Identification of Songs using Humming Query

A Dissertation Submitted for the Degree of Master of Computer Science

I. A. Abeysekera

University of Colombo School of Computing

2021

DECLARATION

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis. This thesis has also not been submitted for any degree in any university previously.

Student Name: I. A. Abeysekera

Registration Number: 2018mcs001

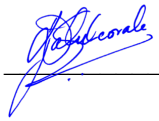
Index Number:18440012

Signature of the Student & Date

This is to certify that this thesis is based on the work of Mr. Ishara Avinda Abeysekera under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by,

Supervisor Name:



29/11/2021

Signature of the Supervisor & Date

I would like to dedicate this thesis to

my beloved parents,

my wife,

who encouraged me to be the man who I am today,

&

academic and non-academic staff

of

University of Colombo School of Computing,

who educated me and enabled me

to reach at this level.

ACKNOWLEDGEMENTS

I am grateful to my supervisor Dr. Ajantha Atukorale of the University of Colombo, School of Computing for his valuable support to make my dissertation and the research project a success. Even with the pandemic situation Dr. Ajantha always had time to conduct over the phone discussion whenever I was in a trouble or need hand in areas when had a question about my research or writing. He consistently allowed this paper to be my own work, but directed me in the right the direction whenever he thought I needed it. His great guidance, excellent dissertation ideas and confidence pointed me to come up with a successful research project. Finally, I must direct my deep gratitude towards my parents for providing me with consistent support and continuous encouragement throughout years of study and through the completion of researching and writing this thesis. This achievement would not have been conceivable without them. Also, I thank all those whose names, though not mentioned for their support and encouragement in completing this project. Thank you.

ABSTRACT

Research focused on ‘pure approach’ such as ‘sound engineering’ to query humming part of sound against music or song to recognize similarities using distance variance. With the increase number of music files on different types of devices it’s very difficult to locate a music file in instant. Thus, audio file content is not visually understandable by humans without any expert help. In this study it has been conducted research on how to the reading of this human understandable streams of music to match with respective hummed input. As for the proposed method in this study it uses query by humming model that can predict song for user humming. In this model it has conducted in many three areas. Which is convert humming wave to frequency vector using ACF based approach. Then further quantify these frequency vectors using hop windows for better frequency wave based on beat of the song. This frequency vectors related average filtering is also discussed in the study where generalized frequency vectors into more informative beat frames-based frequency vectors. Secondly conversion of Frequency vector to Note vector has been conducted. These notes have been gone through several filtering mechanisms and has used note difference calculation to provides note difference vectors which will be used in third step where measure DTW difference calculation to match Hummed queries and real songs. Further study has conducted research on Quantified Dynamic Time Wrapping (QDTW) approach to carry out local minimum distance analysis in manual approach. By conducting research on these areas, it has proven that sound engineering approach can give accurate results when identifying songs for hummed queries over 83% overall accuracy for relatively better pitched Hummings respect to 67% accuracy for selected bad set of Humming data. Since research carried out mainly on identifying Sinhala songs humming data used were created by school of professional singers at perfect pitch, and random pitch, which has been evaluated separately to distinguish performance of the model over giving results ranking at top 5 results scale.

***Keywords**:* Pitch, Note, Melody, Band, humming query, Hop Frequency, Windowing

TABLE OF CONTENTS

List of Tables

List of Figures

List of Equations

1. INTRODUCTION

1.1. Motivation

Text search is something very common and everybody knows with what Ctrl + F do. However, search for audio file without knowing any meta information would be a huge time taking task. What if this audio file is a song or a music? A small piece of melody rhyme on our head but difficult search by playing one by one song at a time. What if a person can find a song without being worried too much about song/melody that echoes in the head?

And secondly for identify copyrights issues prior to composing a new melody without any conflicts by song/ music artists, these artist needs to acknowledge whether they are violating any copyrights intentionally or unintentionally. Thus, it requires consumers to know what they are doing with additional support. Sometimes This artist requires to find these melodies where it actually originates from. So, its again back to our problem 1 (Finding melody without meta data).

There are multiple occasions in which, identifying a song could be crucial for some people. Such occasions can be, when we hear some piece of melody in a TV program, movie, Drama, etc. Since the domain of the study is related to the entertainment industry, which is currently a highly emerging industry in the world, and thus music industry itself is a huge market already (e.g., apple music, Spotify, etc.). It can state that many people (music directly related and directly non-related) will benefit from using this application and study towards individual use to more upcoming future developments.

1.2. Statement of the Problem

With the increase number of various formats music files, people find it difficult to locate music files with a simple effort. Especially when finding a song without knowing the name or the artist it's very difficult with manual approach. And also, song search by audio input and music copyright validation, is still at primitive stage of development with lack of accuracy and reliable compared to other text-based search and validation platforms.

Many people who listen to music or songs sometimes get some musical melody on their heads. But unable to find the song or music with which that melody occurred. Then will be frustrated in searching for the real song or music where that piece of echo in the head occurred. Even with

existing systems that can recognize a very narrowed selection of songs in the domain which may not always give accurate results or no correct result at all.

In the meantime, some melody creators require new compositions that require them to identify not copying other songs which may later need to deal with copyright issues.

Hummed melodies by users are not accurate compared to a real music piece. And the challenge in this study is to accurately map hummed melody to existing music piece or song or get the closest match as much as accurate as possible within a considerable amount of time. Even with a song hummed accurately by a professional singer still, it is hard to match against the original song since there are so many other instruments, voices, and noises attached to that. So, comparing two completely different sources needs huge preprocessing and data transformation into comparable sources.

1.3. Research Aims and Objectives

Focus of this research to recognize human hummed tunes/ melodies important features, so it can be used to create a model which can be then predict top most matches for a given hum query. used recognized as a matching set of songs or music.

1.3.1. **Aim**

Make human hummed tunes/ melodies to be recognized as a matching set of songs or music based on a predefined database of songs with improved accuracy over existing systems.

1.3.2. **Objectives**

- Capture the hummed melodies and their most powerful features using STFT (Short-time Fourier Transform) windowing and related techniques.
- Achieve higher accuracy and speed for retrieval of humming melodies matching with database of songs/music.
- Create a Web UI for users to input humming recordings (upload file or live record) and view the results.
- Optional objective to achieve match for a shortest duration hummed input.

1.4. Scope

- Datasets consist only **Sinhala songs** to narrow down the search scope. With short clipped data filed ranging from 10-15s length to minimize the processing time for the study.
- Identification scope for humming tune is limited to search hit among top 5 of songs since the difficulty of distinct a small variation from one humming to another humming. Which one humming could identify more feature similar to closely feature spread song. Such as strong instrumental songs vs more strong vocal songs.
- Analysis of the Humming system on a narrowed scope with **good quality inputs**, vs **bad quality** inputs to compare how well it recognize song as it is and not recognize when it's not well hummed.
- Further study will not consider genre specific categorization as in other approaches as that will be adding new layer or dimension to the study making it much broader area to be studied.
- Also, other areas will not be covered in this study are noisy, distorted inputs and all other genre base categorizations.
- Study will be within the boundary of a selected range of music, specifically identified songs in different ranges such as rhythmic based, regular to high tempo based, strong music-based song vs strong vocal based, etc.
- Output of the study will be ranked identified set of songs based on DTW distance score (less the score better it is). And categorize more practical scale where 1st five songs of the selection have more relevance to the correct guess rather finding exact 1st song due to the fine-tuning limitations.
- Since this project is mainly regarding improving accuracies of identified songs, in each stage micro evaluations has been carried out, such as **'hit'/'miss' ratio** in each stage of development.

2. LITERATURE REVIEW

2.1. A Literature Review

Several studies had been conducted on music retrieval carried out on different aspects on similar outcomes on the identification of music using several querying techniques and related work. One of the many uses could be to avoid an inadvertent breach of copyright, by using such a system.

A study by A. K. Tripathy et al. on Query by Humming [1] System has recognized an approach to transform hum to match songs. At a glance, the approach of this study is to try to detect pitches in sung melody and compare them against a symbolic representation of the known melodies. Then it tries to rank similar sung pitches against known melodies. This system accepts hummed input and query experiments carried with 22.05KHz WAV format. With the help of public domain MIDI songs each hummed tune will be queried and result in a ranked list of matching melodies. However, the system is restricted to humming with 'ta' syllables for users and is not considered musical key or tempo. Where the stop consonants cause the local energy of the waveform to dip, where it's easier to identify note boundaries. This study has focused more on how WAV to MIDI conversion so that it will not lose the most important data while this conversion. Pitch determination was done during the conversion of WAV to MIDI along with the velocity and duration of each note being played. User Input is classified in one of three ways: note is same as a previous note (S), higher than a previous note (U), lower than previous (D). So, the alphabet of {S, U, D} is used to represent notes. Pattern matching is done on U, D, S strings based on the edit distance of two strings, where the number of point mutations required to change string from s1 to s2 string. (point mutation is: 1. Change a letter, 2. Insert a letter, or 3. Delete a letter).

$$d_{ij} = \min \begin{cases} d_{i-1,j} + w(a_i, 0) & \text{(deletion)} \\ d_{i-1,j-1} + w(a_i, b_j) & \text{(match/change)} \\ d_{i,j-1} + w(0, b_j) & \text{(insertion)} \end{cases}$$

- Where w(ai,0) is the weight associated with **deletion** of ai.
- w(0,bj) is the weight for **insertion** of bj.
- w(ai, bj) is the weight of **replacement** element i in sequence A by element j in the sequence B

For an example, in the sound scale “do re me fa so la te do”, the string pattern of “*do re me me so re*” can be expressed as “U U S U D”. After this query will be matched against a database.

