Anomalous Note Change Detection of Unknown Monophonic Melodies

W. L. D. Fernando

Anomalous Note Change Detection of Unknown Monophonic Melodies

W. L. D. Fernando
Index No: 14000377

Supervisor: Dr. K. L. Jayaratne

January 2019

Submitted in partial fulfillment of the requirements of the
B.Sc in Computer Science Final Year Project (SCS4124)

Declaration

I certify that this dissertation does not incorporate, without acknowledgment, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, be made available for photocopying and for inter-library loans, and for the title and abstract to be made available to outside organizations.

Candidate Name: W. L. D. Fernando

_____

Signature of Candidate Date:

This is to certify that this dissertation is based on the work of Mr. W. L. D. Fernando under my supervision. The thesis has been prepared according to the format stipulated and is of an acceptable standard.

Supervisor Name: Dr. K. L. Jayaratne

_____

Signature of Supervisor Date:

Abstract

Music melody is a sequence of music notes which are arranged in a musically satisfied manner. There should be a scale for each melody performance. A particular music scale has a set of 'related notes', and therefore, a melody consists of a set of scale satisfied notes. However, when 'the scale un-related notes' occur in the note sequence, it will provide less pleasant melodies. These 'uncommon' notes are the situations which refer as the 'anomalous note' in a particular melody. It is a major concern in the context of 'melody evaluation'.

In this study, a novel approach is proposed to detect anomalous notes changes of musical melodies. The proposed model is focused on to have two phases. Within the first phase, melodies are processed to have their pitch estimations. The steps of feature extraction and fundamental frequency estimations are involved in the first phase. After the pitch estimation, a note event model is employed with the application of Long Short Term Memory (LSTM) neural network for the detection of 'anomalous note changes' regarding the estimated pitch values of sampled melody signals.

The proposed model is designed to focus on unknown monophonic melodies, which is the simplest type of musical texture and was able to have 69% overall accuracy for the used dataset.

Preface

An anomalous note change detection approach for unknown monophonic melodies is provided here in this dissertation. This approach is designed to have an accurate way of anomalous note change detection with two main phases which involve signal processing and machine learning aspects together. The overall analysis of this provided design is entirely my own work. It uses the theories and algorithms in signal processing which were found there in several places in the literature. I used a previously collected data set with the permission of that research group and I already collected some new data samples to that existing data set as well. An approach for anomalous note change detection of music melodies has not provided as a combination of several musicological approaches in any other work in the domain of musical content analysis.

Acknowledgement

I take this opportunity to express my sincere gratitude to my supervisor, Dr. K.L. Jayaratne, senior lecturer of University of Colombo School of Computing, for providing me with continuous guidance and supervision throughout the year.

I would also like to give my sincere gratitude to Dr. M.I.E. Wickramasinghe, lecturer of University of Colombo School of Computing and Mr. Malik Silva, lecturer of University of Colombo School of Computing, for providing valuable feedback and comments on my research proposal evaluation and interim evaluation to improve and enhance my study on this research area. I also take the opportunity to acknowledge the assistance provided by Dr. H.E.M.H.B. Ekanayake as the final year computer science project coordinator.

A very special thanks goes to Mr. Suresh Maliyadde, one of the most outstanding music directors in Sri Lanka, for his valuable time, advice and service provided for me throughout this research.

My deepest gratitude goes to my loving parents for their unconditional support, love and encouragement extended throughout this journey of life. Finally, it is a great pleasure for me to acknowledge the assistance and contribution of all the people who helped me to successfully complete my research project.

Contents

List of Figures

List of Tables

Acronyms

ANN	Artificial Neural Network
DNN	Deep Neural Network
HMM	Hidden Markov Model
LSTM	Long Short Term Memory
RNN	Recurrent Neural Network
STFT	Short Term Fourier Transform
SVM	Support Vector Machine

Chapter 1

Introduction

Music is a creative way to express the emotions of human beings as a combination of vocal and instrumental sounds. It is a type of language which is common for everyone in the world. People are tending to try new music creations all over the world. Today, it becomes popular with the usage of modern technologies.

Everyone has tried to hum a song even without the domain knowledge of music. Also, some people are trying to play musical instruments without prior knowledge of music. They may try their own new experimental creations as well. For these situations, it is necessary to have some sort of evaluation process for their new melodies. To address this, we can find several applications which provide the ability for users to have their own musical creations. Automatic melody evaluation is an important fact to be considered in such applications, especially for musically untrained users.

When we try to have an automatic singing skill evaluation or any other kind of automatic music melody evaluation, we must perform various technical approaches to classify the input melodies according to their performance. However, 'the automatic melody evaluation' process depends on a set of decidable facts such as the features which we are using for the evaluation process, the noisiness of the input melodies, anomalous note changes in the singing/playing melodies, fold responses of melodies, etc.

The 'effect of anomalous note changes' is a major concern for melody evalua-

tion. Thus, this approach proposes an automatic way to determine anomalous note changes in the context of unknown monophonic melodies which suggests an improved way for melody evaluations.

1.1 Background to the Research

Human listeners are already able to identify musical sounds easily, even without any pre-known formal domain knowledge of music. But, only musically trained users are sensible enough to identify some out-pitch singing, variations, and other harmonies during any kind of singing. When the basic features of acoustic signals are considered, such as pitch, loudness, duration, and timbre, they are easily perceived and further processed to constitute musical concepts such as melody, rhythm, and harmony.

Melody evaluation is a process of figuring out a set of feasible features and making decisions about them according to the structure of the music. According to the focused set of features, evaluation results can be classified as 'good' or 'poor'. For the process of feature extraction, we have a large collection of music features such as pitch, vibrato, tempo, etc. But it is challenging to handle the interaction of these music information facets for such kind of evaluation of a melody. In the literature, they have already encountered this issue as 'The Multifaceted Challenge' [1].

In order to come up with a final music creation as a complete song, it should be a combination of at least a single vocal melody and some instrumental melodies. The sound which consists of a particular pitch and a duration is referred to as a musical 'note' whereas the sequence of notes arranged in a musically satisfying manner is referred to as the 'melody'. When a particular melody is consisting of only one singing voice (single vocal melody) or only one instrumental melody (single instrumental melody), it is called a 'Monophonic Melody'. When a melody does not have its beat arrangement or previously estimated facts, it is called an 'Unknown Melody'.

Notes can be identified as the fundamental units in a music melody. Each note has its own duration and a pitch, which is an identifiable fact that refers to a single sound in the complete range of sound. Simply, these set of notes consists of different keys which can be named as C, D, E, F, G, A and B. Apart from these major keynotes there are five other note keys (C♯/D♭,D♯/E♭,F♯/G♭,G♯/A♭,A♯/B♭), which called as sharp (♯) notes or flat (♭) notes. These twelve notes together perform the chromatic scale of the music. Each and every melody should include a set of these distinct notes in different octaves, according to the scale of the melody.

While a process of performance evaluation of any melody has encountered, it will occur some patterns or any other rhythm of notes within that particular melody. But when non-musicians/musically untrained users are trying to have their own melodies, they may have some unnecessary note changes in their performances. However, in some occasions, even the musically trained users may have some ambiguous note events in a melody which lead to having anomalous note changes. Therefore, there may be some unclear/unpleasant states in that particular melody. However, identifying these kinds of effects is not a simple task.

1.2 Research Problem and Research Questions

1.2.1 Problem Statement

As mentioned in the previous section, music melody is a collection of notes which are arranged in a musically satisfying manner. Therefore each melody should have a pattern of notes according to its pitch, chord and the scale. Thus, within this note arranging, there may be some situations which are uncommon and ambiguous regarding the pitch, chord and the scale of that particularly considered melody.

If we consider melodies which include those type of anomalous notes, there is a probability of falsifiable results in melody evaluations. Also in melody transcription, these ambiguous situations lead to some false note key detections. The ambiguity occurs because of the harmonic and variations within the melodies. Note that all ambiguous situations in a note sequence can not consider as anomalies because

of the involvement of harmonies and variations. Therefore identifying anomalous notes changes and their effects, involves a valuable step in melody evaluations in the context of music content analysis.

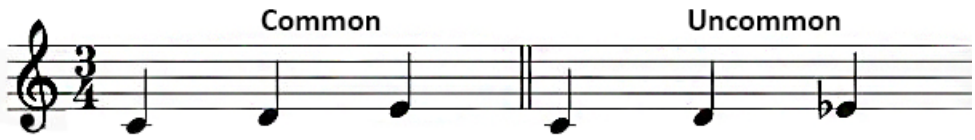


Figure 1.1: Example of a common and uncommon three-note sequence in C major

As an example, in Figure 1.1, two identical three-note sequences are represented in the C major scale. The first three-note sequence represents the notes C,D, and E while the second three-note sequence represents the note sequence as C,D, and E♭. The difference between these two sequences is just a semitone in the last note. But within the C major scale, E♭ note is not an common related note. Thus, this situation can be considered as an anomalous note event in the context of considered music scale.

1.2.2 Research Aim

This research is mainly focusing into conduct an anomaly detection approach for unknown monophonic melodies, by identifying the effects of uncommon note changes in melody performances evaluation.

1.2.3 Research Questions

How to detect anomalous note changes that affect the overall performance evaluation of melodies?

- How do anomalous note changes affect the overall performance of a melody?
- How can one define the anomalous note changes in the sense of uncommon note occurrences in a melody?
- What are the approaches that can be used to detect anomalous note changes?

1.2.4 Research Objectives

- Identify the limitations of overall performances among unknown monophonic melody evaluations, regarding anomalous note events
- Identify the note events which can be identified as anomalies within the note sequence of a melody
- Apply appropriate technical approaches to determine the anomalous note events in unknown monophonic melodies

1.3 Justification for the research

Research approaches for anomaly detection in music melodies are not a well-defined topic in the literature. In a similar content, what can be found in the literature is melody transcription. Melody transcription is the process of converting an audio recording into a musical score or a similar representation. In literature, there are several studies in melody transcription for the different types of melodies. However, because of the hardness of accurate note detection, it has also become an emerging research area. Identifying anomalous note changes is also an important fact due to the note estimations with score information in melody transcription.

As there are limited forms of information that may be available with the extracted features, it is hard for human listeners to analyze an audio melody performance. Thus, an automatic way of performance evaluation gives a value to the musically interested users. If there is a well-established way to detect anomalous note changes in a melody, then there is a chance to enhance the melody evaluation process to have a more reliable classification.

Therefore, conducting an approach for anomalous note change detection in unknown monophonic melodies is an important research area in the domain of computational music content analysis. As a computer science related approach, this proposed methodology will involve different computer science aspects such as machine learning, signal processing, etc. with several theories, models and algorithms.

1.4 Methodology

This study is following a quantitative research methodology, which is mainly going to conduct in the domain of audio signal processing. Within the scope of this research project, vocal melodies and single sound instrumental melodies are considered as unknown monophonic melodies, which involves signal processing techniques.

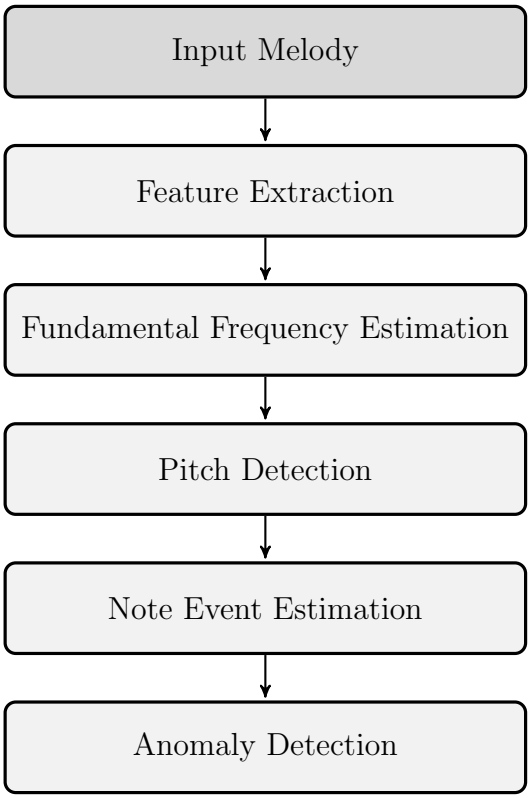


Figure 1.2: Main steps of the approach

As the most fundamental stage of this research, the feature extraction phase is involving the quantitative data analysis of a particular set of signals. Signals of audio melodies are processed to determine the fundamental frequency and the pitch estimations for a particular melody. This stage involves a set of well-defined signal processing techniques and the extracted quantitative features are then processed to introduce to the following stages for further enhancement.

