SYSTEMATIC APPROACHES TO MUSIC REHEARSAL SEGMENTATION

Taken From https://medium.com/@simmab0000/piano-practice-can-be-fun-too-6d9a3e147d1e

TU Delft
Delft University of Technology

YIZI CHEN

Systematic Approaches To Music Rehearsal Segmentation

by

Yizi Chen

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Wednesday October 9, 2019 at 13:00 PM.

Student number:	4621174	
Project duration:	September 1, 2018 – October 9, 2019	
Thesis committee:	Prof. Dr. A. Hanjalic,	TU Delft, Chair
	Dr. C.C.S. Liem,	TU Delft, Supervisor
	Dr. A. Panichella ,	TU Delft, Committee

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Contents

Preface

I started to play piano at the age of four. Until now, music is always an essential part of my life. I ever dreamed and wondered if I could use my academic knowledge to make some contribution to music society. Last year, I was super lucky to meet Dr. Cynthia Liem, who is professional in both music and computer science. I have an opportunity to become one of her master students and to combine computer science and music knowledge into a master thesis topic.

Firstly, I would like to express my great appreciation to my supervisor, Cynthia Liem for her innovative and constructive suggestions during this period of research work. Although she had a very tight schedule, she still made time to have meetings with me every week and provided many useful advises supporting my work. Her works of building a bridge between computer science and music society would benefit many musicians in the future and that impresses me a lot. It became a power in my heart to tackle any difficulties in this thesis work.

Secondly, I would like to thank my grandparents, my parents and the rest of the family members. Their selfless love always makes me feel warm and safe, and they provided shelter to help me go through the darkest moments.

Thirdly, I would express my appreciation to Michael The, the previous master student who worked on this subject and who guided me a lot at the beginning of this work.

In the end, I would express my special thanks to Yi, who gave me many useful advises on how to write a good thesis and gave me countless support on writing.

Yizi Chen
Delft, October 2019

Abstract

Practice is always the secret to the success of musicians. Many musicians record their rehearsal sessions and listen back to reflect on their practice. However, the rehearsal sessions are unstructured, messy and rather long compared to commercial recordings. As a result, musicians may not have the capacity to listen back to all of their practice recordings comprehensively. Nowadays, Music Information Retrieval (MIR) techniques have been developed to better manage and filter informative representations from commercial recordings. However, not much research has been performed focusing on rehearsal sessions. What differentiates rehearsal session recordings from 'regular' recordings, is both their length and the unpredictability of the content within them. Besides, rehearsal recordings will not have labeled ground truth on their content, and obtaining this would require a massive amount of manual labeling work; thus, it is unrealistic to achieve. In this thesis, we therefore propose a systematic development and evaluation framework to deal with these challenges. In detail, we will focus on the segmentation of full rehearsal session recordings into meaningful repeated fragments that could be used by musicians. To this end, we propose a framework which adopts an unsupervised segmentation strategy that can be robust to expected variability in 'meaningful' repeats. While well-defined ground truth is absent, by employing an evaluation strategy that synthesizes the ground truth based on real recordings, we still get insight into the performance of our methods.

1 Introduction

Practice, the act of repeatedly performing an activity in order to acquire or polish a skill, is an inevitable step towards mastering a musical instrument. Becoming a master musician requires at least a decade's worth of contribution to practice [17]. While there are a massive amount of musicians around the world spending almost the same amount of time practicing, only a few of them have become masters. The reason is that only investing many hours is not sufficient; the practice should also be beneficial and smart. In other words, the quality of the practice is the key to becoming a successful musician.

What is the way to assess the quality of practice? In the conservatory, the performance of the students can be assessed through a weekly class with music educators. However, the time of this class is limited. In such a short amount of time, it is not easy to find and correct all the potential problems of a student. As a potential solution, while the students are practicing, they can record their rehearsals. Listening back to these recordings can give further insight into points of improvement. However, the rehearsal sessions can take many hours, and it is, therefore, unrealistic to assume a human can listen back to all of them.

Recently, many kinds of research in the field of Music Information Retrieval (MIR) have been investigated, aiming for extracting useful music content. For example, research has been conducted into automatically extracting music structural information by using machine learning and signal processing techniques. Briefly speaking, music structure analysis, "refers to the process of recovering a description of the sectional form" [18] and "the structure of a musical piece can be described with segments having a specific time range and a label" [17]. As a consequence, research in music structure analysis has largely focused on extracting musical building blocks such as the intro, verse, pre-chorus and chorus (or refrain).

Work in MIR, including music structure analysis, has so far focused on commercial recordings. However, with musical practice being the act of repeatedly performing an activity in order to refine a musical piece, rehearsal sessions also have internal structure, although the structure is less defined than in traditional structure analysis.

More specifically, two types of repetitions exist in the rehearsal recordings. One type of repetitions is due to the material being repeated in a composition. The other type of repetitions are created due to musicians revisiting the material more often in a rehearsal. We ultimately are mostly interested in the latter types of repeats. However, to distinguish between the two, we would need to know upfront what the musicians play. However, as we would like to contribute a framework that is as generalizable as possible, we want to avoid that the availability of this knowledge is required before the analysis can be done. Furthermore, repetitions in the rehearsal recordings are expected to have degrees of variation. They will not be exactly replicated in time, and may contain errors or experimentations. However, they are expected to still maintain the ordering of harmonic content.

Contrasting commercial recordings with rehearsal recordings, several further main differences can be found between the two. Firstly, commercial recordings have well-defined music structures and fewer errors, but real rehearsal recordings are less structured, fragmented, and include much experimentation that is new to existing MIR literature. Secondly, while music structure analysis is focusing on longer structural music patterns, meaningful repeats in rehearsal recordings are expected to be much shorter (usually less than ten seconds) while they are part of a longer recording than a commercial song would be.