Fundamental attributes that were used to distinguish one melody from another are notes, rhythm, tempo, dynamics, and lyrics. And the study found out melody and rhythm are most distinctive. The melody consists of a sequence of notes with varying pitch duration, where the pitch has given the highest distinctive nature. The study was done in 2 main areas: i) Melodies hummed clearly, without distortions, ii) melodies hummed falsely. The study states under ideal situations it achieves closer to 100% accuracy which may due to an overfitting scenario. The significance of the study is the approaches they have used to detect feature, pitch using time-domain autocorrelation which was originally used in another study called Tansen. Where pitch periods are combined to non-overlapping frames as per melody of pieces of music which involves notes with varying pitches. This can be useful in current research to extract the pitch of a hummed midi. However, doing searches with a database can take so long time for so many songs in a database.

A similar study has been carried out by P. Patel called Music Retrieval System Using Query-by-Humming [2] Has proposed a system with the help of Dynamic Time Warping (DTW) to achieve faster results and less false positive rate. In his study, He has identified how to store music data with less storage using SQLite and MIDI format conversion of queries. Music retrieval was performed DTW using feature vectors to find similarities between the hummed query and song features(pitch) stored in the Database. The result is an individual output rather than a list of ranked songs. Evaluation is done mainly using Mean Reciprocal Rank (MRR) index. The study states that MRR is at 0.82 for 5 songs in the database and 0.90 for 100 songs in the database for his QBH system. Apart from identifying songs, the system further carried evaluation based on gender-hummed queries which output MRR 0.84 and 0.80 for Male and Females respectively. Further Identified Professional and Non-Professional humming for MRR rates as 0.90 and 0.72 respectively. Performance evaluated at 77153 songs in the database average retrieval time 24400ms. This is another study stating regarding DTW used for fast retrieval of matches. Thus DTW will be used as one of the main matching components. DTW has been applied to temporal sequences of video, audio, etc. any data that can be turned into linear sequences.

R.A. Putri Has conducted a study MIR using QBH [3] based on DTW for narrowed Indonesian songs context. Which has carried Pitch tracking to convert voice signal to sound object then

convert with a table of frequencies for certain pitches. The study has identified the common difficulty of the human voice maintaining constant pitch as a musical instrument (e.g., piano) which affects performance. Has done refinement of queries by setting threshold length of one note. Searching was performed based on DTW and results showed in a sorted list based on distance cost between query and template. When the query has different keys with the template study states the DTW method is not suitable with experimental results for MRR goes 0.09 for 100 songs database. Using a query with the same key as in the template MRR results have 0.92 for 100 songs database. From this case, it can derive a transcription module where humming input transcribed into a list of semitones architecture. However, this approach was limited to Indonesian song context some feature selections and with its high precision rate, it can assume the system may have overfitted to context.

Multi-classifier-based approach by G Nam et al. has given broader symmetric search approach to get more combined accurate results [14]. In this study they have used Fourier Descriptor to transform humming wave to frequency domain. And has used Quantized DTW and Quantized Linear Scaling as classifiers to increase the accuracies. Further they have combined these 3 with conventional DTW and LS which to enhance performance. With these 5 classifiers they have performed multi-classifier approach in symmetric space.

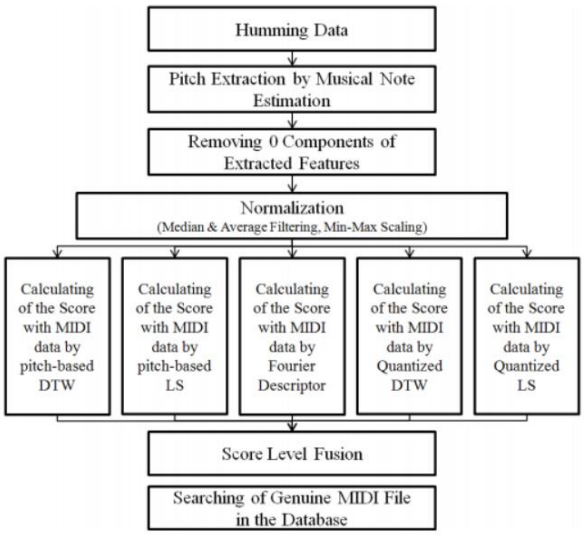


Figure 1: Multi-Classifier-based approach [14]

SoundCompass [4] is developed by a Japanese laboratory application run on Karaoke situations where users can request songs by singing a part of a song. N Kosugi et al. worked on this study

with 21,804 songs which are also in the MIDI format. In this study, songs are segmented holding the same amount of music information which are required for subsequence matching. This process of segmentation is done in two parts, 1st one temporal split which defines the segments and 2nd split is in which every segment has the same music information. Which will be used to create subsubsequences - VSTPV (Variable Slide-length Tone In Phonetic Value rate) from a subsequence. Sum up using an algorithm that takes every melody track based on a time unit, they have defined a length and overlap of segments which are called subsequences - CSTPV (Constant Slide-length Tone-In Phonetic Value).

Hummed Tune processing is done at tunes recorded at 11KHz/8bit resolution/monoral. The pitch detection algorithm is processed with 512 points data for every 256 points (which corresponds to analyzing data every 0.025 seconds with 0.05 seconds of lengthy data each time). Pitch and utterance are determined based on assumption that an utterance corresponds to one note (this was possible most of the melodies have lyrics with a single syllable).

The matching strategy of this study uses similarity distance (city-block distance measurement) between one song of the database and query the minimal distance between sub/subsubsequences.

The following equation has been used to calculate distance:

$$D(h, s) = \min \{ d(h_i, s_j) \}$$

Equation 1: Distance Calculation

Where D is the measure between h (humming) and s(song).

The Paper states that the Retrieval accuracy of the system is around 78% for the first 25 given answers with City block distance measurement used for linear matching, music data were normalized. Around 62% when time normalized with Dynamic Time Warping. And 22% use Dynamic Time Warping alone. Mainly this study will be used as a guideline to work over data sets and another source of view over DTW.

In a very recent (in October 2020) application, Google has introduced their song matching via humming, whistling, or singing [9]. This system is powered by machine learning algorithms built in-house by Google. It doesn't need to sing at the correct pitch but has accommodated for various

degrees of musical sensibility. This system provides a confidence score in a percentage with several possible matches as an output. The approach of the application is to match hum, whistle, or singing to a respective song's unique number which is called fingerprint using machine learning algorithms. In this approach, they have conducted series of filtering such as taking away accompanying instruments, voice's timbre, and tone. What is left is the song's number based-sequence (fingerprint). However, this project is still at its early stages and yet to develop the accuracy of capturing a perfect match. Another limitation is space to store all the fingerprint datasets thus they were continuously removing the least search trends and update popular ones. There is no paper written for this application, however, the abstract application method can be used to compare against its effectiveness towards the current research study. Difference between this approach is it cannot directly search for a song or audio track such as 'Pirith' or 'Shloka', without separately including them as fingerprints in their module. However in our approach once created model with parameters it has good chance of identifying such audio file with vocals similar to any song.

Other mentions: A.Li-Chun's "Shazam"[5] is a commercial application currently available which allows users to identify songs using the Query by example(QBE) approach. Which only identifies the background music of the original song. They use a method called Audio Fingerprinting to identify or search songs in a database and quite fast and only require a portion of the song to recognize a song. This can retrieve a song between 5-500ms in a database of 20k songs. Look up time of this has enhanced to O(1). However fingerprinting approach doesn't allow any variations to be identified such as humming or even singing the song by its artist perfectly will not be supported. Even though Shazam's fingerprinting approach may directly not support hummed queries, analysis can be work towards finding the most matching fingerprint similar to Google's study and then evaluate that fingerprint over accuracies.

3. METHODOLOGY

This study will cover three areas, which will be hummed melodies to convert into several comparable sources. Thus, this will provide a huge contribution to upcoming researches to identify methodologies that can be used to data preparation related techniques. Secondly, this study has been carried on Music engineering based with DTW distance comparison approach. to find the best pattern recognition towards ranking songs based on their feature vectors according to the approach. Finally, the study will contribute to storing formatted songs on flat file structure in support of advanced file managing mechanisms to increase the speed of match retrieval.

The study will be based on English song datasets at first, therefore only English songs will be given preference and later stages will fine-tune to match Sinhala songs with a limited genre of data set in the identification of songs. All other languages have been ignored. Even though analysis of the humming system can be done with two subtasks, one with classic QBSH evaluation (find the ground-truth melody from a user's singing or humming) and Variants QBSH evaluation (queries are variants of "ground-truth" melody) according to MIR [8], but in this study, it has focused mainly on classic QBSH evaluation approach. Further study will be categorized in genres and will be selected most promising genre after an initial study. E.g., it can be very difficult to find songs in Rock music (actual songs consist of many instrument variations, such as bass guitar, electric guitar, drums, electronic music organs, etc.) rather than Country music (which mostly consist of one or two instruments such as Guitars involved). Noisy, distorted inputs and all most of the genre-based matchings have given less priority. Study will be within the boundary of a selected genre of music, will be selected more simplest genre based on easily distinguish melodies between each other. The output of the study will be ranked identified set of songs based on score. The study's focus is mainly regarding improving accuracies of identified songs, in each stage micro evaluations required to be carried out, such as the 'hit'/'miss' ratio in each stage of development.