Within the note detection stage, the decisions have been taken regarding the anomalies of note events. This phase is mainly focusing on predicting notes by figuring out the scale related patterns of melodies. This prediction model is used

to detect the anomalies in the sequence of notes in the melody which has been exposed using the previously enhanced input signals and extracted data. Therefore, a neural network based method has proposed in the process of anomaly detection.

As the evaluation of this research, this uses a dataset of monophonic melodies and the decisions have been taken to identify the more generic anomaly types of melodies. Generally, the proposed approach is mainly following the steps that stated in Figure 1.2 and the research has designed to follow these basic steps through different phases. Details of the entire research design is described in Chapter 3.

1.5 Delimitation of Scope

Since this anomaly detection process is a combination of several music content analysis tasks, this methodology have limitations to be considered as follow.

- Only unknown monophonic melodies (majority of vocal melodies) are considered
  - Monophonic melodies are the simplest texture among three different textures in music[1]. Though, analyzing unknown monophonic melodies as the simplest texture of music is the initial step of conducting an anomaly detection approach in any kind of melody.
- Only the major scale melodies are considered
  - All the types of scales in music can be divided into two main groups as 'major scales' and 'minor scales'. However there are intersections between some major and minor scales as well. For an instance, both the C major scale and A minor scale have a same set of related notes. Therefore, only the major scale melodies are considered into this approach as they do not involve complex note combinations within melodies.

[1]Three textures in music are Monophonic, Homophonic and Polyphonic.

- A pitch-based melody evaluation has applied with limited features
  - In the context of anomalous note event detection, it only needs to analyze the pitch estimations in order to identify note occurrences. Though, pitch-based evaluations are the only needed type of approach it should follow. Also among pitch-based melody evaluations, it needs to figure out only the needed features related to the note event identification.
- A predefined rule set has applied to define anomalies
  - Anomalous note events are defined based on their relatedness to the scale of the melody. Therefore decision about the anomalies should be taken under the scale of particular melody. For this, a rule set is needed to be followed for each type of scale.
- Only some parts of 'Melody Transcription' (not the whole process) has been processed through the design
  - Melody transcription involves a complete process of converting melodies in musical notes (score information) regarding all the pitch estimations, temporal estimations and timing. The process of anomaly detection only needs to follow the pitch estimations without any other temporal and timing related information.

1.6 Definitions

When a particular melody is consisting of only one singing voice (single vocal melody) or only one instrumental sound (single instrumental melody), it is called a 'Monophonic Melody'. Thus in this study, a single type of vocals and a single type of instrumental melodies are considered under the term 'Unknown Monophonic Melodies'.

Through out this dissertation, an anomalous note event is defined as a scale unrelated note occurrence of a melody.

1.7 Outline of the Dissertation

This introduction chapter was organized to give an overall idea to the research domain and the remainder of this dissertation has been organized in the following ways.

Chapter 2 lays out the literature review for the background study and related work with outlined theories and other concepts. The intention of this review is to identify the gaps among the related works done in several places.

Chapter 3 describes the overall research design and further explanations. Components of the research design has analyzed and the justifications for the selected methodologies and designs has also explained in Chapter 3.

Chapter 4 includes the implementation details of the desired design with the discussions of the used technologies and algorithms.

Chapter 5 presents the results for the evaluation process of the desired design. These results has analyzed with the details of the evaluation criteria for the better explanations.

Finally, Chapter 6 concludes the overall work done through this research with a summary of the work carried out. It follows the suggestions to the future work for further enhancements of the proposed approach.

1.8 Conclusion

This chapter analyzed the foundations for the dissertation. It introduced the research problem and the domain, research questions and hypotheses. Then the research was justified and the definitions were presented. The methodology was briefly described and justified with the details. The scope and the limitations were given regarding the problem domain. Finally, the outline of this dissertation was presented to the reader.

Chapter 2

Literature Review

Anomaly detection of unknown monophonic melody is actually a combination of several computational music-content analysis such as singing skill evaluation, melody transcription, music note detection and computational anomaly detection methodologies. In the context of unknown melodies, it also involves a way to identify the anomalies of melody note sequences. Therefore, the literature review was conducted in several stages in the computational music-content analysis.

Since the main aim of this proposed approach is to detect the anomalous note changes in music melodies, the anomaly detection approaches should be combined with appropriate note detection approaches. In the literature, there are no such applicable anomaly detection techniques to follow with the music melodies.

Instead of that, there are several probabilistic models for note events which are defined to automatically identify the non-precise and unambiguous note events during a melody. Actually, that seems to have some replacement for bad occurring note events.

But in the context of anomaly detection, there are several methodologies to follow with computational analyzing. In real computer applications, anomaly detection is considered as a science. Depending on the type of analyzing data, there is the opportunity to follow the suitable techniques within this large collection of methods [2]. Therefore, anomaly detection of music melody should be a combination of the

applicable anomaly detection approaches with the note event detection approaches.

Because of, it is hard to find that kind of combined methods in the literature, this review has followed in three categories (singing skill evaluation, note detection and melody transcription, and anomaly detection) to identify the most appropriate approaches and their applicability for the connectivity of each phase.

Section 2.1 of this chapter discusses related theoretical aspects of the musical terminology according to the domain of this research. Section 2.2 reviews the singing skill evaluation approaches in-order to identify the melody evaluation process. Section 2.3 and 2.4 reviews the methods used in note event detection and anomaly detection respectively. In Section 2.5, the applications of anomaly detection analysis are surveyed.

2.1 Musical Terminology

Music notes are the most fundamental units in music. A single note represents the sound which consists of a particular pitch and a duration. Pitch is the position of a single sound in the complete range of sound. As mentioned in the background section of previous chapter, there are mainly seven major key notes in music which can be named as C, D, E, F, G, A and B.

The pitch difference between two notes is referred to as an interval. These intervals with pitch frequency ratio 2:1 are called as octaves. Each octave is divided into twelve notes in western music and it is referred to as the 'Chromatic Scale'. This twelve-note pattern defines the frequency of each note pitch using the equation 2.1.

$$f = 2^{x/12} f_{base} \tag{2.1}$$

In this equation, f is the frequency of the note pitch where the f_base is the frequency of a reference note. x is an integer offset from that reference note. Usually, f_base value is used as 440 Hz which denotes the frequency value of note 'A' in 4th octave [3, 4].

In here, x is the number of half steps (semitones) away from this reference note. If the required note at a higher note, then x is positive. If the required note on a lower note, then x is negative.

The intervals between the two adjacent note pitches are measured in semitones [5]. This chromatic scale can be noted with sharp signs when ascending and flat signs when descending. Sharp (♯) raises the sound by one semitone and flat (♭) lowers the sound by one semitone. Figure 2.1 shows the twelve notes in the chromatic scale.

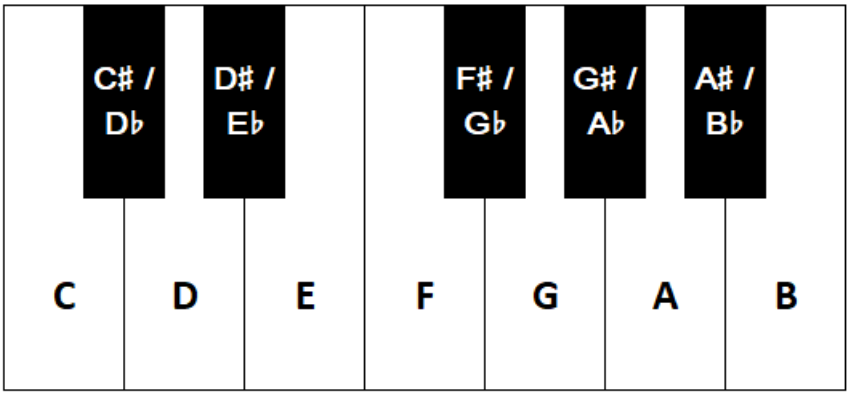


Figure 2.1: Chromatic Scale of Western Music (A Single Octave)

For an example, suppose the frequency value of Note 'C' in 5^{th} octave is needed to be calculated. As it says, this Note C is in the 5^{th} octave. But the Reference Note, f_base in Eq. 2.1 is in the 4^{th} octave. Therefore, this Note C is located from three semitones higher than the reference note. (Note A in 4^{th} octave + 2 semitones = Note B in 4^{th} octave. Difference between Note B and Note C in consecutive two octaves is also a semitone. Therefore, Note B in 4^{th} octave + 1 semitone = Note C in 5^{th} octave). Therefore, the x value in Eq. 2.1 is assigned as +3. Then the frequency of Note C in 5^{th} octave can be obtained as $2^{3/12} * 440 = 523.3$ Hz.

Series of these distinct notes performs a 'Scale' in music. These scales are mainly divided into two categories as Major Scales and Minor Scales. Each major/minor scale consists with a particular set of notes. Therefore, each melody should follow

single key scale with particular note patterns in a musically satisfying manner. There are 12 major scales correspond with one of the twelve notes in the chromatic scale and each scale consist seven notes. The difference of the third and fourth notes for each of these scale is a semitone and the difference between all the other two adjacent notes is a tone. Table 2.1 shows the main seven major scales with the relevant notes.

Table 2.1: Seven Major Scales in Music with Notes

Scale	Note 1	Note 2	Note 3	Note 4	Note 5	Note 6	Note 7
C Major	C	D	E	F	G	A	B
D Major	D	E	F♯	G	A	B	C♯
E Major	E	F♯	G♯	A	B	C♯	D♯
F Major	F	G	A	B♭	C	D	E
G Major	G	A	B	C	D	E	F♯
A Major	A	B	C♯	D	E	F♯	G♯
B Major	B	C♯	D♯	E	F♯	G♯	B♭

2.2 Singing Skill Evaluation Approaches

Several melody evaluation approaches are there to be analyzed as the background domain of melody evaluation. Therefore, some studies on singing skill evaluations are analyzed as the first phase.

According to the used features, we can also classify our evaluation of melodies into different categories. In previous literature, they have used pitch based approaches and tempo based approaches to measure the singing skill. Most of them have used pitch interval accuracy, vibrato, and tempo as the features for their methods.

2.2.1 Pitch Based Singing Skill Evaluation Approaches

Pitch-based techniques are the most common singing skill evaluation approach in literature. Several studies were carried out for the pitch based vocal melody evaluations. Mauch et. al. [6] proposed an approach to evaluate the accuracy of pitch

in playing or singing music. They were targeting unaccompanied solo singing and they have come up with two metrics to measure the accuracy and drift as interval error and pitch error. Their whole approach has provided some insights into intonation and they have come up with a simple model of reference pitch memory with a study on intonation and intonation drift in unaccompanied singing. They were able to provide a deeper analysis of pitch and its behavior in a melody, with a good evaluation process.

