasets over singers and ragas.

# Data Preprocessing

All audio files were converted into multiple 12 second sequences of normalised pitch values. The steps involved include – Source separation and Pitch contour extraction, tonic extraction and normalisation and sequence extraction.

## Source separation and Pitch contour extraction

The vocal audio was separated from the given audio using [Spleeter’s 4 stem source separation model](https://github.com/deezer/spleeter). This audio was then fed into the [Praat](https://uvafon.hum.uva.nl/praat/) API in python –[Parselmouth](https://github.com/YannickJadoul/Parselmouth) to extract the pitch contour of the song. The range of pitches extracted were restricted to the lower octave fifth (mandra sapthak Pa) and the higher octave fifth (taar sapthak Pa). The pitch values were extracted for every 10 ms.

## Tonic extraction and Normalisation

For the pitch representation to be musically relevant, we convert the pitch values from from Hz to Cents. For this conversion, the tonic frequency is considered as the reference frequency (corresponds to 0 cents). Thus the representation we use is independent of the tonic frequency of the song [Gulati et. al. - Mining Melodic Patterns in Large Audio Collections of Indian Art Music].

### Tonic extraction

Tonic was extracted using [essentia](https://essentia.upf.edu/) library’s TonicIndianArtMusic function. All parameters were set to their default values except for maxTonicFrequency which was = 530. This value was determined from my experiments with the JJ dataset. I found that increasing the maxTonicFrequency allowed for the tonics of female singers to be detected more accurately.

For each singer, the singer’s tonic note (such as G3, G#3 etc.) was determined by ear for all the songs. The tonic values extracted by the algorithm (in Hz) were compared to the frequency of the tonic notes determined earlier. The highest occuring correct frequency (determined by minimising the difference between tonic value extracted from the algorithm and frequency value corresponding to the tonic note determined by ear) throughout the dataset for each singer was set as the tonic for all songs by that singer.

|  |  |  |
| --- | --- | --- |
| Singer | Note | Tonic (Hz) |
| CC | C#3 | 138.81 |
| AG | G#3 | 207.88 |
| SCh | A#3, B3 | 234.89, 247.28 |

Table 4: Tonic values determined for each singer; SCh has sung in two tonics in the dataset

### Pitch Contour Extraction and Processing

#### Extraction and Tonic Normalisation

All pitch contours were then normalised with respect to their respective tonic with 100 divisions per semitone.

#### Interpolation

All continuous segments of unvoiced frames for duration < 500 ms, were interpolated with cubic spline interpolation. To calculate the interpolation function, all voiced frames from 0.1 s (10 frames) before the unvoiced segment and all voiced frames 0.1 s (10 frames) after the unvoiced segment were considered. In cases where there were not enough voiced frames around the unvoiced segment, the values were left as they were.

#### Unvoiced Frame Values

The range of extracted values are approximately [-500, 1700] after normalisation. This is because these values correspond to range of frequencies extracted – lower octave Pa and higher octave Pa. I use the word approximately because due to difference in intonation of the note, the cents value of the lower Pa might be a few cents lower than -500 and similarly the cents value of the higher Pa might be a few cents higher than 1700. Hence unvoiced frames are replaced with the value -550 so that the value is closer the range of values extracted for the song.

#### Data Preprocessing

Following this, 2 versions of each pitch contour a standardized version and a normalized version as described below.

##### Pitch contour with standardized Data

Each pitch contour is standardized to have a mean of 0 and a standard deviation of 1[[1]](#footnote-2). This is similar to the preprocessing followed in [Zheng et. Al.](https://www.researchgate.net/publication/303250835_Time_series_classification_using_multi-channels_deep_convolutional_neural_networks)

Standardization for each pitch contour. X – pitch contour values; u – mean of values in the pitch contour; s – standard deviation in pitch contour

##### ***Pitch contour with normalized data***

Each pitch contour is scaled to values between [0, 1][[2]](#footnote-3). This implies that the minimum value (unvoiced frame value) is set as 0 across all songs. This is not the case with standardization since the mean and standard deviation varies from contour to contour.