In this section it can be elaborate how to process Music hums transforming to its feature vectors thus will be provided to suitable matchers e.g., DTW, QDTW, etc. Then the music Data set will be selected and transformed to an application support feature vector. Finally derive a ranking system to sort results to retrieve most correct songs.

3.1. Abstract Design of the Model

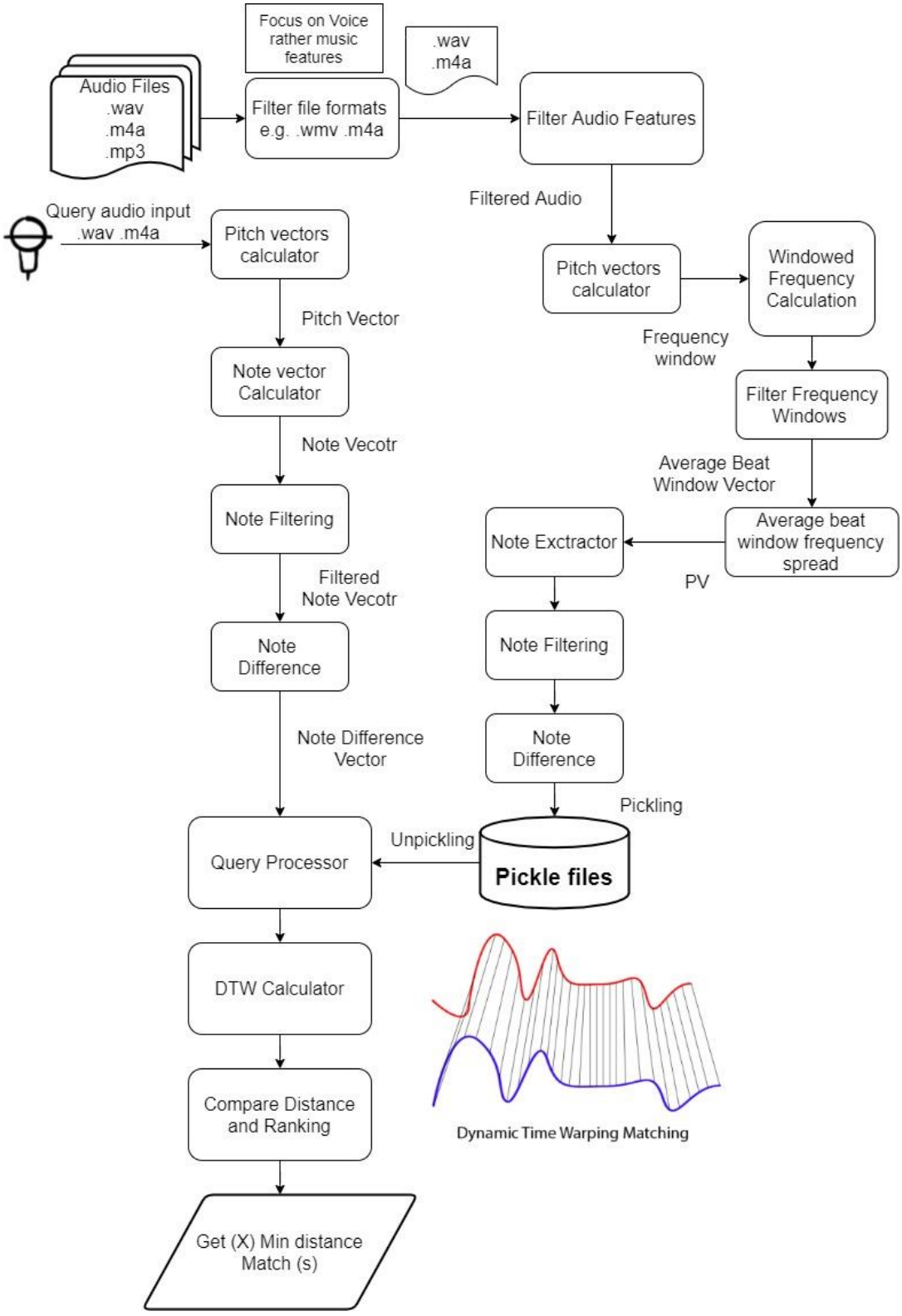


Figure 2: Model Design

As a start, humming to pitch vector conversion can be carried out based on which features we are going to focus on the most. This step is very important where the entire project will be based on the information we convert as windowed frequency lists. This step can be done focusing on pitch detection, contour representation, Energy of the signal etc. Which is a musical representation of the hummed song and the songs in the dataset. This representation is said to be most difficult in studies in conclusions [1,6]. Therefore, research on finding better music representation is required.

3.2. Pitch Vector Calculation

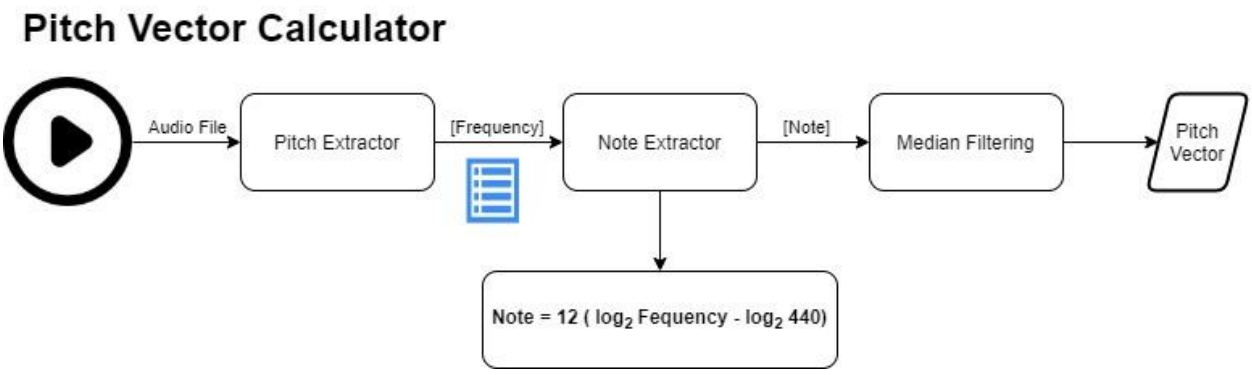


Figure 3: Pitch Vector Calculator

In figure 2 extract features from Humming tune to windowed frequency-time vectors (pitch vectors) has been illustrated. This area took most of research time to find features which yield the maximum outcome for feature matching in the latter part of the project. In evaluation of this study, it has been described how window size and hop size interval was decided with experimental study illustrated with graphical difference of frequency very with its window size.

When going into macro details of the pitch vector extractor, it has used pitch extractor to convert humming query or song to frequency vector. For this purpose, it had to use several techniques such as STFT and windowed ACF to calculate frequency for selected windowed approach.

Further study on music note segmentation on feature vectors to analyses music on segments as per in study of SoundCompass[4] approach.

- Music Information Retrieval (MIR)
- Convert Original songs sequences to feature vectors.

These features are identified in the previous steps for humming data. This is a preprocessing step which will be required to perform on original songs which will be used as a data set. For this purpose, require some publicly available songs with labels.

Query matching between hummed frequency-time feature vectors and original song feature vectors in segments. This is where core research will depend. Where by research what kind of approaches require to query the outcome of both original and hummed songs. As a start in this research planning to go with existing approaches such as Dynamic Time Warping (DTW), QDTW, time normalizing DTW, early stopping algorithm [6], Fast Time Wrapping (FTW), etc. This step seems easy but reality is not giving expected results in most approaches. Which limits some critical matters such as measuring distortion in durations if used DTW. Therefore, this area required extra research effort.

Search results will be listed based on the rankings which have the highest score among distance matches, or any other suitable approach found during research study.

3.3. Autocorrelation Function (ACF) to Pitch Estimation

ACF approach is fast frequency calculation approach used in this study to measure frequency of a wave at a given window. ACF is a measure of the correlation between observations in a time series that are separated by a unit of time. It refers to the correlation of a time series with its own past and future values. ACF requires at least 2 cycles of wave to operate [12]. ACF is calculated using copy of wave itself as a function of delay [13]. Informally, it is the similarity between observations as a function of the time lag between them[wiki].

Once ACF is calculated it is required to calculate another set of steps before actually estimate pitch.

- Find the second order differences
- Find where second order differences are negative (which gives peaks)
- From those peaks find index with max value.

Max value of peaks contains the delay required to calculate pitch which can be obtained by dividing frame rate by this delay.

$$\boldsymbol{Pitch = \frac{Frame\ Rate}{Delay}}$$

Equation 2: Final Pitch Calculation

3.4. Pitch to Note Vector Transformation

In the domain of music engineering sound at different frequency levels are represented by Notes. For an e.g., Pitch Frequency of 440Hz has been represented by musical note 'A'. Between these notes there is frequency gap which is called as semi tone. For an e.g. There are two semitones between note A and B which can be represented as [A, A#, B] or [A, B♭, B].

Thus, this note transformation can be conducted using following equation.

$$\boldsymbol{note = 12(log2\ frequency - log2\ 440)}$$

Equation 3: Note Calculation

This equation provides notes in relative scale compared value indexed at 440Hz representing value 0. Therefore, any frequency above 440Hz will scaled as positive and below will be vise versa. After 12 iterations at 880Hz it occurs a harmony. Which is again represented by note A but in a different octave.