Żwan [7] has done an artificial neural network based method to determine how the quality of a singing voice can be recognized automatically. A quality factor is used as a subjective measure of whether the voice belongs to an amateur or to a professional singer. Żwan has used 1440 singing voice sounds and parameterized them using 331 parameters of different features. A feed-forward neural network was employed for the automatic recognition of the singing voice. Those who are able to maintain a stable vibrato are considered as the professional singers. He has done a comparison between experts' judgment and the performed automatic results to have a clear evaluation criterion. This study has provided some important facts about the ANN based automatic melody recognition.

Nakano and Goto [8] has suggested an automated approach for singing skill evaluation. Pitch interval accuracy and vibrato are used as an acoustic feature for their approach. Since those features are independent of specific characteristics of the singer and the melody, they have aimed to explore a way of automatic evaluation of singing skills without the score information of melodies. They have defined the semitone stability using the pitch interval accuracy. Short Term Fourier Transform (STFT) has used to measure the vibrato and Support Vector Machine (SVM) has used for the final binary classification. Since score information is also unavailable in unknown melodies, this pitch based approach is compatible with the requirements.

Thus, there are several features that can be used for the evaluation of melody performances. But when we try to estimate some note events of a melody, pitch detection is an essential step to follow.

2.2.2 Tempo Based Singing Skill Evaluation Approaches

Apart from the pitch based techniques for melody evaluation, there are several tempo based approaches as well. Most of the tempo based evaluations are following onset detection approaches in the current literature. However, focusing on the tempo of the unknown melodies, it needs an additional attention rather than the pitch-based approaches for unknown melodies.

Among tempo based singing skill evaluations, Toh et al [9]. have used Gaussian Mixture Model Indicators for the classification of audio features of vocal melodies. They have captured the complexity of singing note onsets within a supervised learning approach.

Alonso et al. [10] presented a method for tempo extraction, based on spectral energy flux. They proposed an onset detection based algorithm with the steps of 'extraction of onset by front-end analysis', 'periodicity detection black' and 'temporal estimation of beat locations'.

The onset signing notes estimation processes in both [9, 10] approaches are useful for the note event handling process. The tempo based approaches are not capable of addressing the required process of feature extraction regarding monophonic melodies.

2.3 Note Detection Approaches

Melody transcription is the process of converting the audio melodies into the sequence of notes. Simply, this is an automatic way of having a more precise score information for a vocal melody. Score information includes all the temporal estimations and timing estimations between each and every note pair. It tries to convert the melody signal into note keys by rounding pitch estimations to the relevant MIDI standard note numbers. In order to do that, there should be a technique to identify and detect the notes in a melody.

Despite the difficulty, vocal melody transcription is still an important topic in music information retrieval field, as it enables computers to extract the information carried by singing and facilitates the direct musical interaction between humans and computers [11]. Though as in literature, generating a more accurate and precise note sequence is still a hard task. But there are some well-defined methods that they have tried to achieve this goal with the use of signal processing and machine learning theories.

Ryynänen [3] has done several approaches for melody transcription for different kind of melodies such as polyphonic melodies, homophonic melodies, and monophonic melodies. Among those, he has proposed a model based method for automatic note transcription of monophonic melodies based on two probabilistic models. First one is a note event model which is based on Hidden Markov Models. It has been used to represent note candidates. As the second model, a musicological model has used for the key estimation and likelihoods of different note sequences to examine the transitions between note candidates.

He has used a Hidden Markov Model (HMM) to limit the dynamics of their pitch estimates. Though, this approach will be helpful to identify the process of generating note sequences. In this study, there is an attempt to eliminate some kind of predefined ambiguous notes during their transcription model. With the idea based on Ryynänen's method of using HMM, Sebastian Bock and Markus Schedl [12] have proposed another way of piano note transcription with Recurrent Neural Network (RNN). They have targeted the polyphonic melodies in their approach. But it is based on a Recurrent Neural Network to simultaneously detect the onsets and the pitch values of the notes from spectral features. The memory units in RNN are used in a bidirectional neural network to model the context of the notes.

Mauch et al. [13] has proposed a software tool (called as 'Tony Software') for the interactive annotation of melodies from monophonic audio recordings. In their approach, they have done a melody transcription based on their own pitch detection and note detection algorithms. They have employed two valuable post-processing

steps to increase the accuracy of the system. In the first step, an amplitude-based onset segmentation has used to separate consecutive notes of similar pitches. In the second step, they have used a threshold value to discard the unnecessary note events. This is a special kind of technique to eliminate exceptional notes which can be considered as anomalous notes.

Fundamental frequency estimation and pitch detection are the most fundamental steps in note detection. This is the main step that followed by several other steps to achieve the final detection of notes. This kind of pitch detection is involved in each and every computational music-content analysis technique. Therefore each melody transcription approach has conducted a special considerable analysis on pitch detection.

Among them, YIN algorithm has proposed a more precise way to detect the corresponding pitch for melody notes [14]. YIN includes a number of mathematical formulas for the estimation of the correct pitch. In aforementioned Ryynänen's proposed method [3], he has analyzed each step of this algorithm and mentioned a conclusion of the algorithm for the better performances. Also in 'Tony' software [13], they have used their own defined algorithm called as pYIN [15], which has defined based on the YIN algorithm.

In the proposed methods in [16], Antti Laaksonen has defined his problem of automatic melody transcription based on the chord transcription. But in the sense of note model generation, he has given a clear idea about key estimation, pattern matching and scoring function with the acceptable steps which can be applied for the estimation of notes in a melody.

A change detection approach in multi-voice music has done to evaluate the relationship between primitive and scheme-driven grouping by comparing the ability of different listeners to detect single note changes in 3-voice musical compositions [17]. They have done their evaluation by using both musicians and non-musicians to detect single note changes with both polyphonic and homophonic stimuli.

Bishara and Radvansky [18] has done an approach for the detection and tracing of melodic key change. Adjusting to key changes in music is vital to a listener's understanding of it, although it is unclear when this adjustment occurs. Both the number of notes after the change and the change distance were manipulated and they have tried to adapt new keys for the detected changes of keys. This study can be analyzed for the note changes identification.

2.4 Anomaly Detection Approaches

In the previous melody transcription and note event estimation approaches, researchers have involved in some aspects of machine learning algorithms. As mentioned before, anomaly detection is a well-established research area in different computer science aspects. Therefore, as automatic ways of detecting anomalies, different anomaly detection approaches have been analyzed to identify their applicability for the music melodies.

As in [12], there are some combinations of common anomaly detection methods with the music note events handling approaches. Thus, this study was conducted to analyze the features and their applicability of common anomaly detection methods which can be used in the domain of melody note estimation.

In [19], a time series anomaly detection approach has been done to utilize machine learning and statistical approaches to classify anomalous drops in periodic, but noisy, traffic patterns without a large body of labeled examples. As the first step of their approach, they have used TensorFlow to train their various models including DNNs, RNNs, and LSTMs to perform regression and predict the expected value in the time series. As the second step, they have created anomaly detection rules that compared the actual values to predicted values. At the later part, they have provided the confusion matrices for each model that they were investigated. In aforementioned automatic piano transcription [12], they have already used these kinds of LSTM application to detect the notes in a more accurate way.

In the study of Collective Anomaly in [20], a Long Short Term Memory (LSTM) has employed to identify the anomalies in collective data set. They have provided descriptive details in each step to analyze the process of LSTM in the sense of collective anomalies.

Another study on anomaly detection in time series has done based on the long short term memory networks in [21]. In their approach, the network has trained on non-anomalous data and used as a predictor over a number of time steps. They have proposed a stacked LSTM based prediction model for the experiments and within that proposed model they were able to detect anomalies using the prediction error distribution.

2.5 Analysis

This analyzing process was able to figure out the important facts of music notes anomalies such as what are the possible note patterns to have ambiguities, what type of uncommon changes are there, how the note distances are going to have uncommon changes etc. As an example, we can find that there is a low probability to have flat notes in the scale of C major. Therefore, if a melody has that kind of note patterns, it is a considerable point in the design phase. Thus, that kind of patterns and estimations are encountered in parallel to other analyzing stages. Though, the anomalies within note events are defined as any type of scale unrelated note changes.

However, in all note detection/melody transcription approaches, note event estimation is considered as the most important phase. Note events are detected based on the estimated pitch values. But there is no standard process to identify exceptional note events among the detected note event set. This is the biggest gap in note detection phases in literature. If there is a procedure to identify these exceptional/uncommon note events, it will be useful in every melody transcription or note detection technique.

Also, in many note detection approaches, they have followed neural network based methodologies to improve the estimation accuracy of the note event. In the same context, they have already tried sequential data analysis for those note events as well. Recurrent Neural Network based methods are the most popular type of method that follows sequential data analysis with more accurate results. Even in some music content analysis [12, 20, 21], they have tried with these Recurrent Neural Network methods. Thus, these background details provide valuable information about the applicability of these neural network based methods for anomaly detection in note event series.

2.6 Chapter Summary

An overview of the concepts behind the problem domain was discussed in this chapter. Several studies were conducted for background analysis in singing skill evaluation, note estimation and melody transcription, anomaly detection methods and their applicability etc. Among them, some fundamental concepts in the problem domain and needed steps are identified. The research design has defined based on these identified facts and the details are discussed in Chapter 3.

Chapter 3

Design

This chapter elaborates the overall design of the solution to the pointed research problem. This proposed design is based on the identified facts in the literature review phase. Research design and the used methods have described with the details. Also, the justifications for the selected methodologies are given in this chapter as well.

3.1 Assumptions for the Design

In music melodies, it is possible to have scale related/unrelated variations and harmonic parts within melodies. Especially, harmonization can be found easily in singing. But when it comes to the context of monophonic melodies, the probability of having that kind of situations is very lower than the other textures of the music. Though, this anomalous note change detection is designed under the assumption that *the unknown monophonic melodies do not contain any harmonic/variation point within a single type of sound*. The overall design is proposed to handle only the monophonic melodies under the above assumption.

3.2 Research Design

The overall design of the proposed approach is composed to follow several steps. These noted steps are addressed throughout the current literature review as a combination of different areas. This approach should follow a flow of steps which is more

similar to the melody transcription process. But in advance, required steps should follow a set of well-defined techniques to have a more accurate way in anomaly detection. It is easy to observe that the required process does not need the entire melody transcription here.

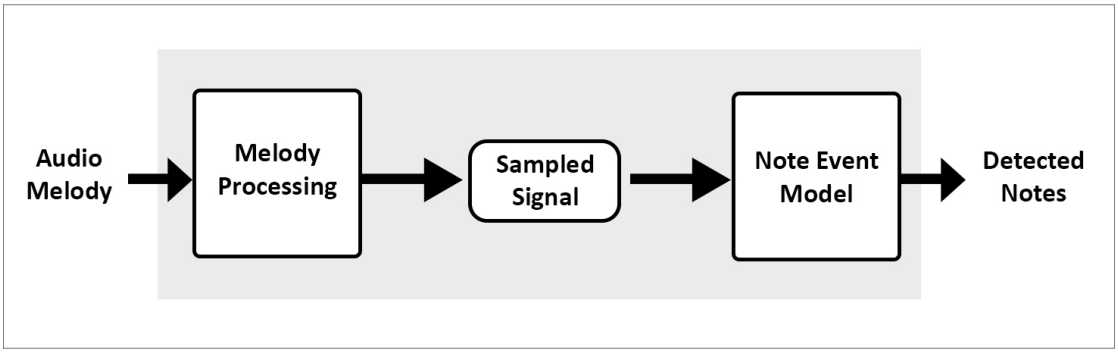


Figure 3.1: High-Level Research Design

Therefore, this research is designed to have a combined process of different steps. Each step has identified as a special part of a model. Figure 3.1 shows the interconnection of these identified phases as a high level research design. Melody processing consists of several steps to follow and the audio melodies are sampled into several frames. The note event model has employed with a Long Short Term Memory (LSTM), which is a special kind of Recurrent Neural Network (RNN). The note event model is the most important part of this anomaly detection approach.