In this thesis, we therefore propose a rehearsal analysis framework that can handle the typical characteristics of rehearsal data and that allows for systematic evaluation, even when objective ground truth is absent.

Considering that different types of repeats exist in the rehearsal recordings, we design a synthetic data generator that can yield fully or partially controllable ground truth to evaluate the performance of rehearsal analysis framework. The following research questions will be addressed:

1. How can we design a rehearsal analysis framework which can automatically extract informative representations from the rehearsal recordings?
2. How can we prepare data to evaluate the performance of the rehearsal analysis system when the ground truth is missing?
3. How can we evaluate the output of information representations extracted from rehearsal analysis system?

Our ultimate goal is to realize the analysis of rehearsal recordings through MIR techniques to assist more musicians and to evaluate their progress of rehearsal in a comfortable, simple, and fast way. Little work has been done so far in this area, and the existing work requires more solid evaluation procedures. This work, and other related themes from MIR, will be discussed in Chapter 2. In Chapter 3, we illustrate a big picture of the rehearsal analysis approach. After that, we elaborate each part in the following chapters. Chapter 4 gives more details about the music pre-processing step. Chapter 5 describes the rehearsal analysis framework, which transforms rehearsal recordings into meaningful and listenable repetitions. Due to the lacking ground truth in the real rehearsal recordings, we present a synthetic data generator in Chapter 6, which can be used to automatically synthesize ground truth. The performance of the rehearsal analysis framework can be evaluated using several segmentation evaluation methods, which are discussed in Chapter 7. Once we have introduced all the methodologies we need, the experimental results on our synthetic data are reported in Chapter 8. Furthermore, we project how our framework can be used on real rehearsal data in Chapter 9. In the end, we present a conclusion and discuss future work in Chapter 10.

2

Related Work

This chapter introduces the background knowledge in the literature that is related to our research. Section 2.1 in this related work expands our vision on the topic of **Music Structure Analysis**, which is a common subject in MIR to extract informative representations in the music. This chapter is organized in the following. Existing methods that find similar music segments in the recordings are described in Section 2.2. Feature extraction is the essential step to describe the content of the audio which is illustrated in Section 2.3. Rehearsal analysis systems use to monitor the progress of the musicians in their rehearsal sessions which is presented in Section 2.4. In the end, several evaluation methods for evaluating informative musical content are described in Section 2.5.

2.1. Music Structure Analysis

The music is built out of notes, which together form patterns and melodies, and the way these are repeated, contrasted, and varied constitute structure and actual musical content. Based on that, paper [16] refers to the music as highly structured content. The structured content can be categorized into repetitions, contrast, variations, and homogeneity. The methods of finding the combinations among the structure content are called Music Structure Analysis [16]. Typically, music structure consists of a large number of common patterns that are known as the repetitions [16]. Repetitions, as one of the most critical concepts indicating the rhythmic and harmonic patterns as well as variations information behind the music, show periodicity information in a piece of music [12]. As a consequence, the repetition-based structure analysis approach is mostly used for detecting the iterated patterns from the music [16].

The music structure can be extracted in a self-similarity based matrix which is used to present mid-level representations [18] of the music. The row and column in the self-similarity matrix correspond to the segmented frames of a single recording. The value in the matrix is calculated through dissimilarity/similarity distance between segmented frames. Once the self-similarity matrix has been constructed, the informative representations in the matrix can be determined. Furthermore, transferring the self-similarity matrix into an image can help us to distinguish the structured content in the music. While the strips in the image show the repetitions, the blocks in the image refer to the homogeneity in the music.

2.2. Match Similar Music Segments

Repetitions are the music segments that share levels of similarities. In the field of MIR, three topics are used to find similar music segments or recordings, namely Music Fingerprinting, Cover Song Detection, and Audio Matching. The Music Fingerprinting invented by Wang et al. in [23] can quickly identify the song in the database by using only tiny music excerpts recorded through a microphone in the cellphone. Cover Song Detection, also known as version identification, can be solved above the tonal or timbre content of the music [2]. The paper [8] written by Liem et al. presents a Cover Song Retrieval(CSR) system which used raw audio as a query to retrieve a different version of the music in the dataset. The system contains two main components, namely feature representation and dissimilarity assessment. Feature representation is the process of transferring an audio signal into feature vectors [8]. Dissimilarity assessment, the other component in her CSR system, measures dissimilarity value between two feature vectors [8]. The Audio Matching is a MIR topic to find the similarity among audios. Audio Matching is also called music synchronization in [15], is aimed for

aligning music recordings to another recording, music scores or MIDIs. Müller et al. proposed a new audio matching technique to align two music recordings by using the dot product to calculate dissimilarity value. After that, two music segments in the audios with the minimum dissimilarity value are considered as the best match.

2.3. Feature Extraction

Feature extraction is a common step we need to take when describing the content of the audio signals. It can transfer the low-level audio signal into feature vectors that can be used for further analysis. There are many feature representations in music, such as dynamics, timbre, chroma, and local tempo variation. The chroma and timbre features are two of the essential features that are widely used in MIR tasks. Mel-frequency Cepstral Coefficients (MFCCs) is normally used as timbre feature that has excellent performance in speech processing task [4]. A potential application of MFCC suggested by Foote is to compute the novelty of the music [5]. Chroma feature is firstly mentioned in [1]. It shows spectral energy within 12 pitch classes with an equal-tempered scale. To make chroma more suitable for distance measuring tasks, Müller et al. proposed a normalized version of chroma feature named Chroma Energy Distribution Normalized Statistics (CENS) [14]. The CENS feature is stabilized to variations when analyzing classical western music that is highly related to the harmonic progression.

2.4. Rehearsal Analysis System

The rehearsal analysis system is a framework that users can monitor the progress of their rehearsal session by listening and visualizing repetitions in the user interface.