3.5. Frequency Threshold by Energy

At Frequency domain transformation it doesn't need to calculate ACF related calculation for some wave amplitude below this threshold. Which saves time by ignoring low energy wave components. In this study with minor experimental tests, it has decided to keep this threshold value at **30% of max energy** of the wave.

3.6. Filtering

This plays the final role before we compare DTW distance between waves. Which filters notes and normalize further to get rid of things like outliers. Filtering includes median filtering, first max frequency filtering, difference filtering, refine model filtering. Each filtering occurred at several stages.

3.6.1. **Median Filtering**

Median filtering is a nonlinear method used to remove noise from the note vector. What this do is replace nearby note values by median values. Thus, noisy distributions will be eliminated.

3.6.2. **Average Frequency Filtering**

This filter created to identify average frequency value obtained within a hop window and persist the average value. Thus, it can clearly visualize where frequency value occurs throughout a hop window. This is very helpful in later stage for calculating note difference in NumPy difference filter.

3.6.1. **Refine Model**

This is the stage which refine notes where it goes to NaN and Inf values while transforming pitch vector to note vector. Which is byproduct created by divide by zeros give Inf (where input frequency at zero) and NaN when frequency excluded by threshold which set to -1.

3.6.1. **Difference Filtering**

Difference is performed in notes vector to normalize the variation of notes. This step is where it calculates ups and downs of variations which will be compared among hummed and real song variations. Thus, will give lower DTW distance.

4. IMPLEMENTATION

In this section it has been elaborate how the system has been implemented using sound engineering techniques and other filtering and comparing techniques. Further additional implementation details such as client interactive interface which created using angular, python REST bridge using flask which was very helpful in such web-based integrations.

4.1. Audio Wave Processing

Before audio wave processing, it is required to get analog information in a digital scale. For this purpose, it has been done mainly using python's library called 'audiosegment'. Which read the audio files and provides useful information such as frame rate and raw audio data which consist of the digital way of representing real wave. Scaling on both negative and positive regions in an oscillation.

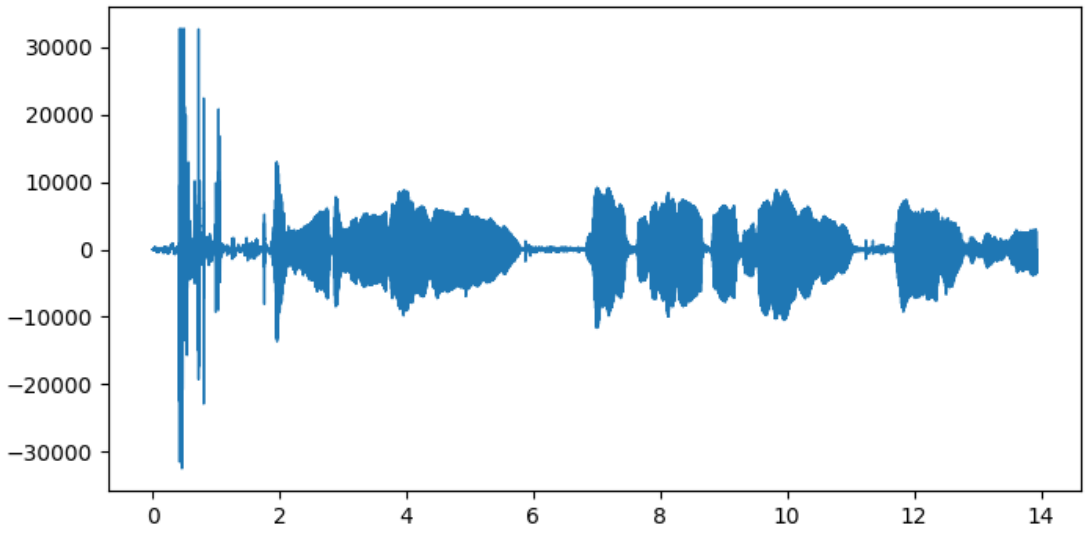


Figure 4: Standard Audio Wave - Signal Attitude/ Time

Then these read data has channels based on given input. However, in this study having single channel of information is adequate. So, filtering channel information into single scale and it has been acquired single channel (first channel) to carry out the rest of the study. However, it is mandatory to note that some filtrations like low band filtering should be performed before channel

separation. Once channel information is retrieved then it is required to represent this audio signal information in more informative way. Which is in a scale of frequency over time.

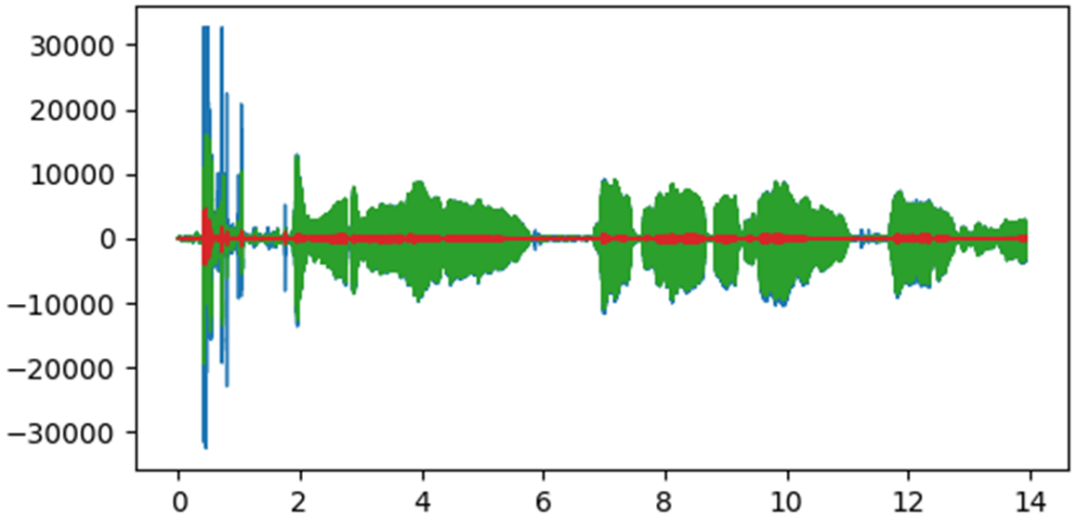


Figure 5: Band Filtered Audio Wave

4.2. Frequency Windowing

For frequency windowing it can be used many approaches such as STFT, ACF. However, in this approach it has taken ACF (Auto Correlation function) based approach to calculate frequency per given window of signal change. Assumption here is in short window frequency change is very small. So, after calculating ACF finding second order differences and identifying negative order difference gives the peak of each sequence change. Then it can take maximum of these peak values has the maximum delay required to calculate the frequency to calculate the correct frequency value by frame rate to delay ratio.

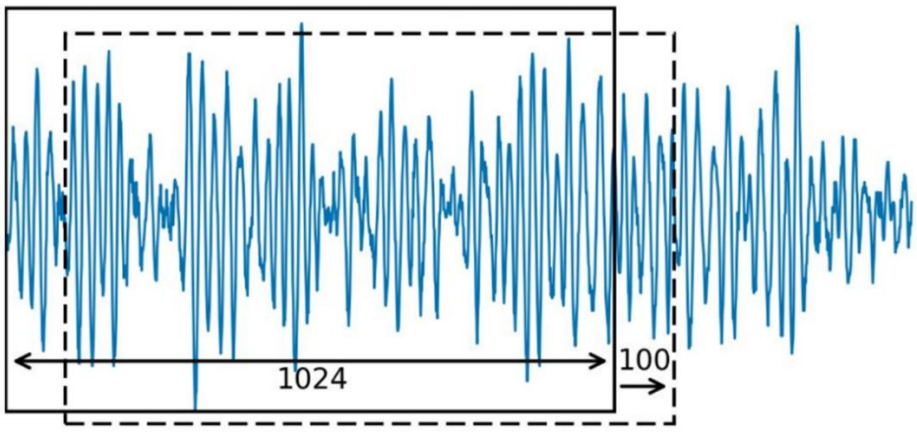


Figure 6: Frequency Windowing

4.3. Frequency Windows Averaging to Beat Windows

Once Frequency vectors obtained by Frequency windows it needs further generalized into beat window frames. Where the assumption is beat frame has the minimum length that particular frequency level upholds. Where later this beat windows will be converted to note vectors. In order to do that frequency vector again categorized into beat windows by calculating BPM (Beats per seconds) using 'librosa' library and then calculating BPS (Beats per second). Next stage is purely experiment basis averaging filter applied on these windows to get weighted average frequency over beat window. Thus, it will give weighted average value per each beat window. Now these frequency vectors can be provided to note vector converter to generate notes out of frequencies.