3.3 Melody Processing Steps

The main aim of this study is to have a way to determine the note sequence anomalies of melodies. Typically, each note is directly mapped to have a pitch value. Though, as the first phase, music melodies should process to identify the fundamental estimations for their pitch values. This involves several audio processing techniques and methodologies to follow. Figure 3.2 shows the basic steps that need to follow, as the melody processing steps.

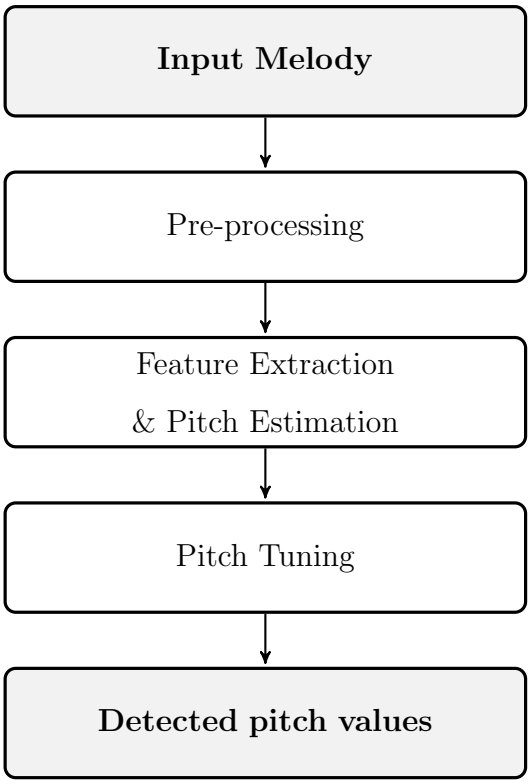


Figure 3.2: Melody Processing Steps

3.3.1 Preprocessing

As the unknown melodies are involved, recorded melodies already contain noisiness and some points in silence. Those silent points can be considered as the time gaps between some notes in the melody. As this study only considers the note changing events, these silence points make no sense in the context. Therefore, a preprocessing technique is used here to minimize the captured unwanted points in recorded vocal melodies.

As an important fact of the needed preprocessing, it is not going to cut off any part which has at least a single point of sound. Each sound point may provide valuable information regarding a minor note change in the melody. There is a chance to have unnecessary out-pitch estimations for this kind of minor estimations as well. This pointed issue has already handled under the tuning phase after the pitch estimation.

3.3.2 Feature Extraction and Pitch Estimation

Pitch detection is the valuable steps of melody processing. This step is going to use the facts of vocal melodies to have some meaningful estimations. Therefore, 'pitch' is the main feature that is going to be extracted from audio recordings in order to estimate the pitch intervals. There are two types of pitch detections as 'Time Domain Pitch Detection' and 'Frequency Domain Pitch Detection', whereas this study follows the frequency domain pitch detection technique. Pitch value is denoted as a frequency value for each point of melody (measured by Hertz (Hz)). YIN algorithm is chosen to use in this study for pitch detection, as originally proposed by Cheveigne and Kawahara in [14].

Typically, in the context of music, pitch values are defined based on the key distribution in a piano. A standard modern piano has 52 white keys and 36 black keys which distribute across seven octaves and three extra keys. As mentioned in the literature review, an octave consists of twelve keys as seven white keys for natural notes and five black keys for sharp/flat notes. Piano keys are named to represent their music notations with their location of the octave. For an instance, the 'A' note key in the 4th octave is named as the key 'A4' in a piano. Thus, a standard modern piano has named its keys from A0 to C8 as in Figure 3.3. Middle 'C' is denoted as 'C4' in this format. When playing the piano, this 4th octave is the most used set of keys. This representation is referred to as the 'Pitch Note Representation' in further discussions.

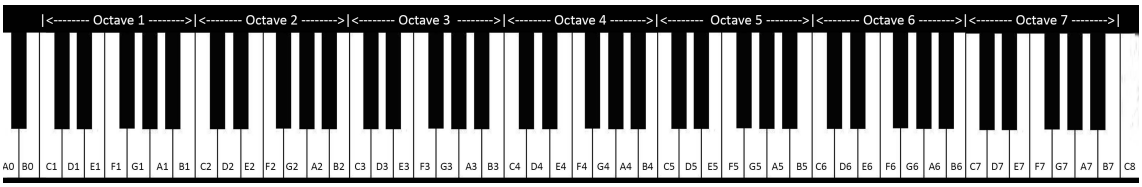


Figure 3.3: Pitch Notation Representation of a Piano

Each and every pitch notation in the above representation has its own unique frequency value. These pitch values are used to estimate the corresponding note for each time lag in a melody. In literature, all melody transcription techniques have followed this standard pitch notation values to obtain the notations even from a

simple melody. These values are very useful in MIDI representation of melody transcription. Each of these note has its own MIDI number in a standard way. But for this study, MIDI numbers are not needed, and hence only the estimated pitch values are used for further enhancement. Standard fixed pitch values for $4th$ octave notes are presented in the Table 3.1 (See Appendix A for frequency values of other octaves). However, the pitch values for human singing can differ from these values within small range [22]. But they can be mapped to the standard representation.

Table 3.1: Frequency values for Notes in 4th Octave

Note	**Frequency (Hz)**
C4	261.63
C♯4/D♭4	277.18
D4	293.66
D♯4/E♭4	311.13
E4	329.63
F4	349.23
F♯4/G♭4	369.99
G4	392.00
G♯4/A♭4	415.30
A4	440.00
A♯4/B♭4	466.16
B4	493.88

As described in the literature review, there are different ways to estimate the pitch values of audios. YIN pitch estimation is a very precise way to estimate the pitch values for any kind of melody. Within YIN, audio signals are split into overlapping frames, and the fundamental frequency of each considering point is represented by the extracted pitch under a predefined interval value. YIN includes several features which are very useful for an accurate estimation of relevant pitch values. At different stages, it uses more robustness ways to determine correct values among a set of estimations. Because of the robustness and its computational easiness, YIN algorithm was selected in this stage for the pitch estimation of audio melodies.

YIN mainly involves five main steps to follow within its main functioning procedures. Given that s is a discrete signal of input melody, 'N' is defined as the frame length and 'k' is defined as the constant threshold value for YIN. The complete algorithm is formulated as follows:

Algorithm 1: YIN Algorithm

Result: pitch value estimations for s

1 Calculate auto-correlation method

2 Calculate the squared difference function $d(\tau)$,

$$d(\tau) = \sum_{n=0}^{N-1} (s(n) - s(n+\tau))^2 \tag{3.1}$$

3 Calculate the cumulative mean normalized difference function $d^{'}(\tau)$,

$$d^{'}(\tau) = \begin{cases} 1 & \text{when } \tau = 0 \\ d(\tau)/[(1/\tau)\sum_{j=1}^{\tau} d(j)] & \text{otherwise} \end{cases} \tag{3.2}$$

4 Absolute threshold method

5 Parabolic Interpolation

As the first step of YIN, it generates the auto correlation of particular discrete signal by calculating corresponding lag values at time index t, and pre-defined window size. Then it calculates the difference function $d(\tau)$ for a given range of lag values τ of the signal (using Eq. 3.1). Then it formulates into the cumulative mean normalized difference function for the same signal (using Eq. 3.2). Within the absolute threshold method, It finds the smallest value of τ for which a local minimum of $d^{'}(\tau)$ is smaller than the given absolute threshold 'k'. If no such value is found, it finds the global minimum of $d^{'}(\tau)$. As the final step, it uses parabolic interpolation method which interpolates the $d^{'}(\tau)$ function values at $\{\hat{\tau} - 1,\ \hat{\tau},\ \hat{\tau} + 1\}$ where $\hat{\tau}$ is the found lag value in absolute threshold method. It finds the minimum of the polynomial in the range $(\hat{\tau} - 1,\ \hat{\tau},\ \hat{\tau} + 1)$ to obtain an estimate of the fundamental period.

3.3.3 Pitch Tuning

Only musically trained people can sing in the absolutely correct pitch for any kind of melody. Musically untrained people usually have a reference for their singing in order to have good pitching. But, sometimes they may sing between two distinct note pitches. This can happen even with or without a reference singing as well. Though at that time, the estimated pitch values may not have the exact values as associated with the notes. Then it is a problematic situation to handle that kind of estimated values. This is a major issue for long melodies.

Pitch tuning is the process of determining more accurate values for the pitch estimations by applying technical improvements. In this approach, a tuning process is applied as the final step of the YIN implementation. It uses a simple methodology as a part of the parabolic interpolation to obtain better estimation. Figure 3.4 shows a sample pitch estimations before and after the tuning process. Blue dots represent the initial pitch estimations and red dots represent the tuned pitch estimation.

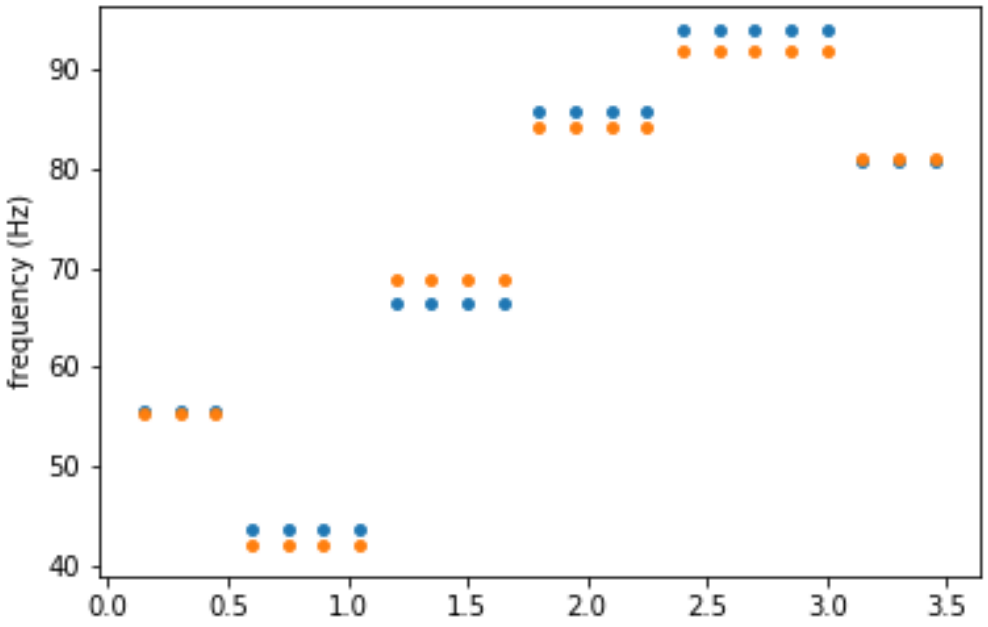


Figure 3.4: Pitch Tuning Example

But even after the tuning process, it is hard to find the correct note pitch for each and every point in time. For an example, a melody can have a value of 268 Hz as the estimated pitch value at one point. But that value is not the exact value for any note in any octave. 268 Hz is in between C4 and C♯4 in the pitch notation representation. As a solution for this, a min-difference technique has applied to figure out the relevant pitch note. Following is the basic difference function which is used here.

$$\Delta x = |x_{est} - x_{ref}| \tag{3.3}$$

x_{est} is the estimated tuned pitch value and x_{ref} is the reference pitch value which is taken for the closest two pitch note values in pitch note representation, one by one at a time. That means the estimated tuned pitch value is taken to pair with every note value in the pitch note representation. So the absolute difference is taken as Δx for each pair and, the minimum value is considered as the correct one. Though, the x_{ref} value is assigned as the pitch note estimation when the absolute difference between that x_{ref} value and the estimated tuned pitch value (x_{est}) is the minimum.