Normaliz*ation of* the pitch contours. x - pitch contour

## Sequence Extraction

Given that each video is 25 fps, according to Jin’s description in the email thread: there are sequences of duration 12 s extracted, a stride of 40 frames (40/25=1.6s) between each sequence and a random number (-20, 20) ((-20/25, 20/25) = (-0.8 s, 0.8s)) added to the sequence starting point before its extraction.

Hence, 12 s (=1200 audio frames, since we extracted pitch values at 0.01s intervals) sequences were extracted from the tonic normalised pitch contours of all songs in the dataset. Following is a description of the steps followed to extract sequences from a given song.

1. A random number between (0, 80) is selected. This will indicate the starting index of the audio frame to begin with for the sequence. This is because 80 audio frames corresponds to 0.8s.
2. 1200 audio frames from this start index are extracted and stored as a sequence.
3. 160 (corresponding to 1.6 s) is added to the current starting index, to account for the stride discussed in Jin’s correspondance. To this new start index a random integer between (-80, 80) is added corresponding to the random number added before each sequence extraction.
4. Steps 2 and 3 are repeated until we reach the end of the pitch contour

(There was no minimum threshold imposed on the fraction of voiced frames in a sequence, as I had done in the previous experiments. This was done to simulate the procedure Jin followed as much as possible)

Songs less than 12 seconds were discarded. Since we are normalizing each contour before splitting it into sequences, padding the sequence with a fixed constant value (other than 0) seemed incorrect. Since Jin also followed this method of discarding songs less than 12 s, I decided to stick with this for now. We can change this later if required.

There were 6568 sequences extracted using this method (6133 sequences extracted from 55 Alap songs and 429 sequences from 31 Pakad songs).

Sanity check for the number of sequences extracted:

1. Alap dataset contains 55 songs of with a mean duration of 190s. Since the average stride between all sequences is 1.6 seconds, there should be (190-12)/1.6 = 111.25 number of sequences per song. Hence a total of approximately 6119 (=55 \* 111.25) sequences from the dataset which is close to the 6133 sequences actually extracted from this set.

2. Pakad dataset contains 37 songs with a mean duration for 30s out of which only 31 are used. Since the average stride between all sequences is 1.6 s, there should be (30-12)/1.6 = 11.25 sequences per song. Hence a total of approximately 416 (=11.25\*37) sequences should be extracted. This estimate is close to the 429 sequences extracted.

|  |  |  |
| --- | --- | --- |
| Raag | Number of Sequences | Average Fraction of Voiced Frames |
| Bag | 720 | 0.74 |
| Bahar | 678 | 0.83 |
| Bilas | 702 | 0.75 |
| Jaun | 698 | 0.77 |
| Kedar | 709 | 0.72 |
| MM | 845 | 0.71 |
| Marwa | 740 | 0.74 |
| Nand | 710 | 0.76 |
| Shree | 760 | 0.72 |
| **Sum** | **6562** |  |

Table 5: Statistics of sequences extracted

The task and data splits

Using the sequences generated from the previous section, the aim is now to be able to identify the raga that each sequence belongs to. For this, we have utilised 2 types of train-test splits:

# Train-test splits

## Easy split – 1 [split 1 – singer separate]

There are 3 types of splits for this scenario – one for each singer. For a given singer, say AG, the second take for each raga in the Alap dataset is put in the test set (filenames are of the form AG\_\*b\_\*.mp4). In the case where there is only one take for a raga (AG has only one take of Kedar) that particular take is placed in the test set. All the other songs in the Alap and Pakad dataset by all three singers are put in the train test. This results in 46+31=77 songs in the train set and 9 songs in the test set for each singer.