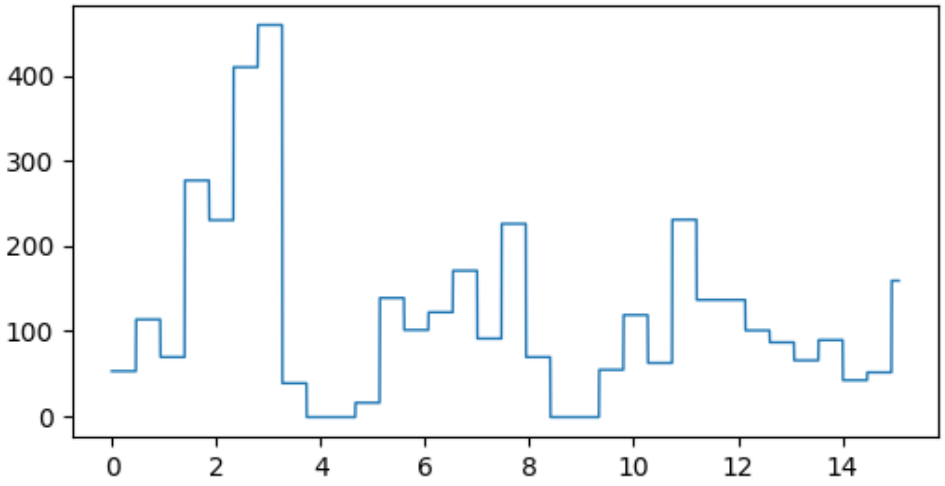


Figure 7: Average Frequency/ Beat windows

4.4. Converting Beat Window Pitch Vectors to Note Vectors

In Note converter it takes frequency values and convert them based on index value of frequency value A which is 440Hz as 0. Then all other values respective to A will categorize in a scale from negative to positive. Based on it frequency values. Equation is considered half note level gap between each note. For e.g., A->B difference is 0->2. Where it represents as A, A#/B♭, B. In earlier chapter it has elaborate how equation ($\boldsymbol{note = 12(log2\ frequency - log2\ 440)}$) has been used to find this note values.

4.5. Evaluation model

In order to evaluate model it has used evaluation module which has supported using 'sklearn' metrics. Where it provide confusion matrix and classification report detail support. To support this model separate evaluation model has been created to convert model data and results to matching parameters to this 'sklearn' based evaluation.

```
def evaluate_model():
    file = "data/selected_set"
    model = unpickle_data(file=file_pickle_rename(file))
    filters = ['m4a']
    test_files = get_filtered_files(test_file_dir, filters)
    tests_result = {}
    actual_data = []
    for test_file in test_files:
        actual_data.append(file_path_to_name_formatter(test_file))
        query_data = create_query_data(test_file)
        _list = query(model, query_data, False)
        temp = []
        for i, predict in enumerate(_list[:5]):
            name, dis = predict
            temp.append(file_path_to_name_formatter(name))
        tests_result[test_file] = temp
    print('Predicted:' + str(tests_result))
    print('Actual data:' + str(actual_data))
```

4.6. Query Song to Humming data Approach

Querying is the most time-consuming and highest time complexity part of the model's algorithm. To overcome this hurdle, it has been introduced concurrent future approach. Where by run all DTW related calculations (using a worker called distance_calculator_worker) with parallel to each other and compute final values at the completion of all future tasks.

```python
def query(data_model, _query_pv, zero_remove=False):
    log_time("query start")
    distance = {}
    log_time("data_model:loop start")

    with concurrent.futures.ThreadPoolExecutor() as executor:
        futures = [executor.submit(distance_calculator_worker, item, _query_pv, zero_remove) for item in
                   data_model.items()]
        for future in concurrent.futures.as_completed(futures):
            distance.update(future.result())

    log_time("data_model:loop end")
    sorted_items = dict(sorted(distance.items(), key=lambda item: item[1])).items()
    item_list = list(sorted_items)
    log_time("query end")
    return item_list
```

Figure 8: Distance Calculation using concurrent futures and workers

4.7. Client Interactive Interface

Addition to research area it has been introduce small program to clients so they can interact with this model. Using Angular this front end has been created. Where it can capture humming vocals and let user listen to what they have hummed. And submit to model so they can get predication based on most matching top 10 results with its distance values.

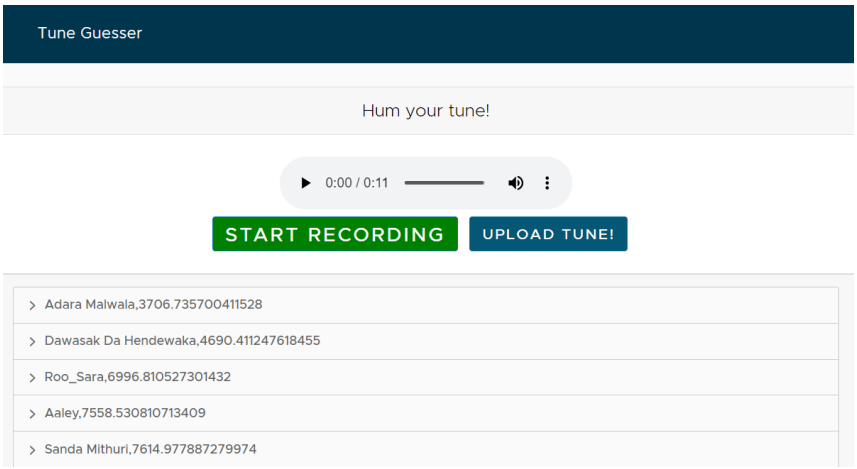


Figure 9: Client Interface

4.8. Flask Bridging between Python and REST

In order to plug client interface between Python based project it has been created Flask bridge REST layer. Which has allowed to upload hummed audio files using POST method and retrieve response in JSON based output.

5. EVALUATION AND RESULTS

Evaluation chapter elaborate the approach used to measure the proposed solution's reliability by conducting experimental based evaluation. Decision of accepted outcome is achieved is measured based on multiple acceptance criteria. Since requirement of hum at more closely to real song have major impact evaluating results just for top most result gives very poor result for non-perfect pitch humming tunes, this study has focused on in following identification techniques. Initially identify correct result within top 5 hit for a given hum input was measured. Data sets used to carry out the testing was real songs which were specially processed to capture the beginning of the main melody or chorus of a song and end within in 10 seconds to 15 seconds of the duration. This is actually narrowed scope to identify song rather identifying tune of the song itself. To calculate Accuracy, precision, recall and f1-score this study also consider standard confusion using a separate evaluation method written to detect each value. This was done due to difficulty achieving first most value due to very close DTW value difference between each actual to hum songs were achieved. Therefore, study considering to get top most matches as correct result when evaluating among other songs. So once predicted song doesn't land in top results then will be taken to be true positive or false positive based on actual first match in the evaluation.

Data set used to test humming's test set has been set to 9 selected songs within range of humming tunes, and collected Hummings are multiples of 9 songs with respect to good and bad Hummings which adds up to 18 per person. Then study will be carried based on two major categories which gives results for good Hummings and bad Hummings separately. This was done due to recognize how model vary based on bad humming inputs. Later it can do a regression analysis based on these data bad Hummings has direct correlation to selecting correct or similar song/music.

Further study has done individual unique feature-based analysis. Such low band filtered set, zero filtered, hop per beat change, etc. These results will indicate how each individual feature affect the results and how overfitting may cause which make almost all Hummings categorized mostly at the same song.

5.1. Confusion Matrix

Confusion matrix is a NxN matrix, which is a tabular representation of model predictions vs actual values. Each column and row are dedicated to one class. On one end we have the actual values and on the other end predicted values. In our case rows and columns represent hummed song query to real song

- Since in this research we have not only consider top most result, only true positive and false Negatives will be classified, Negatives will be put under Non match class for better understanding of not correctly identified song for a given humming.

5.2. Evaluation Metrics

- Accuracy: the proportion of the total number of humming tune predictions that were correct.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{True\ number\ of\ samples}$$

Equation 4: Accuracy Calculation

- Precision: the proportion of positive cases that were correctly identified.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Equation 5: Precision Calculation

- Recall: the proportion of actual positive cases which are correctly identified.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Equation 6: Recall Calculation

- F1 Score: F1-Score is the harmonic mean of precision and recall values for a classification problem

$$F1 = \left(\frac{recall^{-1} + precision^{-1}}{2}\right)^{-1} = 2.\left(\frac{precision\,.recall}{precision\ +\ recall}\right)$$

Equation 7: F1 Score Calculation

5.3. Top 5 Listing Evaluation with 9 Hums to 9 Songs Summing up to 12 Song Tracks

Confusion matrix of indicates how results has been identified by the model by spreading its matches over the matrix. In diagonal axis along top left comer to bottom right corner indicates correctly matched elements of model.

5.3.1. **Good Pitched Hummings over Model**

	1	2	3	4	5	6	7	8	9
1. Aley	**1**	0	0	0	0	0	0	0	0
2. Adara Malwala	0	**3**	0	0	0	0	0	0	0
3. Ananathayata yana	0	0	**3**	0	0	0	0	0	0
4. Chandrayan pidu	0	0	0	**4**	0	0	0	0	0
5. Dagalanna ba	0	0	0	0	**4**	0	0	0	0
6. Galana gaga	0	1	1	0	0	**4**	0	0	0
7. Kuweni	2	0	0	0	0	0	**4**	0	1
8. Roo Sara	0	0	0	0	0	0	0	**4**	0
9. Sanda Mithuri	1	0	0	0	0	0	0	0	**3**

Table 1: Confusion Matrix: Good Pitched Hummings

For the tested over 36 Hummings over 12 song tracks (of selected 9 songs) has given following results in classification report. According to this classification it can be seen model accuracy is over 80% for given humming with good precision (nearly perfect pitch). It can then analyze that some of songs have different behavior of getting recall such as “Kuweni” has 0.57 recall which leads other Hummings to miss match and identified it as this. However, with f1-score it has reached 0.73 which can be identified as yet acceptable margin for such Sound Engineering based model. Evaluation only has test set since model requires not training data set since it based on highly sound engineered mechanical structured model. With each feature change we can clearly display how the model behave and classify hummed queries in different songs.