For the above example of 268 Hz tuned estimation, x_{est} is 268. Then the closest two pitch notes, C4 and C♯4 are considered as x_{ref}. Then Δx is calculated as $|268 - 261.63| = 6.37$ for C4. Δx for C♯4 is obtained as $|268 - 277.18| = 9.18$. So the minimum is 6.37 and therefore, the pitch value of C4 is assigned as the correct pitch value. These identified pitch values are used as the inputs to the note event model for note event handling.

3.4 Note Event Model

This study is mainly focusing on the context of note changes in a melody. Hence, the note changes should be identified through the reference pitch values. After identifying the relevant note changes, a machine learning model is employed to determine the anomalous note changes in a particular melody. Thus, this note event model is composed of two major components as shown in Figure 3.5.

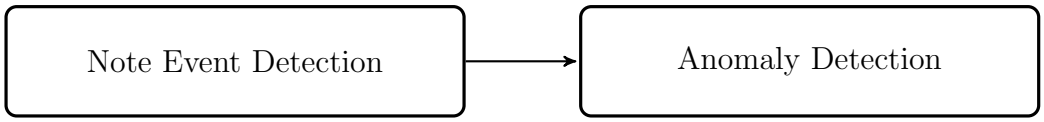


Figure 3.5: Note Event Model Components

3.4.1 Note Event Detection

The term 'Note Event' is a widely used concept in various domains of music content analysis. Simply, a note event is a changing point of any note estimation. But this can have more definitions in different perspectives of music analysis. In melody transcription processes, a note event is estimated with a collection of different attributes of each couple of note estimations. A note event with a melody transcription should include all the relevant details of note estimations such as time interval between two nearby estimation points, reference pitch values of those two estimations etc. The score information in melody transcription process is probably based on these note event identification.

In the context of anomalous note change detection, a note event has identified as a simple change of a note estimation which only based on its pitch value. Time intervals or time gaps between two estimation points are not an important fact in order to identify a note change. Therefore, only the pitch values are considered for this note event estimation. But in advance, note events are defined within a single octave range. All detected reference pitch values are given a score regarding its note by ignoring its octave. This provides two notes in two octaves which have the same key note, as a single score in note event range. For an example, there can be two notes in key A in two different octaves as A3 and A4. But when the note event is encountered, both of these two notes are getting a single score which represents the note A in note event range.

A melody is also consisting with repetitions of the same note in nearby estimations. But in the context of identifying note changes, only the first occurrence of a note estimation is considered, and all the immediate repetitions after that position are ignored. However, if the same note occurs once after some other notes (when it is not a consecutive occurrence), it is considered as another note event.

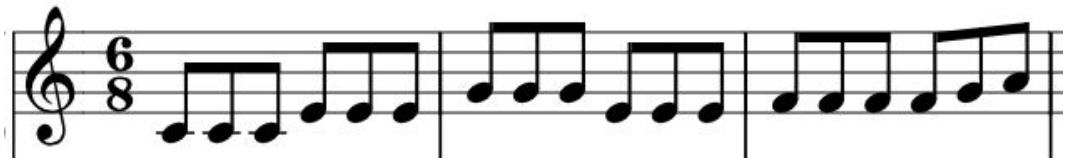
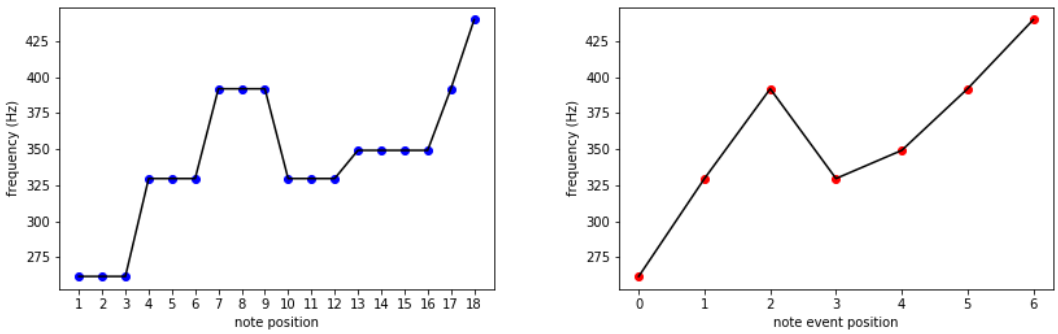


Figure 3.6: Example note sequence for note event detection

For an instance, Figure 3.6 shows a simple note sequence of a melody. The corresponding note sequence for that western music notation is 'C, C, C, E, E, E, G, G, G, E, E, E, F, F, F, F, G, A'. It is easy to see that, there are some repetitions of the same note in nearby positions in the sequence. Note 'C' is repeated three times in first three positions. Note 'E' is repeated three times in second, third and fourth positions and again three times from ninth position. Note 'G' and Note 'F' also have same kind of repetitions within the sequence as well. Therefore, the corresponding note events are identified as 'C, E, G, E, F, G, A' for the above note sequence. These are the note changes that can be identified in that particular note sequence which denoted in the Figure 3.6. Figure 3.7 shows the pitch value distribution and corresponding note event estimations for the above note sequence.

3.4.2 Anomaly Detection

Detected note events are the inputs for the most important phase of the note event model. In this phase, a machine learning model is employed for the final anomaly detection of the detected note changes. As mentioned before, a note change is a



(a) Pitch value distribution (b) Note event distribution

Figure 3.7: Note event estimation example

simple change in the pitch value distribution. These note events are also represented by a pitch value. Therefore, it is considered as a note sequence which does not have the same note (or pitch value) consecutively.

As described in the literature review, each collection of these note events should have a set of unique note patterns according to the scale of the considered melody. However, in the domain of harmonic melodies, this can differ as well. But for monophonic melodies, the set of notes associated with each scale in music can be easily determined. Although, these set of notes for each scale performs a musically satisfying note pattern within a melody. In here, once a scale unrelated note occurs within a pattern, it is considered as an anomaly. These patterns only represent notes, which occurs after a change of pitch value. Hence, these types of note event changes are considered as anomalous note changes in this study.

In here, a machine learning model is focused to have predictions of each note, indicating whether it is an anomaly or not, by considering the note patterns. As found in the literature, there are several time series applications which is designed to identify anomalies in different types of sequential data. Methodologies that they have followed in their time series predictions are also applicable to note sequence analysis as well. As a prediction based analysis, deep learning techniques are identified as most flexible neural network methods. However, thousands of types of specific neural networks are there to have a deep learning analysis of such cases.

Each prediction of an upcoming event within a sequence should have a knowledge of previous events. Traditional neural networks can not handle this kind of procedure for upcoming predictions. Recurrent Neural Network (RNN) is a deep learning neural network which were specifically designed to work with prediction problems in sequences. RNNs have a special kind of architecture which conducts looping of neurons. These are useful in handling future predictions of sequential data.

Long Short Term Memory (LSTM) Model for Anomaly Detection

Long Short Term Memory (LSTM) is a special kind of RNN, which works smarter than the standard RNN. LSTM was introduced by Hochreiter and Schmidhuber in order to overcome issues in RNN topology [23]. Hochreiter has encountered a major issue called as 'vanishing gradient problem' in RNN. In a multi-layer network, gradients for deeper layers are calculated as products of many gradients. When those gradients are small or zero, it will easily vanish. So it becomes very hard to calculate and update them. They have introduced LSTM to overcome the above mentioned issue and to work smarter in sequence analysis problems. As found in the literature, recurrent neural networks with long short term memory are the widely used neural network in the process of anomaly detection. They are able to model temporal contexts due to the use of recurrent connections in the hidden layers which enables blocks to act as a memory cell [24].

However, a LSTM has several advantages over typical RNN topology. LSTMs enable RNNs to remember their inputs over a long period of time. In literature, also several types of time series predictions have used LSTM for their predictions because of this additional feature. Also, it is applicable to the detection of anomalies in melodies. As in [12], it is possible to combine LSTM neural network structure to handle facts in melody evaluation as well.

Therefore, a LSTM has employed in this note event model to figure out the predictions from note event patterns. The model first converts the detected note event sequences as sub-sequences based on the predefined note sequence length. Each predicted note event is then compared with the actual note event in order to determine the scale relationship between those two. Final anomaly prediction is then decided based on that procedure based on the trained LSTM. Decision rules for the above sub-sequence analysis and other relevant details about this model are discussed in the implementation chapter.

3.5 Chapter Summary

The overall design of the proposed approach has two main phases as melody processing phase and note event model. Within the melody processing phase, several steps such as preprocessing, feature extraction and pitch detection, and pitch tuning are employed to have the final pitch value representation for the monophonic melodies. Estimated tuned pitch values are then used to the note event model which is the main part of the research design. After estimating the note events within the note event model, it uses a LSTM for the anomaly detection of those note event patterns. Implementation details of these components are further discussed in Chapter 4.

Chapter 4

Implementation

The implementation of this proposed approach was carried out using technologies and software tools which are more related to signal processing and machine learning. This involves a set of models, which use signal processing algorithms and neural network methodologies. Python 3.6 was used as the main supporting language to design these models, and several collections of libraries are used for the support of signal processing and machine learning aspects.

According to the research design described in the previous chapter, two main phases are considered as two separate models for the implementation. The first part of this implementation includes a collection of signal processing algorithms, whereas the second part includes neural network-based methods.

The first section of this chapter elaborates the supporting languages and libraries which are used for the desired implementation. The second section describes the algorithms and their implementation details, which are used for the melody processing phase. In the third section, it discusses the implementation details of the note event model.

4.1 Tools and Technologies Used

4.1.1 Python

Python is a well-known interpreted programming language which supports a large collection of signal processing techniques. Python 3.6 was used as the version for the implementations of the proposed approach. All the modules in this approach were developed using python with other supporting libraries.

4.1.2 Python Keras

Keras is one of the most popular open source neural network library which is written in python. It provides the capability to run on different platforms such as TensorFlow, CNTK or Theano. Keras provides a set of functions regarding machine learning methodologies. Especially, it provides fast and easy implementation of deep neural network topologies. The proposed LSTM has designed using the Keras library in order to have such deep learning methodologies for the anomaly detection.

4.1.3 Python NumPy

Python NumPy is a powerful python library introduced for scientific computing. It uses a large set of data manipulation methods and provides a large collection of mathematical functions for other libraries such as Liborosa, Keras etc. Also, it provides the ability to use other powerful operations such as array based operations, linear algebra etc. for python programming as well. Though, in this approach, the Numpy library has used in each step of implementation to manipulate mathematical details easily.

4.1.4 Python SciPy

Python SciPy is an open-source python library introduced for scientific computing which has a collection of modules for useful aspects in image processing, signal processing and other related areas. The SciPy library depends on NumPy and provides user-friendly and efficient numerical practices.

4.1.5 Python Scikit-Learn

Python Scikit-Learn library provides many tools which facilitate classification, regression, clustering, model selection and pre-processing for different aspects of machine learning. In this proposed approach, Scikit-Learn is used for the implementation of YIN pitch detection for monophonic melodies.

4.2 Implementation of Melody Processing Phase

Before moving into the pitch detection of a melody, melodies were processed through some preprocessing steps such as silence removing, lowpass filtering and downsampling to improve the process of feature extraction. Following two code segments (4.1 and 4.2) are used as the silence removing and filtering methods respectively.

```
def silence_remover(signal):
    started = False
    sig_arry = array('h')
    for i in signal:
        if not started and abs(i) > threshold:
            started = True
            sig_arry.append(i)
        elif started:
            sig_arry.append(i)
    return sig_arry
```

Python Code 4.1: Silence removing of a signal

```
def lowpass_filter(signal, cutoff, Fs, order=6):
    yq = 0.5 * Fs
    new_cutoff = cutoff / yq
    m, n = scipy.signal.butter(order,
                               new_cutoff,
                               btype='low',
                               analog=False)
    signal = scipy.signal.lfilter(m, n, signal)
    return signal.astype(np.int16)
```

Python Code 4.2: Low pass filtering

These preprocessing steps have implemented based on the signal class implementation in Scipy library. Output signals from this phase are used for the pitch detection phase.