To the best of our knowledge, only three rehearsal analysis systems have been mentioned in the literature. The first system is proposed by Xia et al. which can record, organize, retrieve and review the repetitions in the rehearsal audio [25]. The silent music segments are removed from the rehearsal recordings by using the Ada-boost classifier before they get into next stage of analysis. The non-silent music segments remains as independent music segments. The music segments are transferred into CENS feature vectors [15]. Moreover, the audio matching method proposed by [15] is used to find similar music segments in the recordings. As a consequence, the users can visualize and evaluate the repetitions in the user interface in the system.

Another rehearsal analysis system proposed by Winter et al. in [24] is named as automatic logging system. Unlike the silence removal method proposed by Xia et al. [24], a pre-defined threshold is set in the recorder before the rehearsal session starts. The energy of the input lower than the pre-defined threshold will not be recorded. After that, the rehearsal recordings are transferred into the CENS feature vector. Unlike in Xia et al. 's paper [25] , Winter et al. [24] segmented the feature vectors equally. Once the segmentation have been done, Dynamic Time Warping (DTW) is used as a distance function to determine whether two feature vectors are repetitions or not. The rehearsal recordings is aligned with reference recordings that are used as ground truth to evaluate the repetitions extracted from rehearsal analysis system.

There are both pros and cons in both rehearsal analysis framework. Although silence removal task in Xia et al. 's paper [25] have better performance comparing to Winter et al. in paper [24], it requires many manual works on labeling the silent and non-silent music segments. Moreover, despite Xia et al. 's system has resulted longer music segments that are more listenable comparing to Winter et al. 's system, many false matches are contained in the repetitions in Xia et al. 's system.

Recently, rehearsal analysis has been proposed again by The in his thesis [22]. The system uses the same segmentation and audio matching techniques mentioned by Xia et al. in [25]. The system designed by The is called rehearsal progress monitoring [22] that can interact with users by visualizing the most frequently played music segments in the rehearsal recordings. The repetition output is evaluated manually by listening. We want to emphasize that the user interface in The's thesis [22] is the first time that user interface has appeared among all the rehearsal analysis system.

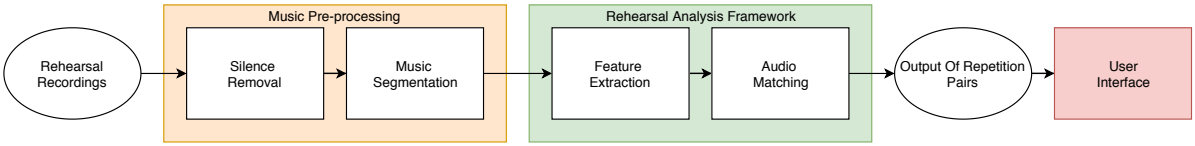

Figure 2.1: Rehearsal analysis system in [22, 25].

2.5. Segmentation Evaluation Method

A rehearsal analysis system is successful when it manages to extract meaningful repetitions; that is, repetitions are segmented at an appropriate time resolution. If we attempt to evaluate repetitions, the closest existing works deal with the evaluation of structural segmentation outcomes, taking a different aspect of the segmentation into account. The paper [21] comes up with the topic Music Information Retrieval Evaluation eXchange (MIREX) which published common metrics that are mostly used to evaluate the algorithms in structure segmentation tasks. The first matrix is called as pairwise retrieval matrix. This matrix includes precision, recall, and f-measure as proposed in [6]. The precision and recall are calculated through dividing the number of correct segmentation by the total number of the ground truth of segments or by the number of predicted segmentation, respectively. The score of F-measure is weighted for both. Except for measuring the number of correct segmentation, the boundary retrieval evaluation calculates the percentage of time overlapping between ground truth and predicted repetitions, as described in [21]. Levy et al. [7] uses the concept of boundary retrieval evaluation to measure the percentage of the missing boundary between predicted segments and the total ground truth of segments by the length of time. Furthermore, Lukashevich et al. proposed concept of under-segmentation and over-segmentation conditions for the purpose of quantifying the level of segmentation.

The evaluation method described in paper [21] requires ground truth of repetitions that we do not have in our rehearsal data. However, in the future chapter, we will propose a synthetic data generator framework in which we synthesize rehearsal type of data with such ground truth data. Although the ground truth helps us to evaluate the repetitions, there exists anonymous (or named as ambiguously) of ground truth that introduces difficulties to explain the correctness of the designed algorithms [21]. In the meanwhile, the paper [19] illustrates that “the number of patterns found by the algorithms exceeds patterns in human annotations by several orders of magnitude, with little agreement on what constitutes a pattern.”, which means the ground truth of common patterns or structure extracted by algorithms is far more different from the ground truth labeled by humans. To sum up, the ambiguity of ground truth is one of the biggest difficulties in MIR evaluation tasks.

3

Rehearsal Analysis Overview

To solve the research questions mentioned in Chapter 1, we firstly proposed a rehearsal analysis framework utilizes segmenting, clustering, and merging the segments to extract listenable and musically meaningful repetitions within the rehearsal recordings. As we mentioned in Section 2.5, to tackle the missing ground truth in rehearsal recordings, synthetic data generator can create synthesize data which contains with ground truth. In the end, evaluation methods are used to evaluate the performance of rehearsal analysis system by using synthesize data.

This Chapter is organized as following. Section 3.1 gives the definitions of repetitions in the rehearsal recordings. The Music pre-processing step in Figure 3.1 will be described in Section 3.2 and Chapter 4. Rehearsal analysis framework in Figure 3.1 will be described in Section 3.3 and Chapter 5. Synthetic data generator in Figure 3.1 will be discussed in Section 3.4 and Chapter 6. Evaluation in Figure 3.1 will be discussed in Section 3.5 and Chapter 7.