	Precision	Recall	f1-score	Support
Adara mal wala	0.25	1	0.4	1
Aley	0.75	1	0.86	3
Anantayata yana	0.75	1	0.86	3
Chandrayan pidu	1	1	1	4
Dagalanna ba	1	1	1	4
Galana Ganga	1	0.67	0.8	6
Kuweni	1	0.57	0.73	7
Roo Sara	1	1	1	4
Sanda Mithuri	0.75	0.75	0.75	4
Accuracy			**0.83**	36
Macro avg	0.83	0.89	0.82	36
Weighted avg	0.91	0.83	0.85	36

Table 2: Good Pitched Hummings Classification Report

5.3.2. **Bad (Careless) Pitched Hummings over Model**

	1	2	3	4	5	6	7	8	9
1. Aley	**1**	0	0	0	0	0	0	0	0
2. Adara Malwala	0	**2**	0	0	0	0	1	0	0
3. Ananathayata yana	0	0	**3**	0	0	0	0	0	0
4. Chandrayan pidu	1	0	0	**3**	0	0	0	0	0
5. Dagalanna ba	0	0	0	0	**3**	0	0	0	0
6. Galana gaga	1	1	1	0	1	**3**	0	0	1
7. Kuweni	0	0	0	0	0	0	**3**	1	0
8. Roo Sara	0	1	0	0	0	1	0	**3**	0
9. Sanda Mithuri	1	0	0	1	0	0	0	0	**3**

Table 3: Confusion Matrix: Bad (Careless) Pitched Hummings

	Precision	Recall	f1-score	Support
Adara mal wala	0.25	1	0.4	1
Aley	0.5	0.67	0.57	3
Anantayata yana	0.75	1	0.86	3
Chandrayan pidu	0.75	0.75	0.75	4
Dagalanna ba	0.75	1	0.86	3
Galana Ganga	0.75	0.38	0.5	8
Kuweni	0.75	0.75	0.75	4
Roo Sara	0.75	0.6	0.67	5
Sanda Mithuri	0.75	0.6	0.67	5
Accuracy			**0.67**	36
Macro avg	0.67	0.75	0.67	36
Weighted avg	0.72	0.67	0.66	36

Table 4: Bad (Careless) Pitched Hummings Classification Table

From the results it can see for Bad Hummings there is drastically dropped match rate accuracy around 60% range. This indicates model requires more clean and pitched at almost similar or same level pitch to get good results.

5.3.1. **Distance Indexed DTW Approach**

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1. Aley	**2**	0	0	0	0	0	0	0	0
2. Adara Malwala	0	**3**	0	0	0	0	0	0	0
3. Ananathayata yana	0	0	**3**	0	0	0	0	0	0
4. Chandrayan pidu	0	0	0	**3**	0	0	0	0	0
5. Dagalanna ba	0	0	0	1	**4**	0	2	0	1
6. Galana gaga	0	0	0	0	0	**3**	0	0	0
7. Kuweni	0	1	0	0	0	0	**1**	0	0
8. Roo Sara	1	0	0	0	0	0	0	**4**	0
9. Sanda Mithuri	1	0	1	0	0	1	1	0	**3**

Table 5:Distance Index DTW Approach Confusion Matrix

	Precision	Recall	f1-score	Support
Adara mal wala	0.5	1	0.67	2
Aley	0.75	1	0.86	3
Anantayata yana	0.75	1	0.86	3
Chandrayan pidu	0.75	1	0.86	3
Dagalanna ba	1	0.5	0.67	8
Galana Ganga	0.75	1	0.86	3
Kuweni	0.25	0.5	0.33	2
Roo Sara	1	0.8	0.89	5
Sanda Mithuri	0.75	0.43	0.55	7
Accuracy			**0.72**	36
Macro avg	0.72	0.8	0.73	36
Weighted avg	0.8	0.72	0.72	36

Table 6: Distance Indexed DTW Classification Table

In Distance indexed DTW approach it tries to index distance relative to zero scale or silent tone which gives distance values between itself and DTW silent tone. Then these values will be reduced when calculating the real value between real song and hummed tune. This approach was tested to get rid of any biasness caused by some specific features like strong musical features, strong vocal support, high temp variations, etc. So, all the measurements would've taken in a same index. However, while evaluating this approach, it seems this biasness removal technique is not that much supportive compared to direct DTW assessment. Additionally, DTW values has gone negative for some previously high biased songs which negating more distance score. Since it gave less score than original approach it can relate that indexing with distance for DTW won't give better results.

5.3.1. **Without Low Band Filtering**

	1	2	3	4	5	6	7	8	9
1. Aley	**2**	0	0	0	0	0	0	0	0
2. Adara Malwala	0	**3**	0	0	0	0	0	0	0
3. Ananathayata yana	0	0	**4**	1	2	0	0	0	0
4. Chandrayan pidu	0	0	0	**2**	0	0	0	0	0
5. Dagalanna ba	0	1	0	0	**1**	0	0	0	0
6. Galana gaga	0	0	0	0	0	**4**	0	0	0
7. Kuweni	0	0	0	0	0	0	**2**	0	0
8. Roo Sara	0	0	0	1	0	0	0	**4**	0
9. Sanda Mithuri	2	0	0	0	1	0	2	0	**4**

Table 7: Without Low Band Filtering Confusion Matrix

	Precision	**Recall**	**f1-score**	**Support**
Adara mal wala	0.5	1	0.67	2
Aley	0.75	1	0.86	3
Anantayata yana	1	0.57	0.73	7
Chandrayan pidu	0.5	1	0.67	2
Dagalanna ba	0.25	0.5	0.33	2
Galana Ganga	1	1	1	4
Kuweni	0.5	1	0.67	2
Roo Sara	1	0.8	0.89	5
Sanda Mithuri	1	0.44	0.62	9
Accuracy			**0.72**	36
Macro avg	0.72	0.81	0.71	36
Weighted avg	0.85	0.72	0.73	36

Table 8: Without Low Band Filtering Classification Table

Low band filtering is a feature filtering technique which reduce musical instruments related frequency band thus resulting vocals sections to be highlighted. Doing so it can see some features

like strong music reduction has better chance of identifying matching song using hum with results. It can say with proper music filtration humming and vocals of song can match more conveniently.

5.3.2. **Multiple Sound Tracks for Local Minima Analysis**

	1	2	3	4	5	6	7	8	9
1. Aley	**1**	0	0	0	0	0	0	0	0
2. Adara Malwala	0	**3**	0	0	0	0	0	0	1
3. Ananathayata yana	1	0	**3**	0	0	0	0	0	0
4. Chandrayan pidu	0	0	0	**4**	0	0	0	0	0
5. Dagalanna ba	0	0	0	0	**4**	0	0	0	0
6. Galana gaga	0	1	1	0	0	**4**	0	0	0
7. Kuweni	2	0	0	0	0	0	**4**	0	1
8. Roo Sara	0	0	0	0	0	0	0	**4**	0
9. Sanda Mithuri	0	0	0	0	0	0	0	0	**2**

Table 9: Multiple Sound Tracks for Local Minima Confusion Matrix

	Precision	**Recall**	**f1-score**	**Support**
Adara mal wala	0.25	1	0.4	1
Aley	0.75	0.75	0.75	4
Anantayata yana	0.75	0.75	0.75	4
Chandrayan pidu	1	1	1	4
Dagalanna ba	1	1	1	4
Galana Ganga	1	0.67	0.8	6
Kuweni	1	0.57	0.73	7
Roo Sara	1	1	1	4
Sanda Mithuri	0.5	1	0.67	2
Accuracy			**0.81**	36
Macro avg	0.81	0.86	0.79	36
Weighted avg	0.9	0.81	0.82	36

Table 10: Multiple Sound Tracks for Local Minima Classification Table

In this Some selected songs have been selected based on what users tried to hum to identify which parts of song is related to the original song. For e.g., some tried to hum chorus where as some tried to hum start of the song. Based on that rather dynamically process entire song with song chunks, it has created specific song parts which includes parts original songs belongs to these chorus and

start of track parts lengthen max 15 seconds durations. Then it can be seen With such inclusions local analysis matters to get better result than having only entire song as global context. Which by eliminating most unwanted musical instrumental parts and other song's repetitive parts which can be obviously understood by start or chorus primarily.