YIN algorithm has used as its original version with some minor changes to have better performance for monophonic melodies. Each function and step in the YIN algorithm has implemented as separate functions in python. The main function is defined to call those sectional methods. The final output has obtained as a feature vector. This output vector consists of the frequencies obtained for the pitch class estimations during the time frames. Following code segment has used as the main function for YIN pitch estimation. In here, the preprocessing function is also called by the main function. Preprocessing function is calling the above mentioned two functions for silence removing and low pass filtering through a down-sampling method. (The complete code for each function is attached in Appendix B).

```
def yin_algorithm(file_name, threshold, t_fs, freq_range, timestep):

    #extract signal from audio file
    signal, fs = getdata(file_name)
    window = int(np.ceil(Fs/freq_range[0]))

    #pre-processing
    signal, fs = preprocess(signal, fs, window, t_fs, timestep)

    #apply the steps in the algorithm
    auto_cor_mat = auto_correlation(signal,window)
    diff_eq = diff_equation(auto_cor_mat)
    c_diff_eq = c_mean_diff_equation(diff_eq_mat)
    taus = absolute_threshold(c_diff_eq, freq_range, threshold, fs)[0]
    periods = parabolic_interpolation(diff_eq, taus, freq_range, fs)
    estimates = get_estimates(signal, fs, periods, window)

    return estimates
```

Python Code 4.3: YIN implementation

4.3 Implementation of Note Event Model

The first part of the note event model is implemented to detect note events within a sequence of pitch value estimations. This uses a mapping algorithm to have a score for note events. Within this mapping, all the notes which have the same key in different octaves are taken as a single score. The note event model also uses a simple algorithm to ignore repeating pitch values within consecutive estimations. Resulting note event sequences are then used for the LSTM model as the most important part of the note event model.

The required LSTM model is implemented based on the sequential way of building deep learning networks in Keras. Keras neural network models mainly involve in two forms as sequential and functional API. Sequential model allows to easily adapt sequential layers where functional API allows to build more complicated network structures. In this implementation, sequential form has used to have the recurrent layers in the network.

Three LSTM layers are added to the sequential model of the network and three different values are used as the number of nodes in the hidden layers. LSTM layer is a Recurrent Neural Network layer that takes a sequence as an input. The first two LSTM layers are specified with the parameter *return_sequences=True* which ensures that the LSTM cell returns all of the outputs from the unrolled LSTM cell through time. Final LSTM layer is not specified with the above parameter and it simply provides the output of the LSTM cell from the last time step.

A dropout layer has designed to follow each LSTM layer in order to prevent overfitting. It has a special technique that consists of setting a fraction of input units to 0 at each update during the training. Dense layer is a fully connected neural network layer where each input node is connected to each output node. The Activation layer determines what activation function will use to calculate the output of a node in the model. *Softmax* activation was chosen as the activation function for the model.

```
1    model = Sequential()
```

```
    model.add(LSTM(
            256,
            input_shape=(network_input.shape[1],
                         network_input.shape[2]),
            return_sequences=True
             ))
    model.add(Dropout(0.3))
    model.add(LSTM(512, return_sequences=True))
    model.add(Dropout(0.3))
    model.add(LSTM(256))
    model.add(Dense(256))
    model.add(Dropout(0.3))
    model.add(Dense(n_vocab))
    model.add(Activation('softmax'))
```

Python Code 4.4: Keras implementation of LSTM model

Each parameter value for the Keras LSTM implementation was chosen after an initial evaluation, which performed to achieve the best suited model for the problem.

However, before train the model, encoded data should be converted in to a normalized standard. This model input conversion has done by reshaping the input into a format compatible with LSTM layers. That converted input was then normalized according to the vocabulary size of the entire dataset.

```
def data_conversion(data,vocab_size):
    n_patterns = len(data)
    network_input = numpy.reshape(data,
                                  (n_patterns,sequence_length,1))
    network_input = network_input / float(vocab_size)
    network_output = np_utils.to_categorical(data)
```

Python Code 4.5: Function to convert input data

After the training with accurate weights, model has designed to test with data to have the predictions from unknown patterns. Final predictions were then extracted using following function.

```
def prediction(data,vocab_size):
    prediction_output = []
    for i in range (len(data)):
        pattern = data[i]
        prediction_input = numpy.reshape(pattern,(1,len(pattern),1))
        prediction_input = prediction_input / float(vocab_size)
        prediction = model.predict(prediction_input, verbose=0)
        index = numpy.argmax(prediction)
        prediction_output.append(index)
    return prediction_output
```

Python Code 4.6: Function to get prediction outputs

4.4 Chapter Summary

This chapter discussed the implementation of the proposed design of the approach. Python was used to this implementation as the supporting language. Also, several other libraries have used for signal processing and machine learning tasks. The implementation of the pitch detection phase was entirely designed with the YIN algorithm. LSTM model was implemented as the main part of the note event model. This used the Keras neural network implementation which works smart for deep learning. The implementation of the testing phase of the model is described in Chapter 5 with the evaluation phase.

Chapter 5

Results and Evaluation

As mentioned before, this approach defines a combined process of several computational music content analysis techniques. The entire design has two phases where the second phase used the results obtained from the first phase. Thus, this chapter has structured to analyze the results of the pitch detection phase as well as the overall evaluation of the anomaly detection phase.

In general approaches, melody transcription techniques can be qualitatively or quantitatively evaluated. But in this research design, melody transcription is not involved as the whole structure, and it was designed to follow only some critical steps in melody transcription. Therefore a quantitative evaluation methodology is employed for the final note detection, by measuring the difference between the reference note sequences and the transcribed detected note sequences.

5.1 Analysis of Pitch Detection

Under the melody processing phase, the YIN algorithm has used to obtain the pitch values within the melody samples. These pitch values are the inputs to the note event model and its final anomaly detection phase. The LSTM model was designed to use these obtained pitch values after the detection of reference note events. Therefore, the pitch detection phase has performed a valuable step in this proposed approach. In here, the YIN results are analyzed here as the initial step.

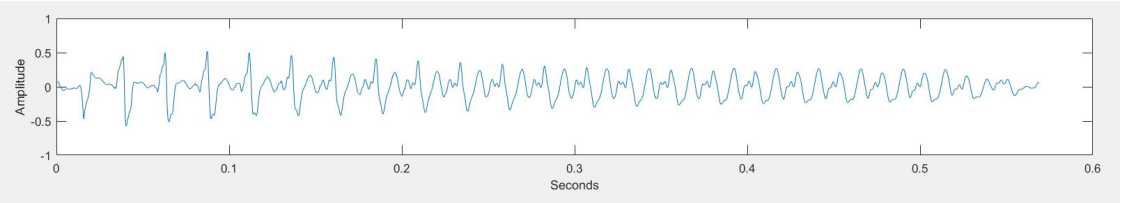


Figure 5.1: Sample input signal for YIN Algorithm

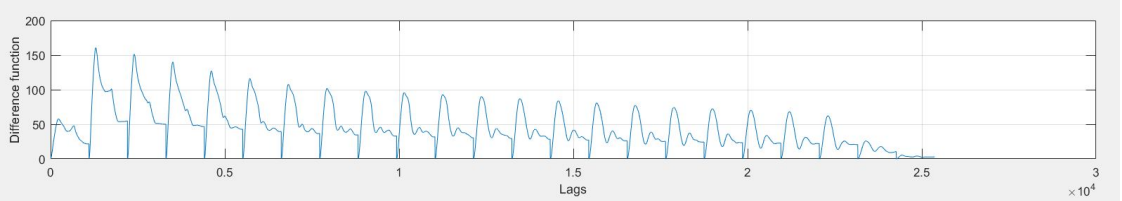


Figure 5.2: Output for difference function in YIN Algorithm

Figure 5.3: Output for cumulative mean difference function in YIN Algorithm

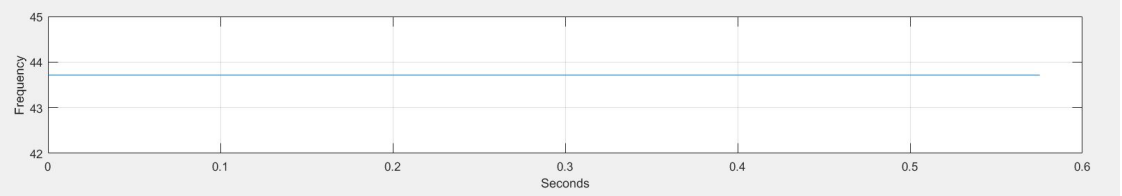


Figure 5.4: Output Pitch for YIN Algorithm

The YIN algorithm has been used with several steps, as it is by the original authors. Each step has analyzed through the implementation, and additional points are added to overcome some issues with the input formats regarding the monophonic melodies. Within the YIN algorithm, the average magnitude difference function has used, and several modifications have extended to improve the pitch estimation accuracy and robustness.

In here, a sample single note melody has been considered for the YIN algorithm and the input signal is shown in Figure 5.1. That signal represents the melody of 'E' note in the first octave. The obtained results for difference function and

the cumulative mean normalized difference function of YIN algorithm are shown in Figure 5.2 and Figure 5.3 respectively. Figure 5.4 shows the final detected pitch estimations for the above single note melody. Since the melody consists of a single note, the output provides a single pitch value as 43.715 Hz.

The obtained value is then tuned with the proposed tuning algorithm and the tuned pitch value for the above 'E' note has obtained as 41.85 Hz. However, the actual pitch value for the note 'E' in 1st octave is 41.20 Hz. This correct pitch value has achieved as the reference pitch value through the difference function.

Figure 5.5 shows a long period sample melody input and its outputs for intermediate steps in YIN. Figure 5.6 shows the output pitch estimations for that particular melody sample without applying the tuning process.