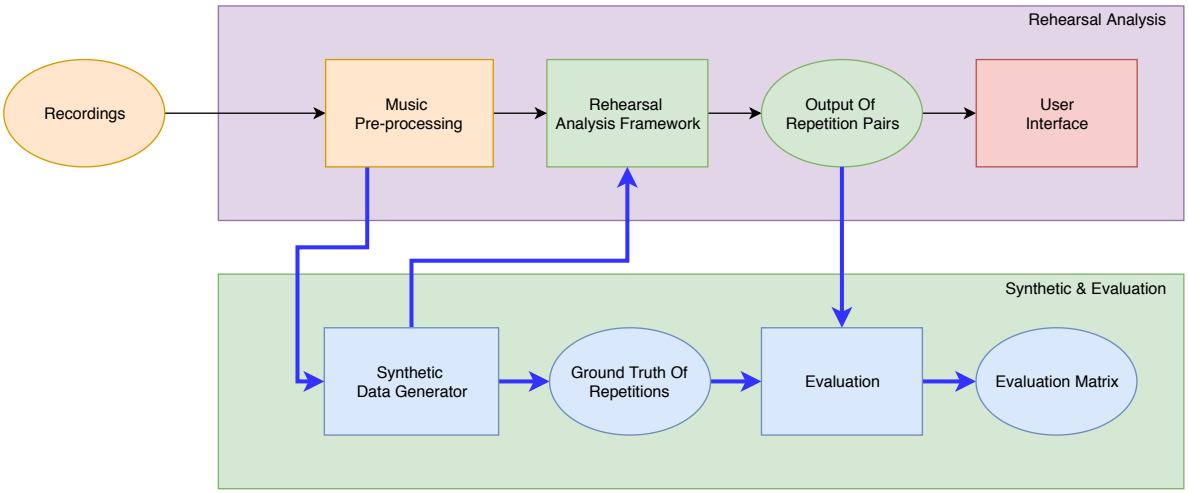

Figure 3.1: This Figure shows two workflows in the rehearsal analysis approach. We proposed the synthetic data generator and evaluation as it shown in the blue line. Our rehearsal analysis approach contain every elements in this figure except for the user interface. Section 3.2/Chapter 4 for music pre-processing; Section 3.3/Chapter 5 for rehearsal analysis framework; Section 3.4/Chapter 6 for synthetic data generator; Section 3.5/Chapter 7 for evaluation.

3.1. Definition Of Repetitions In Rehearsal Recordings

Repetitions differ in length of time, which have been extracted by using audio matching method proposed by Müller et al. [15]. The order of harmonic content is not considered as criterion ot determine whether two music segments are repetitions. As it is shown in Figure 3.2, the music fragment 1 and 2 are both considered as repetitions in the music by using audio matching method [15], however, we have found that the repetition pair in music fragment 2 should not be considered as repetitions. Therefore we define the repetitions in the rehearsal recordings: two music segments have identity or similar order of harmonic content with containing acceptable variations can be as a repetition pair.

To let readers have clear understanding of what is the variations in the repetition pair, we drew Figure 3.3. In Figure 3.3, the square red box shows music segment that are not repeated over time as well as is 'small enough' to be ignored, given that the surrounding context repeats in the same order. As for what is still 'small enough', this is where a **tolerance** level can be defined. Except for the fact that repetitions may contain different length of variations, it might also vary in time resolution which shows in Figure 3.4.

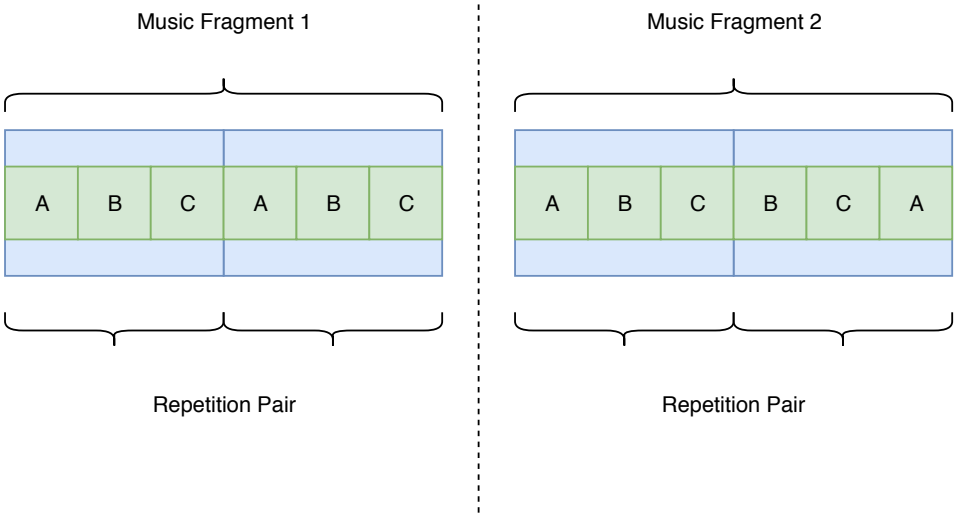

Figure 3.2: The repetition pair in both music fragment 1 and 2 are determined as repetitions by using audio matching technique in [15]. However, we think that the music fragment 2 are not consider as repetitions.