5.3.3. **Selection of Time per Hop in Frequency Windowing Relation**

	1	2	3	4	5	6	7	8	9
1. Aley	**2**	0	0	0	0	0	0	0	0
2. Adara Malwala	0	**2**	0	1	1	0	2	0	0
3. Ananathayata yana	1	0	**4**	0	0	0	0	0	0
4. Chandrayan pidu	0	0	0	**2**	1	0	0	0	0
5. Dagalanna ba	0	0	0	0	**1**	0	0	0	0
6. Galana gaga	0	2	0	0	0	**4**	0	0	0
7. Kuweni	1	0	0	0	0	0	**2**	0	0
8. Roo Sara	0	0	0	1	0	0	0	**4**	0
9. Sanda Mithuri	0	0	0	0	1	0	0	0	**4**

Table 11: 0.0016 Time per Hop Value Confusion Matrix

	Precision	**Recall**	**f1-score**	**Support**
Adara mal wala	0.5	1	0.67	2
Aley	0.5	0.33	0.4	6
Anantayata yana	1	0.8	0.89	5
Chandrayan pidu	0.5	0.67	0.57	3
Dagalanna ba	0.25	1	0.4	1
Galana Ganga	1	0.67	0.8	6
Kuweni	0.5	0.67	0.57	3
Roo Sara	1	0.8	0.89	5
Sanda Mithuri	1	0.8	0.89	5
Accuracy			**0.69**	36
Macro avg	0.69	0.75	0.68	36
Weighted avg	0.78	0.69	0.71	36

Table 12: 0.0016 Time per Hop Value Classification Table

	1	2	3	4	5	6	7	8	9
1. Aley	**1**	0	0	0	0	0	0	0	0
2. Adara Malwala	0	**2**	0	0	0	0	0	0	0
3. Ananathayata yana	0	1	**3**	0	0	0	0	0	1
4. Chandrayan pidu	1	0	0	**4**	0	0	0	0	0
5. Dagalanna ba	0	0	0	0	**3**	0	0	0	0
6. Galana gaga	1	0	0	0	0	**4**	0	0	0
7. Kuweni	0	1	0	0	0	0	**4**	0	0
8. Roo Sara	0	0	1	0	0	0	0	**4**	0
9. Sanda Mithuri	1	0	0	0	1	0	0	0	**3**

Table 13: 0.0064 Time per Hop Value Confusion matrix

	Precision	**Recall**	**f1-score**	**Support**
Adara mal wala	0.25	1	0.4	1
Aley	0.5	1	0.67	2
Anantayata yana	0.75	0.6	0.67	5
Chandrayan pidu	1	0.8	0.89	5
Dagalanna ba	0.75	1	0.86	3
Galana Ganga	1	0.8	0.89	5
Kuweni	1	0.8	0.89	5
Roo Sara	1	0.8	0.89	5
Sanda Mithuri	0.75	0.6	0.67	5
Accuracy			**0.78**	36
Macro avg	0.78	0.82	0.76	36
Weighted avg	0.86	0.78	0.8	36

Table 14: 0.0064 Time per Hop Value Classification Table

This is another feature related coefficient found during the study. Where had to incur significant time to get hold with how this value behave which give optimal time and qualitative measurements. From the results it can be seen when given low value for this it goes beyond the scale and tries to identify songs for Hummings with overfitting scenario. However, when reducing value beyond some point results becomes meaningless. So, in later our study we have shown how this value

selection has been achieved by step-by-step evaluation where it has given more realistic and time efficient results based on coefficient at around 0.0032 time per hop.

5.4. Humming to Real Song Matching

Following is sample query alignment matched to real song

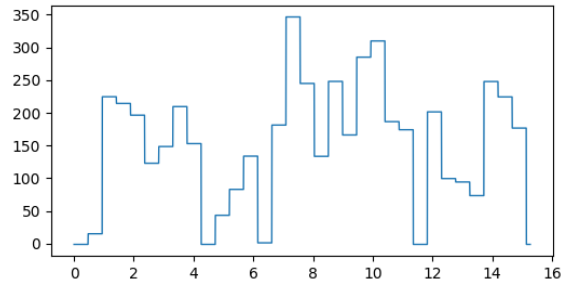
<table>
<tr><td colspan="2">Nadi ganga tharanaye by Chithral Somapala (15 seconds split)</td></tr>
<tr><td>Frequency domain of Hum after average filter
 

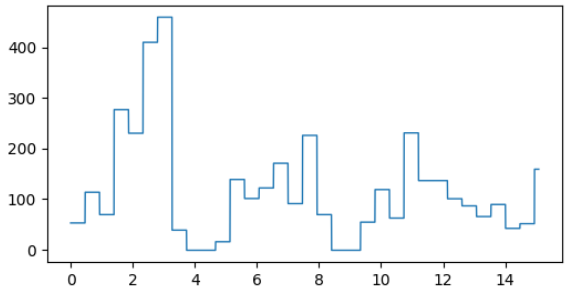
Figure 10: Frequency Domain of Humming after Average Filter</td><td>Frequency domain of Song after average filter
 

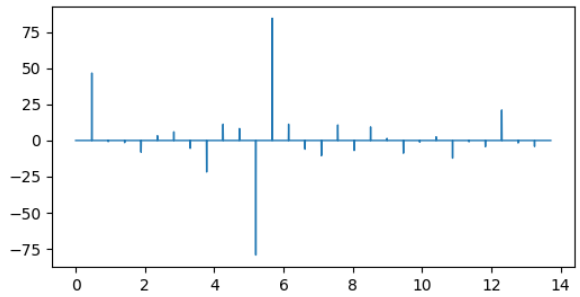
Figure 11: Frequency Domain of Song after Average Filter</td></tr>
<tr><td>Difference of notes Filtered silences of Hum
 

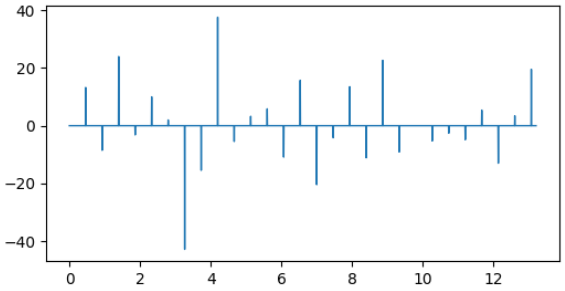
Figure 12: Note Difference of Humming</td><td>Difference of notes Filtered silences of Song
 

Figure 13: Note Difference of Song</td></tr>
<tr><td>Nadi ganga tharanaye Hummed qury
Frequency - Time</td><td>Nadi ganga tharanaye Song
Frequency - Time</td></tr>
</table>

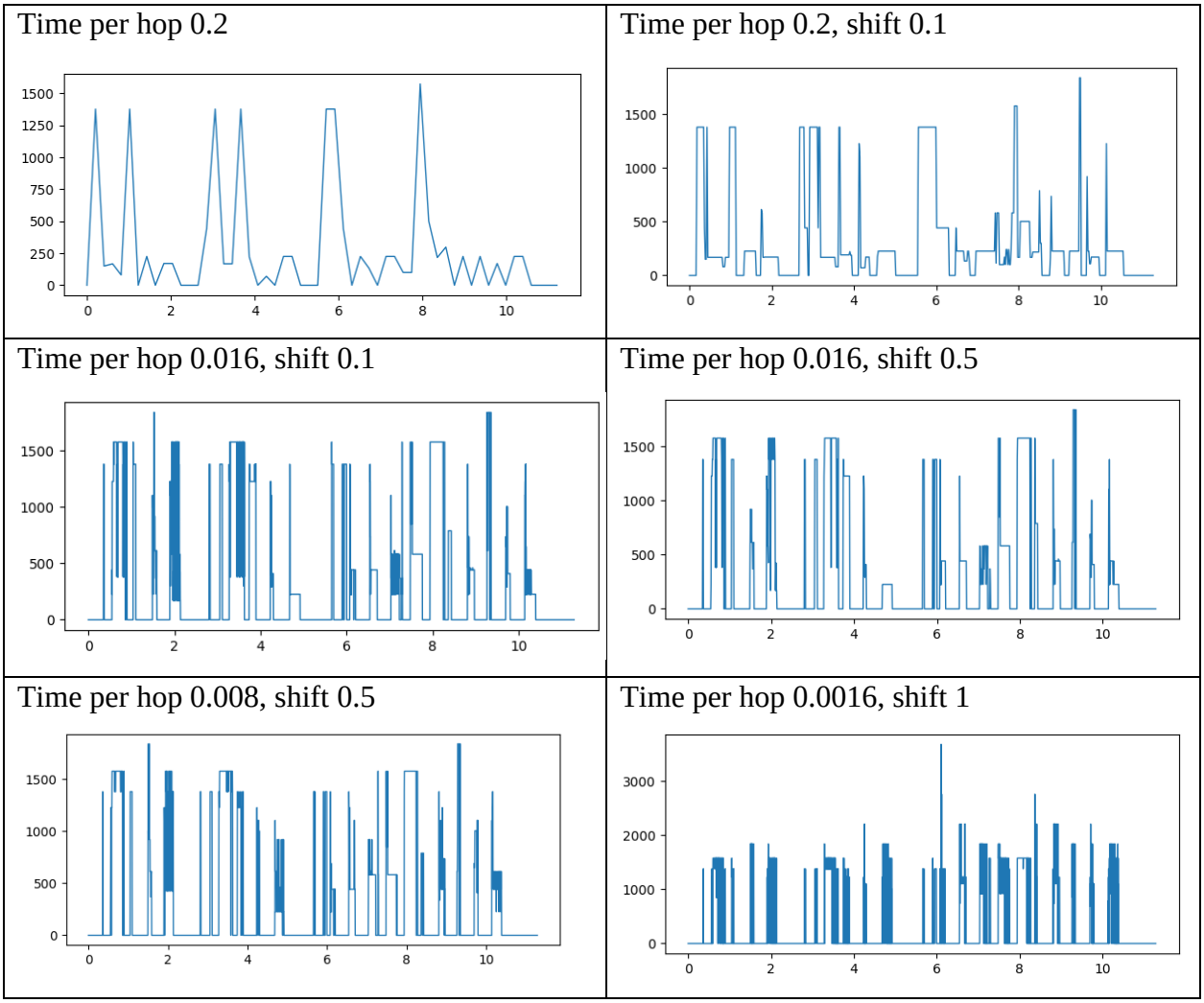
Table 15: Humming to Real Song Matching

Table 1 indicates That hum query frequency – time graph (Figure 4) compared with real song's frequency – time graph (Figure 5), both has similar shape of peaks patters but in different frequency ranges. Since in comparisons we consider difference between shapes of the peaks it can compare most of the frequency peaks have similar changing behavior compared to hummed and original song. In Figure 6 and Figure 7 it can see how the Note vector difference compared with

Hummed query and Song. By calculating difference of pitch variation, we can normalize different pitch levels to a common value. Further this study has been considered about local minima analysis which only consider part of the humming will be match with part of the main song where quantized DTW will calculated. This will give much less DTW distance score if found a correct match. Initially this local analysis was about to carried out based on algorithmic way. But doing so it has been instead time and space complexity which algorithm required to carried out DTW recursively over each sequence for local areas. Since DTW consumes more time resource than any other algorithmic areas Local analysis weight has been reduced using clipped audio files.