### Easy split – 1 - AG

In this split, the second take of each raga by AG is placed in the test set. Raga Kedar has only one take and hence is placed in the test set. All other songs by all singers in the Alap and Pakad datasets are placed in the train set.

|  |  |  |
| --- | --- | --- |
| Raag | Number of sequences in train | Number of sequences in test |
| Bag | 616 | 104 |
| Bahar | 571 | 107 |
| Bilas | 597 | 105 |
| Jaun | 588 | 110 |
| Kedar | 599 | 110 |
| MM | 738 | 107 |
| Marwa | 633 | 107 |
| Nand | 596 | 114 |
| Shree | 652 | 108 |
| **Sum** | **5590** | **972** |

Table 6: Ragawise distribution of data in easy\_split\_1-AG

### **Easy split – 1 – CC**

In this split, the second take of each raga in the Alap dataset by CC are put in the test set. All the other songs from Alap and Pakad dataset, by all singers is kept in the train set.

|  |  |  |
| --- | --- | --- |
| Raag | Number of sequences in train | Number of sequences in test |
| Bag | 605 | 115 |
| Bahar | 553 | 125 |
| Bilas | 586 | 116 |
| Jaun | 584 | 114 |
| Kedar | 596 | 113 |
| MM | 715 | 130 |
| Marwa | 623 | 117 |
| Nand | 601 | 109 |
| Shree | 624 | 136 |
| **Sum** | **5487** | **1075** |

Table 7: Ragawise distribution of data in easy\_split\_1-CC

### Easy split – 1 – SCh

In this split, the second take of each raga in the Alap dataset by SCh are put in the test set. All the other songs from Alap and Pakad dataset, by all singers is kept in the train set.

|  |  |  |
| --- | --- | --- |
| Raag | Number of sequences in train | Number of sequences in test |
| Bag | 603 | 117 |
| Bahar | 570 | 108 |
| Bilas | 594 | 108 |
| Jaun | 596 | 102 |
| Kedar | 601 | 108 |
| MM | 741 | 104 |
| Marwa | 625 | 115 |
| Nand | 608 | 102 |
| Shree | 650 | 110 |
| **Sum** | **5588** | **974** |

Table 8: Ragawise distribution of data in easy\_split\_1-SCh

## Easy split – 5 [split 5 – random solo] [TO BE UPDATED]

For this we have listed one file from each raga, split across the 3 singers for the test set. All the sequences extracted from these 9 files will constitute to the test set whereas the sequences extracted from all the other 83 (46 alap + 37 pakad) files in the dataset will make the train set.

List of the 9 files in the test data:

|  |  |  |
| --- | --- | --- |
| Filename | Singer | Raga |
| AG\_1b\_Jaun | AG | Jaun |
| AG\_2a\_Marwa | AG | Marwa |
| Sch\_2b\_MM | Sch | MM |
| CC\_8a\_Bag | CC | Bag |
| Sch\_6a\_Kedar | Sch | Kedar |
| Sch\_4b\_Nand | Sch | Nand |
| CC\_5a\_Shree | CC | Shree |
| CC\_1b\_Bilas | CC | Bilas |
| AG\_7a\_Bahar | AG | Bahar |

Table 9: List of test files for easy split

There 5578 sequences in the train set and 900 sequences in test set.

|  |  |  |
| --- | --- | --- |
| Raag | Number of sequences in train | Number of sequences in test |
| Bag | 606 | 115 |
| Bahar | 575 | 105 |
| Bilas | 586 | 116 |
| Jaun | 588 | 110 |
| Kedar | 604 | 105 |
| MM | 741 | 104 |
| Marwa | 631 | 109 |
| Nand | 310 | 102 |
| Shree | 637 | 124 |
| **Sum** | **5578** | **900** |

Table 10: Ragawise distribution of data in the easy split

## Hard split [split 2 – unseen singer] [TO BE UPDATED]

This split tries to test if the model can learn, independent of the singer. Hence, all the songs from a given singer, say AG, are kept in the test set while the songs from the other 2 singers are kept in the train set. Since there are 3 singers, there are 3 such splits that can be obtained – each split containing one of the singers in the test set.