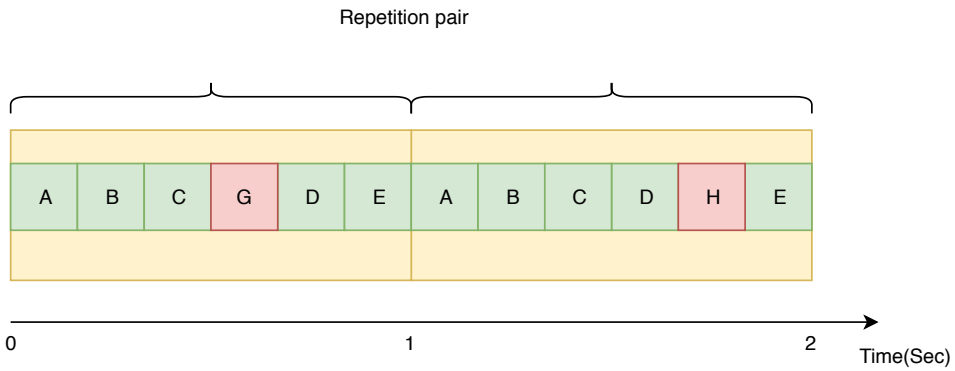

Figure 3.3: The large repetition pairs are shown in orange boxes in the rehearsal recordings. In this example, since there are many small repetitions in the repetition pair, we are only consider the repetition pair with one second long for each segment. The small red square box in the figure shows the length of variations within the repetition pair and the length may vary.

3.2. Music Pre-Processing

The function of music pre-processing is to remove useless information(here we mean silent music segments) from the recordings and to transfer low-level music signal into feature vectors which can be used in further analysis. The music pre-processing step in this thesis is shown in Figure 3.5.

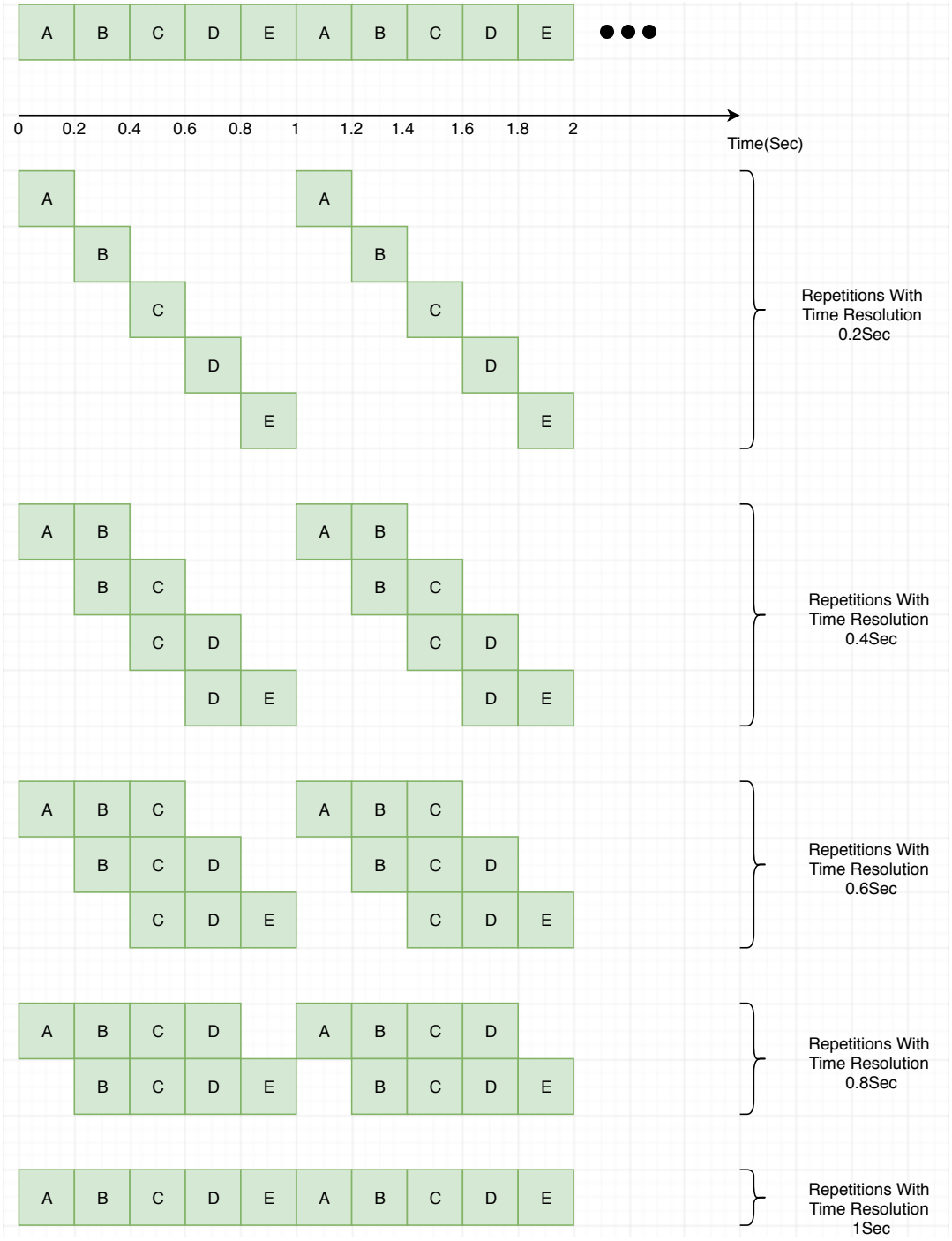

Figure 3.4: Repetitions with different time resolutions.

3.2.1. Silence removal

The repeating silent music segments are useless in practical applications. The music educators will not expect silent music segments as informative representatives appearing in the output of rehearsal analysis system. The way of solving this issue is to remove silent music segments from the music array before it goes into rehearsal analysis framework.

We are using both 'Normal' and real rehearsal recordings in this thesis. The 'Normal' recordings is the music recordings that strictly followed the written music score without containing error and experimentations. The silent segments in 'Normal' recordings is removed after the music has been recorded. Thus we only have to remove silent segments in real rehearsal recordings.