5.5. Frequency Window Size Estimation

Step by step frequency over time improvement with window size. (Happy Birthday Music .wav)



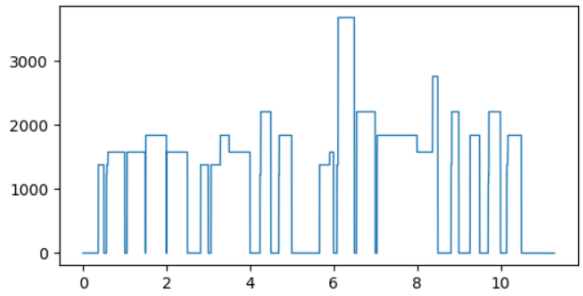
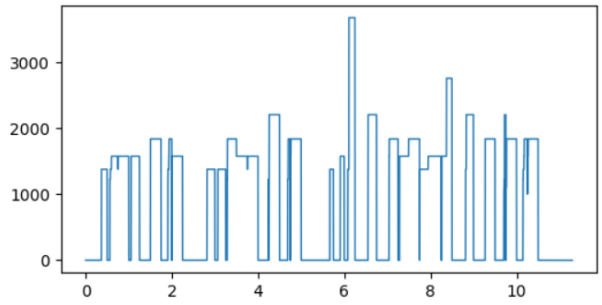
Time per hop 0.0032, shift 1 Applied Average pitch filter: 306 hop rates (half window) 	Time per hop 0.0016, shift 1 Applied first max pitch filter: 1/2 beat (1/4th sound), 153 hop rate (quarter window) 

Table 16: Step by Stem Frequency Step Analysis

Frequency for each hop is calculated by deciding random hop time for given frame rate of the song. However, this size needs to optimized to get an optimal output which is decided by a domain specialist which each hops window will be considered to as note frequency for a brief period of time. Which is then decided by tempo of song and calculating half beat length. By this graph it can see from 0.2 seconds hop time to 0.0016 hop time it has improved its distinct nature of frequency for each frame. Then by applying first average pitch filter we can graphically illustrate at ½ beat frequency levels by each pillar. However further reducing hop rate from 306 to 153 (1/2 times) thin it immerged frequency differences than required to identify original semitone differences between beats. So, its overfitting to give in depth knowledge which is unique to the selected song itself. There for optimal solutions hop rate kept half of the initial hop rate which will be calculate using $(\text{time for hop})^{-1}$.

5.6. Data Sets

There are two types of data sets used for model creation. One is Humming audio set and Song Chunk data sets. So basically, both sets will be used as test set for evaluate the module. Since there is no training involve with the study every corresponding match can be considered as a test. However, model development and fine tuning have been carried out based on a selected set of data.

a. Humming data sets used for the study MTG-QBH [10] and MIR-QBSH [11] data sources plus privately created data set (mainly used) for Sinhala songs.

b. Humming data gathered from real people who contributed to provide Hummings. Which are given as good set (trying extra effort to keep pitch correct when humming) and bad set (hum freely effortlessly plus going off pitch) of Hummings.
c. Songs' data set mainly created using existing real songs as partial chunks splice at song vocals begin constant times (10 -15 seconds)

5.7. Baseline for Results

Baseline for evaluation will be based on DTW distance score between each other songs. Lesser the distance better it is. If real song used as test data DTW distance will be 0. With humming tune to song, getting a match closer to 0 is not possible. However even match coming with top most in the list with greater DTW distance can be considered as no match and smaller DTW distance but not the correct song also as no match.

6. CONCLUSION AND FUTURE WORK

In this section, conclusion of the system and model designed is explained with Further improvements that can be done to the current implementation.

Study was conducted in heavy musical engineering aspects where need long learning curve to come to a point where it requires to understand how waves behave and how to tackle with challenges like identify frequency at which point. However, with literature conducted in several areas related to selected area for current study was very helpful achieving greater results at the end of the study.

This research mainly focuses on identifying original song for hummed song. With the current model with current test effort, it has reached more than 80% accuracy for randomly hummed tunes. With Local minimum sequence analysis approach frequency difference comparison has achieved much higher accuracy values.

In this study mainly focused on Sinhala and English songs which is mainly used on low distorted original music files. If used on Low varying songs in genre like pop, country, etc. Study gives acceptable results. However, it is still requiring to pitch Hummings better to perfect pitches. Otherwise, model confuse hum with other songs with less DTW scores. As further work pitch correcting before applying to model can be considered to boost accuracies of bad pitch Hummings.

According to the literature humming tone can be vary from person to person, age differences, gender differences, even can be affected by demographical factors. As a further step we can consider research on these factors and how model will respond to such factors.

With the current pandemic situation increasing use of online institutions for almost every need. Finding for songs or even audio files require more efficient and fluent search engine like google for text-based searches. Therefore, Audio file search could be developed with the help of current model with tweaking local minima approach.

For current study it had no humming dataset for Sinhala songs as it was for English but done by east Asian countries with bit biased pitch for those countries accent. This could be improved by having community support to grow a Sinhala song hummed database so people who research in this area can be benefit and compare results among each model.

APPENDICES

APPENDIX A

- Full results

```
[[1 0 0 0 0 0 0 0 0]
 [0 3 0 0 0 0 0 0 0]
 [0 0 3 0 0 0 0 0 0]
 [0 0 0 4 0 0 0 0 0]
 [0 0 0 0 4 0 0 0 0]
 [0 1 1 0 0 4 0 0 0]
 [2 0 0 0 0 0 4 0 1]
 [0 0 0 0 0 0 0 4 0]
 [1 0 0 0 0 0 0 0 3]]
                  precision    recall  f1-score   support

 Adara mal wala       0.25      1.00      0.40         1
           Aley       0.75      1.00      0.86         3
Anantayata yana       0.75      1.00      0.86         3
Chandrayan pidu       1.00      1.00      1.00         4
   Dagalanna ba       1.00      1.00      1.00         4
   Galana Ganga       1.00      0.67      0.80         6
         Kuweni       1.00      0.57      0.73         7
       Roo Sara       1.00      1.00      1.00         4
  Sanda Mithuri       0.75      0.75      0.75         4

       accuracy                           0.83        36
      macro avg       0.83      0.89      0.82        36
   weighted avg       0.91      0.83      0.85        36
```

REFERENCES

[1] Tripathy, A., 2009. Query By Humming System.

[2] Patel, P., 2019. Music Retrieval System Using Query-By-Humming. SJSU ScholarWorks.

[3] Putri, R. and Lestari, D., 2015. Music Information Retrieval Using Query-By-Humming Based On The Dynamic Time Warping. The 5th International Conference on Electrical Engineering and Informatics 2015.

[4] Kosugi, N., Sakurai, Y. and Morimoto, M., 2014. Soundcompass: Normalization Of Scalable And Shiftable Time-Series Data And Effective Subsequence Generation. NTT Cyber Space Labs.

[5] Li-Chun Wang, A., 2003. An Industrial-Strength Audio Search Algorithm. Shazam Entertainment, Ltd.

[6] DURRIEU, J., 2006. MUSIC INFORMATION RETRIEVAL A QUERY-BY-HUMMING (QBH) SYSTEM SEGMENTATION Of The SONGS And APPROXIMATIVE MELODY MATCHING Based On The DTW Algorithm.

[7] Antonelli, M., Rizzi, A. and Vescovo, G., 2010. A Query By Humming System For Music Information Retrieval. 10th International Conference on Intelligent Systems Design and Applications.

[8] Music-ir.org. 2021. 2020:Query by Singing/Humming - MIREX Wiki. [online] Available at: <https://www.music-ir.org/mirex/wiki/2020:Query_by_Singing/Humming> [Accessed 4 February 2021].

[9] Google. 2021. Song stuck in your head? Just hum to search. [online] Available at: <https://blog.google/products/search/hum-to-search> [Accessed 4 February 2021].

[10]"MTG-QBH | Music Technology Group", Mtg.upf.edu, 2021. [Online]. Available: http://mtg.upf.edu/download/datasets/MTG-QBH. [Accessed: 24- Jun- 2021].

[11]"2020:Query by Singing/Humming - MIREX Wiki", Music-ir.org, 2021. [Online]. Available: https://www.music-ir.org/mirex/wiki/2020:Query_by_Singing/Humming. [Accessed: 24- Jun- 2021].

[12]"Fast and Efficient Pitch Detection: Bitstream Autocorrelation", Cycfi.com, 2021. [Online]. Available: https://www.cycfi.com/2018/03/fast-and-efficient-pitch-detection-bitstream-autocorrelation/. [Accessed: 25- Jul- 2021].

[13]"Autocorrelation - Wikipedia", En.wikipedia.org, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Autocorrelation. [Accessed: 25- Jul- 2021].

[14] G. Nam and K. Park, "Multi-Classifier Based on a Query-by-Singing/Humming System", Symmetry, vol. 7, no. 2, pp. 994-1016, 2015. Available: 10.3390/sym7020994.