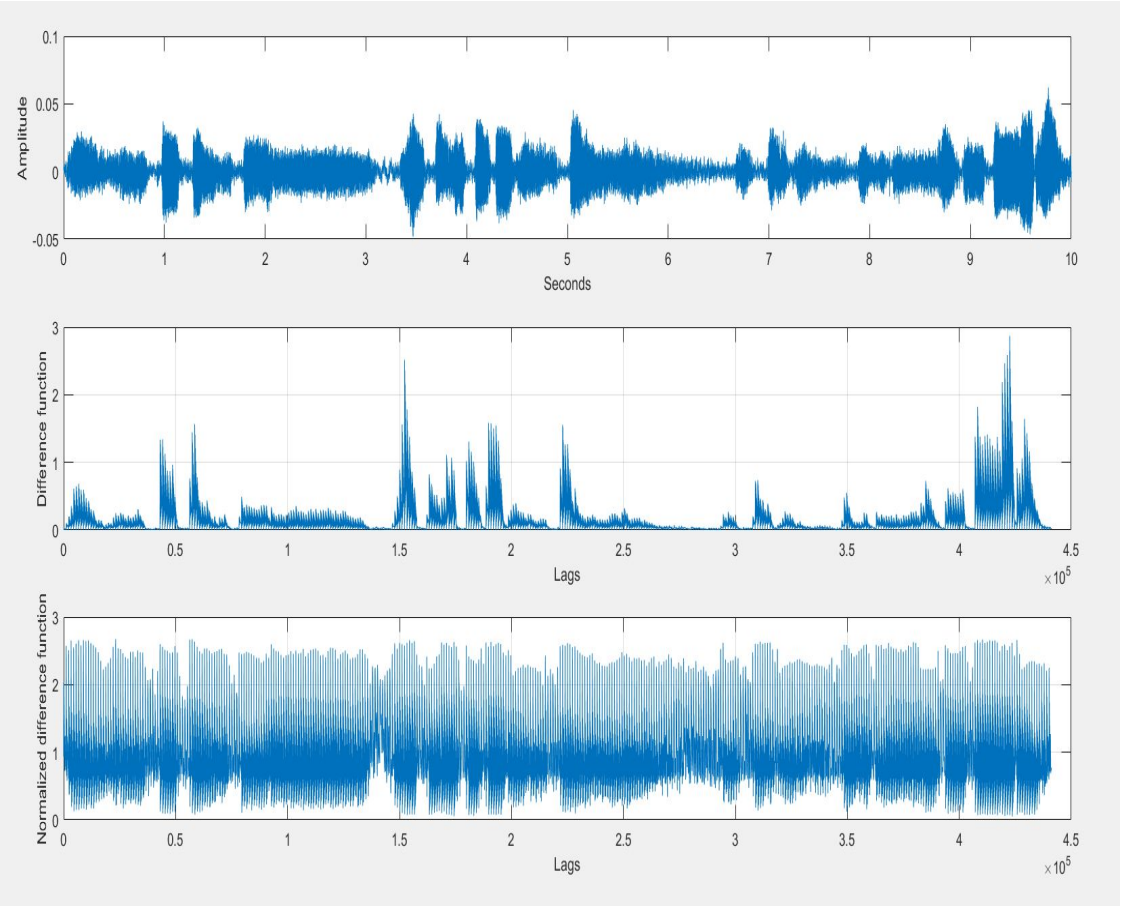


Figure 5.5: YIN process for a long-period melody sample

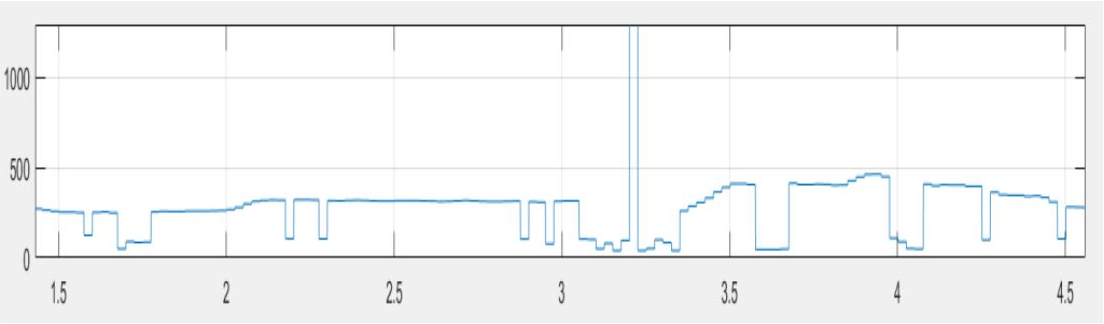


Figure 5.6: YIN output for a long-period melody sample

Though, as explained in the above sample melody, the YIN was succeeded to achieve more accurate estimations for the note pitch values.

5.2 Overall Evaluation

As the required monophonic melody dataset, a set of recorded audio melodies are considered as the unknown monophonic vocal and instrumental melodies. Mainly there are two different types of audio formats as lossless audio formats and lossy audio formats. Lossless audio formats such as WAV, FLAC, and AIFF etc. preserve the original data as it is, whereas lossy audio formats such as MP3, OGG etc. omit some data from the original data. However, the file size of lossless audio files is larger than the lossy audio file size. Despite the large file size, WAV audio format at a sampling rate of 44 100 Hz is used in these recorded vocal melodies. Each sample audio contains 12 to 16 seconds long melodies. The sample audio clips are used to analyze the feature extraction phase of the design and all the behaviours are encountered through the theoretical concepts.

However, the proposed design is analyzing a set of notes as a sequence one at a time. Therefore, it is necessary to have a set of melodies which includes a large set of patterns in all the scales. Since this research is assumed to use only major scale patterns, the dataset should include melodies with different types of patterns of all major scale as much as possible. This is a very important fact to train the LSTM model with more possible patterns. But in the real world, some music scales occur in very few songs or melodies. As an example, there are only a few sinhala song melodies in B major scale. Therefore it is hard to have an equal no of melodies

for each scale type. The dataset has performed with melodies in all seven major scales as much as possible.

The overall evaluation of the approach is based on the note event model which was designed with a LSTM neural network. The final evaluation was done using the monophonic melody dataset which has collected under the mentioned conditions. 95% of melodies in this dataset are vocal melodies and the rest of the melodies are instrumental melodies. However, all the melodies consist of single vocal melody or a single instrumental melody. Therefore, the dataset is fully concerned as a set of unknown monophonic melodies.

5.2.1 Training

The model has trained with a set of melody note patterns in all seven major scales which includes more precise scale related notes. The model was designed to train on note event sequences of predefined length and to predict the next note event of that sequence. The note sequences length was chosen as seven for the LSTM model. In order to have an unbiased training process, the training data has collected across all major scales as presented in the Table 5.1. Implemented model in Keras was trained with the *model.fit()* function for 150 epochs with the batch size of 64 samples.

Table 5.1: Training Dataset

Training Data		
Scale	**Samples**	**Total Note Event Patterns**
C Major	29	1698
D Major	24	1260
E Major	23	991
F Major	39	1822
G Major	25	1263
A Major	27	1390
B Major	11	673
Total	**178**	**9097**

5.2.2 Testing

The trained model was tested with a set of unknown monophonic melodies, which already contains melody samples of all major scales. Table 5.2 shows the testing dataset which was also collected across all major scales. This test data set was collected from a melody set which has more less pleasant melodies. The reason for taking that kind of melodies for the testing process is to have much more scale unrelated notes. Note events are detected through the estimated pitch values of the melodies. As in the training process, note event sequences of the same predefined length are tested to predict the next note event of the pattern. The predicted note event is then tested with the actual note event of the sequence. Final prediction decision has taken upon these two note events based on the distance measure.

Table 5.2: Testing Dataset

Testing Data		
Scale	**Samples**	**Total Note Event Patterns**
C Major	19	1423
D Major	14	980
E Major	12	891
F Major	19	1445
G Major	18	1087
A Major	13	995
B Major	4	203
Total	**99**	**7024**

5.2.3 Test Results

The LSTM model has designed to predict the next upcoming note event of each note pattern. This predicted note event is then compared with actual note event to have the final prediction. Though, the final prediction will provide the result as whether the observed note is an anomaly or a not. Table 5.3 presents the results obtained from the model for each pattern in each group of scale. The performance of

the anomaly detection was measured with error rate for each major scale category. As mentioned there, the model was able to achieve **68.2% overall accuracy**.

Table 5.3: Results for test data

Scale	True Predictions	False Predictions	Error Rate
C Major	1002	421	0.295
D Major	675	305	0.311
E Major	620	271	0.304
F Major	1090	355	0.246
G Major	659	428	0.393
A Major	642	353	0.355
B Major	102	101	0.497
			0.318

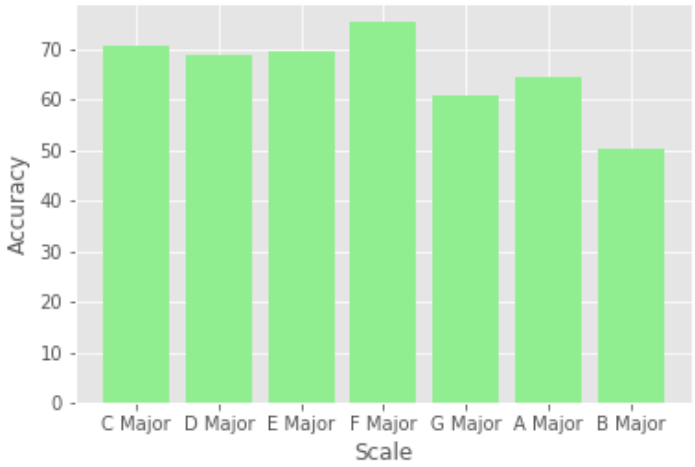


Figure 5.7: Accuracy by Scale Groups

5.2.4 Comparison with known piano melodies

In here, an additional evaluation was done for the note event model with a monophonic piano melody data set. The main aim of this additional evaluation is to compare the performance of the model between two different data representations. The piano melodies which are used here are collected with their MIDI pitch value representations. Therefore, those melodies do not need the initial pitch detection as their pitch values are already known.

Table 5.4: Performance of the model for piano melody dataset

Scale	True Predictions	False Predictions	Error Rate
C Major	695	45	0.061
D Major	673	92	0.120
E Major	345	89	0.205
F Major	541	93	0.147
G Major	269	86	0.242
A Major	433	145	0.251
B Major	108	42	0.280
			0.161

However, the overall accuracy for this piano melody data set has obtained as 84%, which is better than the previously used unknown monophonic dataset. The Table 5.4 shows the performance of the model for this monophonic piano melody set. The reason for this better performance is that the model did not apply the pitch estimation process for the melodies. Instead, it uses MIDI references pitch values which are already defined without any estimation.

5.3 Discussion

The results and the analysis of the pitch detection phase indicate that the pitch estimations were obtained more accurately with the use of a tuning algorithm. If this pitch detection was not capable of providing more reliable estimations, then there is a chance to have falsifiable predictions within the note event model.

The note event model was designed with a Long Short Term Memory (LSTM) Neural Network as a sequence prediction model. The evaluation results for the unknown monophonic data indicates that the design of this prediction model is applicable to the detection of anomalous note changes.

With the comparison with piano melodies of MIDI note references, it is clear that the model can perform better when the pitch estimations and note event estima-

tions have their best values. Though, a better pitch detection methodology can improve the performance of the entire model.

5.4 Chapter Summary

This chapter discussed the results obtained for the proposed model with the details of the evaluation plan, used dataset and limitations. Results of the note detection phase have analyzed with selected examples. The overall performance of the model has obtained nearly 69% accuracy for the used dataset. Also, an additional evaluation was done for a monophonic piano melody dataset, in order to have a comparison of results. The obtained results for the first unknown monophonic melodies are then compared with the results of this piano melody evaluation. As stated in the discussion Section, this comparison can conclude that a better pitch detection methodology can improve the performance of the overall model.

Chapter 6

Conclusion

6.1 Introduction

This chapter focuses on the conclusions based upon the proposed research approach. The aim, research questions and the objectives of the research as stated in Section 1.2 has been analyzed with the obtained results from the evaluation. Finally the conclusions are given for the overall research design and the further improvements are suggested as future works.

6.2 Conclusions about research questions

The main aim of this research was to conduct an anomaly detection approach for unknown monophonic melodies, by identifying the effects of uncommon note changes in melody performances evaluation.

This approach was designed to have a more generalize way of anomalous note detection in the context of computational music content analysis. Thus, following research question and its sub questions were stated for the approach.

Research Question:

- ***How to detect anomalous note changes that affect the overall performance evaluation of melodies?***

- *How do anomalous note changes affect to the overall performance of a melody?*
- *How can one define the anomalous note changes in the sense of uncommon note occurrences in a melody?*
- *What are the approaches that can be used to detect anomalous note changes?*

As the first step of this research approach, melody evaluation approaches were analyzed to identify its limitations and barriers which lead to having weak performances. The effect of anomalous note changes was then highlighted among them and had an individual analysis based upon that effect. Since there are several melody evaluation approaches in literature, it was important to figure out the limitations of those methods regarding the context of note change events in the melody sequences.

In this study, anomalous note changes were defined as the scale unrelated note changes of a melody. Effects of harmonies and other variations of melodies are ignored and all the melodies were considered as just simple monophonic melodies. Among the identified uncommon note events, a rule set for its scale was defined to remark the anomalies in the note sequence in a melody.

In literature, there are different kinds of machine learning based methods have used as the anomaly detection approaches for sequential data. Long Short Term Memory (LSTM) was chosen among them, which was already used in computer based music content analysis approaches such as melody transcription, melody creation etc. Also LSTM was able to perform better in sequential data analyzing approaches as well.