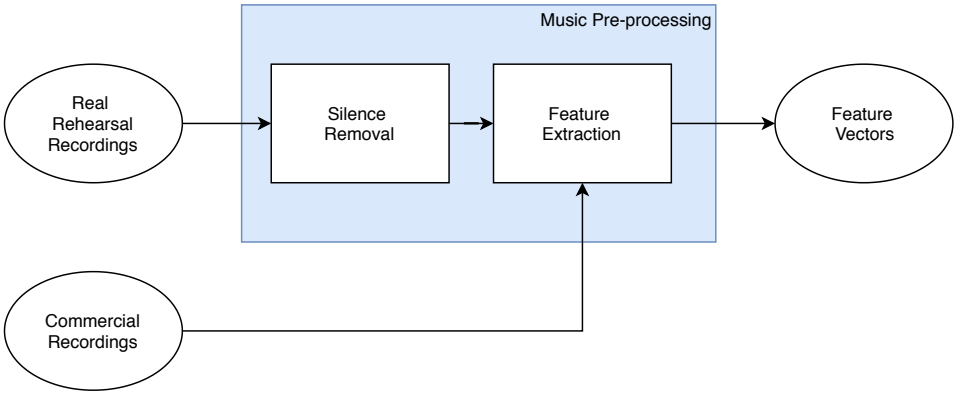

Figure 3.5: Pre-processing step in the rehearsal analysis approach. The box in blue is the pre-processing step.

3.2.2. Feature extraction

Once the silent is removed from the recordings, the recordings array is required to be transferred into useful feature representatives as an essential step to describe musical content. Since our data is western solo piano works that are strongly correlated to harmonic progression, chroma feature has representation of those works. The goal of feature extraction is to transform audio signal into feature vectors that can be used for measuring the level of similarities between music segments.

3.3. Rehearsal Analysis Framework

Once we obtain feature representations from the rehearsal recordings, we transform feature vectors into listenable repetitions for the use of musicians which are shown in 3.6. The rehearsal analysis framework consists of three steps, features windowing, features clustering and small to large segment merging.

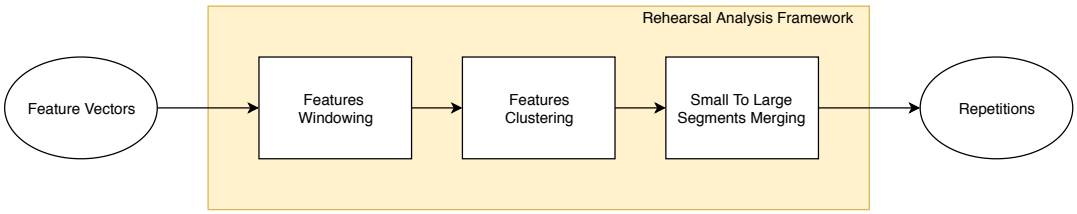

Figure 3.6: The yellow box is the workflow of the rehearsal analysis framework. The input of the framework is the chroma feature vectors, it can be fully, semi synthetic feature vectors or feature vectors from real rehearsal recordings.

3.3.1. Features windowing

The feature extraction in the pre-processing step changes audio array into feature vectors. It will lead to a substantial computational time if each feature vector is used in pairwise distance measurement. Feature windowing step groups consecutive feature vectors which yields the decreasing of the number of distance pairs, therefore reduce the computational time.

3.3.2. Features clustering

Extracting repetitions in the music requires grouping feature vectors according to their distance value. The distance is measured through distance function, and the distance threshold is used to determine whether two feature vectors belong to same cluster or not.

3.3.3. Small to large segment merging

Once each feature vectors have been clustered, each feature vectors containing with labels are merged into longer and listenable repetitions by using segment merging algorithms with acceptable variations.

3.4. Synthetic Data Generator

Regarding evaluating the performance of the rehearsal analysis framework, the ground truth of repetitions is necessary. However, it has been illustrated in Chapter 1 that labeling repetitions in real rehearsal rehearsal is an expensive and almost an unapproachable task. In this work, we propose a synthetic data generator which can generate two types of synthetic data, namely **Fully-synthetic** and **Semi-synthetic** data with automatically created ground truth of repetitions. **Fully-synthetic** data is created by extracting, modifying, and concatenating music segments that have fully controllable ground truth. **Semi-synthetic** data is created through extracting, modifying the music segments and inserting those segments back into recordings with partially ground truth.

3.5. Evaluation Of Different Synthetic Data

With the ground truth of repetitions from the **Fully-synthetic** and **Semi-synthetic** data, we could evaluate the performance of rehearsal analysis framework by using synthesize data. We look at evaluation by two methods. The first evaluation method calculates the percentage of correctly predicted repetition pairs over either the ground truth of repetitions or the total repetition pairs. The second evaluation method measures the percentage of the time-overlapping among those corrected prediction of repetitions over the total length of the ground truth of repetitions.

4

Music Pre-processing

The music pre-processing step removes unnecessary silence music segments from the rehearsal recordings and transfers the low-level audio array into feature representatives which are the input of rehearsal analysis framework. Section 4.1 introduces the silence removal method to remove useless silent repetition pairs from the rehearsal recordings, and Section 4.2 describes the feature extraction methods that transfer the low-level audio signal to feature representations.

4.1. Silence Removal Method

There are two silence removal methods mention in literature by Xia et al. and Winter el al. . Although the first approach gives a superior result of silence and non-silence classification, it still requires users to provide ground truth of silent and non-silent segments in the recordings. This is an expensive and time-consuming task for long rehearsal recordings. The second method is much easier to implement without any ground truth of silent segments in the recordings. However, the shortcoming of this approach is that, finding the appropriate threshold to distinguish the silent and non-silent segments requires many experimentation for adapting different environment. In our music pre-processing task, we decided to apply the heuristic approach used in The's thesis [22]. The energy threshold for distinguishing silent and non-silent music segments has been set to 20db, and it has the best result of separating silent and non-silent music segments.

4.2. Feature Extraction

Musics share with level of similarities. Those similarities can not be easily discovered through one-dimensional audio array. For the purpose of finding the similarities between musics, useful music feature representatives are firstly extracted from the recordings.