### AG split

In this split all sequences generated from AG’s songs are kept in the test set. This results 4718 train sequences and 1850 test sequences.

|  |  |  |
| --- | --- | --- |
| Raag | Number of sequences in train | Number of sequences in test |
| Bag | 507 | 214 |
| Bahar | 467 | 213 |
| Bilas | 482 | 220 |
| Jaun | 481 | 217 |
| Kedar | 598 | 111 |
| MM | 628 | 217 |
| Marwa | 523 | 217 |
| Nand | 488 | 224 |
| Shree | 544 | 217 |
| **Sum** | **4718** | **1850** |

Table 11: Ragawise distribution of sequences in the AG hard split

### CC split

In this split all sequences generated from CC’s songs are kept in the test set. This results in 4111 train sequences and 2457 test sequences.

|  |  |  |
| --- | --- | --- |
| Raag | Number of sequences in train | Number of sequences in test |
| Bag | 438 | 283 |
| Bahar | 420 | 260 |
| Bilas | 445 | 257 |
| Jaun | 434 | 264 |
| Kedar | 450 | 259 |
| MM | 550 | 295 |
| Marwa | 467 | 273 |
| Nand | 447 | 265 |
| Shree | 460 | 301 |
| **Sum** | **4111** | **2457** |

Table 12: Ragawise distribution of sequences in the CC hard split

### SCh split

In this split all sequences generated from Sch’s songs are kept in the test set. This results in 4307 train sequences and 2261 test sequences

|  |  |  |
| --- | --- | --- |
| Raag | Number of sequences in train | Number of sequences in test |
| Bag | 497 | 224 |
| Bahar | 493 | 207 |
| Bilas | 477 | 225 |
| Jaun | 481 | 217 |
| Kedar | 370 | 339 |
| MM | 512 | 333 |
| Marwa | 490 | 250 |
| Nand | 489 | 223 |
| Shree | 518 | 243 |
| **Sum** | **4307** | **2261** |

Table 13: Ragawise distribution of sequences in SCh hard split

# Literature I have gone through

1. Human activity recognition with smartphone sensors using deep learning neural networks – Roanao C. A., Cho S.  
   They use CNNs to tackle the proble of HAR because human activities are hierarchical in nature (multiple layers of CNN takes care of this), translation invariant (Taken care of by the pooling operator) and temporally correlated (taken care of by convolutional operator).  
   They do a greedy wise tuning of hyperparameters in the following order – number of layers, feature maps, filter size, pooling size. They also experiment with learning rate. However, the base model that they started with has not been clearly defined. The data they use contains a 6 channel sequences of length 128 (data sampled at a rate of 50 Hz, hence this corresponds to 2.56 seconds). There were 6 different activities and data collected from 30 different people.   
   They were able to achieve 94.79% accuracy with raw sensor data. They achieved almost perfect classification of ‘walking down’ and ‘walking up’ which was percieved as difficult using pre-existing methods.
2. Deep Convolutional Neural Networks On Multichannel Time Series For Human Activity Recognition – Yang J. B. et al.  
   This paper uses CNNs for HAR since HAR tasks are hierarchical in nature, i.e. they are made of smaller actions. HAR signals are different from image or speech because, most of the signal belongs to the null class (activities occur during a small fraction of the signal) and one activity consists of smaller continuous movements. This paper proposes a method to process the data along the temporal dimension and combine the units in the CNN from each sensor (which is first processed individually) to obtain a final prediction.  
   2 datasets were used: Opportunity Activity Recognition Dataset (Input dimensions - 113x30, i.e. 30 time steps from 113 sensor; 18 output classes) and Hand gesture dataset (Input dimensions 15x32, i.e. 32 time steps from 15 sensors with 12 classes).  
   Their CNN model performed better than baseline models described in the paper along all metrics. They use a smoothing technique on the predictions, which increases the accuracy on the CNN and keeps the rank of performance among models intact.  
   Main contributions of this paper  
   1. Features are task-dependent and non hand-crafted  
   2. Features thus have more discriminative power for HAR task  
   3. Feature extraction and classification are performed by the same model so performance of both are mutually enhanced.

1. This was done with scikit-learn's StandardScaler module (https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html) [↑](#footnote-ref-2)
2. This was done with scikit-learn’s MinMaxScaler module

   (https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html) [↑](#footnote-ref-3)