Though, as the required technical perspectives, a set of signal processing and machine learning approaches have been applied in this approach, to identify the above-defined anomalous note changes in unknown monophonic melodies.

6.3 Conclusions about research problem

A novel approach was proposed to detect anomalous notes changes of musical melodies. The proposed model is composed with two main phases. Within the first phase, melodies are processed to have their pitch estimations through steps of feature extraction and fundamental frequency estimation. Then a Long Short Term Memory (LSTM) neural network is employed as a note event model for the detection of 'anomalous notes' regarding the estimated pitch of sampled melody signals.Though by introducing a LSTM based neural network method as a note event model, this research encountered the anomalous note change detection of unknown monophonic melodies. Note event model uses the melodies which are already processed through several stages of melody processing steps.

This study contributed to the domain of computational music content analysis by conducting an approach for anomalous note change detection. This approach is capable of detecting anomalies of note events of unknown monophonic melodies as the simplest type of musical melodies. This study provides a methodology, which can be applied as an application for melody performance evaluation to improve their results. Also, comparison with the existing melody transcription approaches, this study can be introduced as an improvement for the note event estimation. By identifying and eliminating anomalous note changes, melody transcription can provide better score information with more accurate note sequences as the output. This can be also combined with the melody performance evaluations as well. The combined process will provide a better melody performance evaluation results by minimizing the effect of anomalous note changes.

6.4 Limitations

This study was mainly designed to focus on unknown monophonic melodies. Monophonic melody is the simplest texture of music. It is the biggest limitation as a novel approach for anomalous note change detection. The monophonic melodies are considered to have no harmonic or any other variations within the melodies in order to detect the scale unrelated note changes as anomalous note changes.

Therefore, this study of anomalous note change detection was designed to follow an accurate way of anomaly detection in a selected category of music melodies under certain conditions.

6.5 Implications for further research

Identification of different types of variations of the pitch can provide more reliable results for anomalous note change detection regarding the scale unrelated note events. Thus, this study can be further implicated to have such a way to distinguish the variations of the pitch from anomalous note changes. However, for polyphonic melodies, it is difficult to identify anomalous note events among two or more simultaneous pitch sounds. But that is also possible to conduct a more complicated approach based on this study, to perform under polyphonic melodies as well.

References

[1] J. Downie, "Music information retrieval," *Annual Review of Information Science and Technology*, vol. 37, no. 1, pp. 295–340, 2005.

[2] "The science of anomaly detection," *Numenta, Redwood City, CA*, 2015.

[3] M. Ryynänen and A. P. Klapuri, "Modeling of note events for singing transcription," *Proc. ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio*, 2004.

[4] "Physics of Music Notes." `http://pages.mtu.edu/~suits/notefreqs.html`, accessed 2018-12-01.

[5] "Understanding Music." `https://www.musictheoryacademy.com/`, accessed 2018-10-12.

[6] M. Mauch, K. Frieler, and S. Dixon, "Intonation in unaccompanied singing: Accuracy, drift, and a model of reference pitch memory," *The Journal of the Acoustical Society of America*, vol. 136, no. 1, pp. 401–411, 2014.

[7] P. Żwan, "Automatic singing quality recognition employing artificial neural networks," *Archives of Acoustics*, vol. 33, no. 1, pp. 65–71, 2008.

[8] T. Nakano, M. Goto, and Y. Hiraga, "An automatic singing skill evaluation method for unknown melodies using pitch interval accuracy and vibrato features," pp. 1706–1709, 2006.

[9] C. Toh, B. Zhang, and Y. Wang, "Multiple-feature fusion based onset detection for solo singing voice," *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pp. 515–520, 2008.

[10] M. Alonso, B. David, and G. Richard, "Tempo and beat estimation of musical signals," *Proc International Conference on Music Information Retrieval*, vol. 4, pp. 158–163, 2004.

[11] A. Mesaros, "Singing voice identification and lyrics transcription for music information retrieval," *Proceedings of International Conference on Speech Technology and Human-Computer Dialogue, Cluj-Napoca, Romania*, pp. 1–10, 2013.

[12] S. Bock and M. Schedl, "Polyphonic piano note transcription with recurrent neural networks," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 121–124, 2012.

[13] M. Mauch, C. Cannam, R. Bittner, G. Fazekas, J. Salamon, J. Dai, J. Bello, and S. Dixon, "Computer-aided melody note transcription using the tony software: Accuracy and efficiency," *Proceedings of the First International Conference on Technologies for Music Notation and Representation, Research Institute in Musicology*, pp. 23–30, 2015.

[14] A. de Cheveigne and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[15] M. Mauch and S. Dixon, "pyin: A fundamental frequency estimator using probabilistic threshold distributions," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 659–663, 2014.

[16] A. Laaksonen, "Automatic melody transcription based on chord transcription.," *International Society for Music Information Retrieval (ISMIR)*, pp. 119–124, 2014.

[17] E. Crawley, B. Acker-Mills, R. Pastore, and S. Weil, "Change detection in multi-voice music: The role of musical structure, musical training, and task demands," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 28, no. 2, pp. 367–378, 2002.

[18] A. Bishara and G. Radvansky, "The detection and tracing of melodic key changes," *Perception Psychophysics*, vol. 67, no. 1, pp. 36–47, 2005.

[19] D. Shipmon, J. Gurevitch, P. Pisellis, and S. Edward, "Time series anomaly detection; detection of anomalous drops with limited features and sparse examples in noisy highly periodic data," 2017.

[20] L. Bontemps, J. McDermott, N. Le-Khac, and V. Loi-Cao, "Collective anomaly detection based on long short-term memory recurrent neural networks," *International Conference on Future Data and Security Engineering, Springer*, pp. 141–152, 2016.

[21] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," *European Symposium on Artificial Neural Networks*, vol. 23, 2015.

[22] S. Mahendra, H. Patil, and N. Shukla, "Pitch estimation of notes in indian classical music," 2010.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] "Understanding LSTM Networks." `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`, accessed 2018-11-01.

Appendices

Appendix A

Frequencies for Music Notes

Table A.1: Frequency values for music notes

Note	**Frequency (Hz)**
C0	16.35
C♯0/D♭0	17.32
D0	18.35
D♯0/E♭0	19.45
E0	20.60
F0	21.83
F♯0/G♭0	23.12
G0	24.50
G♯0/A♭0	25.96
A0	27.50
A♯0/B♭0	29.14
B0	30.87
C1	32.70
C♯1/D♭1	34.65
D1	36.71
D♯1/E♭1	38.89
E1	41.20
F1	43.65

Note	**Frequency (Hz)**
F♯1/G♭1	46.25
G1	49.00
G♯1/A♭1	51.91
A1	55.00
A♯1/B♭1	58.27
B1	61.74
C2	65.41
C♯2/D♭2	69.30
D2	73.42
D♯2/E♭2	77.78
E2	82.41
F2	87.31
F♯2/G♭2	92.50
G2	98.00
G♯2/A♭2	103.83
A2	110.00
A♯2/B♭2	116.54
B2	123.47

Note	Frequency (Hz)
C3	130.81
C♯3/D♭3	138.59
D3	146.83
D♯3/E♭3	155.56
E3	164.81
F3	174.61
F♯3/G♭3	185.00
G3	196.00
G♯3/A♭3	207.65
A3	220.00
A♯3/B♭3	233.08
B3	246.94
C4	261.63
C♯4/D♭4	277.18
D4	293.66
D♯4/E♭4	311.13
E4	329.63
F4	349.23
F♯4/G♭4	369.99
G3	392.00
G♯4/A♭4	415.30
A4	440.00
A♯4/B♭4	466.16
B4	493.88

Note	Frequency (Hz)
C5	523.25
C♯5/D♭5	554.37
D5	587.33
D♯5/E♭5	622.25
E5	659.25
F5	698.46
F♯5/G♭5	739.99
G5	783.99
G♯5/A♭5	830.61
A5	880.00
A♯5/B♭5	932.33
B5	987.77
C6	1046.50
C♯6/D♭6	1108.73
D6	1174.66
D♯6/E♭6	1244.51
E6	1318.51
F6	1396.91
F♯6/G♭6	1479.98
G6	1567.98
G♯6/A♭6	1661.22
A6	1760.00
A♯6/B♭6	1864.66
B6	1975.53

Note	Frequency (Hz)
C7	2093.00
C♯7/D♭7	2217.46
D7	2349.32
D♯7/E♭7	2489.02
E7	2637.02
F7	2793.83
F♯7/G♭7	2959.96
G7	3135.96
G♯7/A♭7	3322.44
A7	3520.00
A♯7/B♭7	3729.31
B7	3951.07

Note	Frequency (Hz)
C8	4186.01
C♯8/D♭8	4434.92
D8	4698.63
D♯8/E♭8	4978.03
E8	5274.04
F8	5587.65
F♯8/G♭8	5919.91
G8	6271.93
G♯8/A♭8	6644.88
A5	7040.00
A♯8/B♭8	7458.62
B8	7902.13

Reference : http://pages.mtu.edu/suits/notefreqs.html

Appendix B

Code Listings

B.1 YIN Pitch Detection

B.1.1 Autocorrelation Method

```
    def auto_correlation(signal, tau, w):
        autocorr_mat = np.zeros((signal.size//(2*w), w), np.float32)
        sig_orig = list(signal)
        for i in range(autocorr_mat.shape[0]):
            t = 2*i*w
            signal = signal_orig[t:w+t] - np.mean(sig_orig[t:w+t])
            x_tau = x_orig[t+tau:t+tau+w]
                    - np.mean(x_orig[t+tau:t+tau+w])
            autocorr = np.correlate(sig_tau, sig_tau, 'full')
            autocorr_mat[i, :] = autocorr[autocorr.shape[0]//2:]
        return autocorr_mat
```

Python Code B.1: Autocorrelation method in YIN

B.1.2 Difference Function

```
    def difference_function(auto_corr_mat, signal, w):
        diff_fn_mat = np.zeros(auto_corr_mat.shape, np.float32)
        for i in range(0, diff_fn_mat.shape[0]):
            diff_fn_mat[i,:] = (auto_corr_mat[i, 0]
                                + auto_correlate(signal, i, w)[i, 0]
                                - 2 * auto_corr_mat[i,:])
```

```
        return diff_fn_mat
```

Python Code B.2: Difference function in YIN

B.1.3 Cumulative Normalized Mean Difference Function

```
    def cumulative_difference_function(diff_fn_mat):
        cum_diff_mat = np.zeros(diff_fn_mat.shape, np.float32)
        cum_diff_mat[:, 0] = 1
        for t in range(diff_fn_mat.shape[0]):
            for j in range(1, diff_fn_mat.shape[1]):
                cum_diff_mat[t, j] = diff_fn_mat[t, j] /
                                     ((1/j)*sum(diff_fn_mat[t,1:j+1]))
        return cum_diff_mat
```

Python Code B.3: Cumulative Mean Normalized Difference function in YIN

B.1.4 Absolute Threshold Method

```
    def bsolute_thresold(cum_diff_mat, freq_range, threshold, fs):
        tau_min = int(fs)//freq_range[1]
        tau_max = int(fs)//freq_range[0] - 1
        tau_star = 0
        minimum = 1e9
        taus = np.zeros(cum_diff_mat.shape[0], np.int16)
        for i in range(taus.size):
            cum_diff_eq = cum_diff_mat[i,:]
            for tau in range(tau_min, tau_max):
                if cum_diff_eq[tau] < threshold:
                    taus[i] = tau
                    break
                elif cum_diff_eq[tau] < minimum:
                    tau_star = tau
                    minimum = cum_diff_eq[tau]
            if taus[i] == 0:
                taus[i] = tau_star
        return taus, cum_diff_mat
```

Python Code B.4: Absolute Threshold Method in YIN