4.2.1. Chroma Feature

Since our analyzed data is western solo piano works that are strongly correlated with the harmonic progression, the chroma feature is the most appropriate feature to represent the data and it is widely used in similarity/dissimilarity measurement between two feature vectors. Each chroma feature has twelve different pitch classes over time. 12 pitch classes stand for the frequency range of the 12 keys in the middle of the piano.

The chroma feature is robust to timbre changes in different instrumentation. The way of getting chroma feature describes in paper [15]:

1. Changing audio signal into 88 frequency bands related to the musical notes from A0 to C8(pitch level from p=21 to p=108). A elliptic filter is used with excellent cut-off properties to separate adjacent notes.

2. Calculating the short-time mean-square power (STMSP) in each band through 200ms rectangular window that has half size of signal overlap.

3. Computing short-time-mean-square in each pitch class and adding up into chroma classes, so that a 12-dimensional feature vector is created for each window.

4. Each energy distribution relative to the 12 chroma classes is calculated via dividing 12 feature vectors by the sum of energy distribution.

4.2.2. CENS Feature

Due to the sensitivity in articulation and tempo variance of energy distribution in 12 chroma classes, a normalized chroma feature named Chroma Energy Distribution Normalized Statistics (CENS) feature has been proposed by müller et al. [15]. The CENS feature is build based on the chroma features by adjusting two extra steps, namely quantization and smoothing. Feature vectors are firstly normalized between zero to one by using the Manhattan norm (l_1 norm). Then the quantization process transforms the intensity value in chroma feature into integers from zeros to four based on four different levels. Once the chroma vector is quantify, a Hann window function is used to smooth out the value in those chroma vectors. In the end, the intensity value in chroma features are downsampled as factor of 10 and normalized by euclidean norm (l_1 norm) [15].

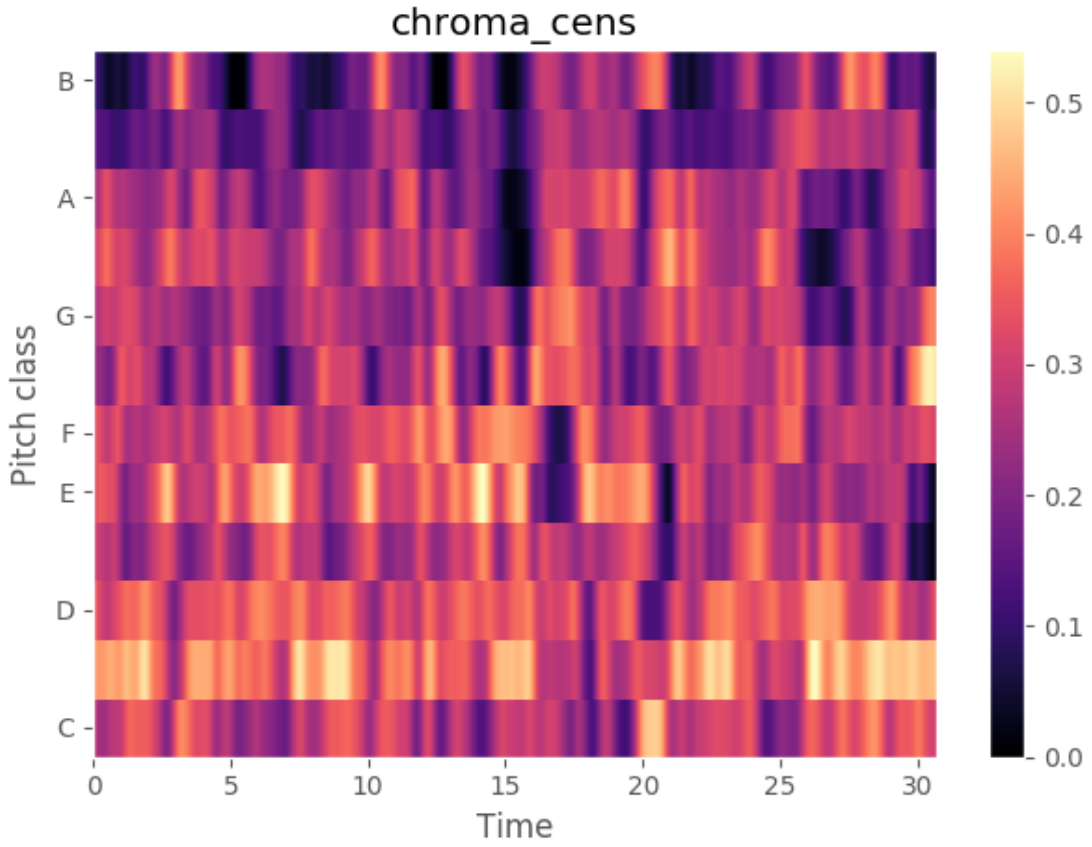

Figure 4.1: This is the CENS feature for 31 seconds of the rehearsal recordings. The vertical axis represents twelve pitch classes with pitch level {C, C#, D, D#, E, F, F#, G, G#, A, A#, B}. Different color refer to the intensity value in different pitch class. For example, pitch-class C is the sum of all intensity value related to C key in full piano scales.

5

Rehearsal Analysis Framework

We designed a rehearsal analysis framework including feature windowing, clustering and segments merging steps to automatically extract listenable repetitions from the rehearsal recordings. Section 5.1 describes the feature windowing step through grouping consecutive feature vectors. Section 5.2 describes the feature clustering step. Section 5.3 illustrates the segment merging step.

5.1. Feature Windowing

In this thesis, we decided to group consecutive feature vectors to decrease total number of feature vectors in order to decrease the processing time. The parameter n is the number which is used to group consecutive feature vectors. For instance, if the frame is length 200ms and n is 5, it means 5 consecutive feature vectors are grouped into a single feature vector with a duration of 1 second.

5.2. Feature Clustering

Feature clustering is the unsupervised approach of grouping feature vectors based on their distance. The label is given to each feature vectors. There are two popular ways in unsupervised clustering, either setting the value of threshold such as hierarchical clustering method or choosing numbers of clusters such as K-means clustering method.

5.2.1. Distance function used in clustering

The dissimilarity distance between feature vectors is used in clustering step. There are two general ways of calculating the distance between two feature vectors, namely the cosine distance and euclidean distance. We choose euclidean distance as our distance function. The reason is shown in Figure 5.1, where it shows that cosine distance is sensitive to the angle between two feature vector instead of value in the feature vector, while the euclidean distance is opposite. The function of distance is calculated in Formula 5.1. There exists two consecutive chroma vector X and Y and the distance feature vector X = (x_1, x_2, x_3,...,x_n) and Y = (y_1, y_2, y_3,...,y_n), where x and y is the single feature vector groups with n consecutive feature vectors. x_n represents a CENS feature vector with 12 pitch class. The euclidean distance with normalization between two feature vector is given by

$$d(X,Y) = \frac{\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\cdots+(x_n-y_n)^2}}{\sqrt{12}\times\sqrt{n}} = \frac{\sqrt{\sum_{n=1}^{n}(x_i-y_i)^2}}{\sqrt{12}\times\sqrt{n}} \tag{5.1}$$

5.2.2. K-Means

K-Means clustering is a vector quantization methods. If there exists a observation set (a_1, a_2, a_3,..., a_n), each observation is a 2-dimensional feature vector. The K-Means algorithm is to assign n numbers of observation into K numbers of clusters where the number of K is small and it equals to the total number of observations. Once the cluster is assigned for each observations, it has to be sure that the sum of square distance within

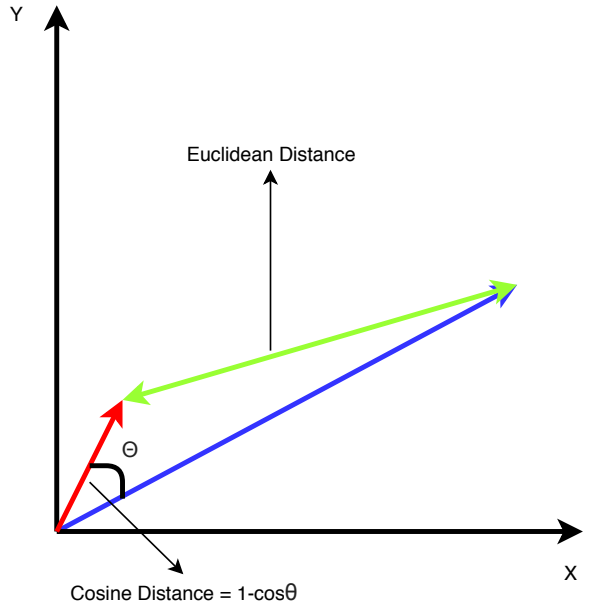

Figure 5.1: The red arrow is the feature vector A and blue angle is the feature vector B. The angle between two feature vectors is θ. The green line is the euclidean distance between two feature vector. The cosine distance is one minus cosine value between two feature vector.

clusters S_i has the minimum value. The square distance within clusters can be calculated by equation 5.2.

$$\sum_{i=1}^{k} \sum_{x \in S_i} \| x - \mu \|^2 \tag{5.2}$$

Firstly, randomly chose K number of points from dataset and assign each point into an independent clusters. Secondly, for the rest of points, do not assign into any of the clusters and calculate the distance with the points which already have a cluster. Assign the rest of points with the closest distance value with the assigned clusters iteratively. After every point have already been assigned for a cluster, calculate the average point for each cluster and update new clusters for all observation until the assignment for the observation does not change.

The K-Means algorithm is easy and fast to implement, and it can obtain a pretty tight result. However, K-Means is very sensitive to scaling as the outliers in the dataset may heavily influence the result of the cluster. The real rehearsal analysis data is messy, unstructured and it contains many experimentation, which makes it hard to choose the right number of clusters to describe the inner structure of the recordings.

5.2.3. Hierarchical Clustering

Hierarchical clustering is a clustering method which constructs a hierarchy representation of the data. There are two types of hierarchical clustering methods exists in literature [20], agglomerative and divisive hierarchy clustering. Agglomerative clustering, one of the hierarchical clustering methods, starts from many single clusters until all of the clusters are merged into one cluster. Divisive clustering is a clustering methods that split one cluster into many clusters to the bottom of the hierarchy. There are variety of rules to merge the segments between different observations such as complete, single, and average linkage according to the distance measured between music segments. The output of those clustering methods has a graphic representation called dendrogram[20]. In the end, clusters are created by cutting dendrogram with a given threshold value. The structure information in rehearsal recordings are not clear so that it is hard to choose the appropriate linkage methods as well as the value of cutting threshold to group meaningful clusters.

5.2.4. Customize Clustering Algorithm

To find groups of meaningful clusters from unstructured and messy rehearsal recordings, we proposed a customized clustering algorithm which could effectively cluster feature vectors. The first step of the clustering algorithm is to build a self-similarity metrics. A distance distribution $d_i[n]$ can be calculated through each

feature vector with the rest of the feature vectors show in Figure 5.2, where the n represents the index of the feature vectors.

Once the distance distribution is built for each feature vector, the local minimum points in distance distribution $d_i[n]$ are considered as the repetitions [3]. However, it is unrealistic that repetitions are too close together. The parameter **Order** is used to control the number of total points on determining whether the local minimum point in the distance distribution $d_i[n]$ are repetitions or not. In case many local minimum points are in the range of **Order**, only the local minimum with the smallest value will be kept and others will not be considered as repetitions. The green box with label 1 and 2 in Figure 5.3 and 5.4 show that local minimum points within the range of order number have been filtered out.

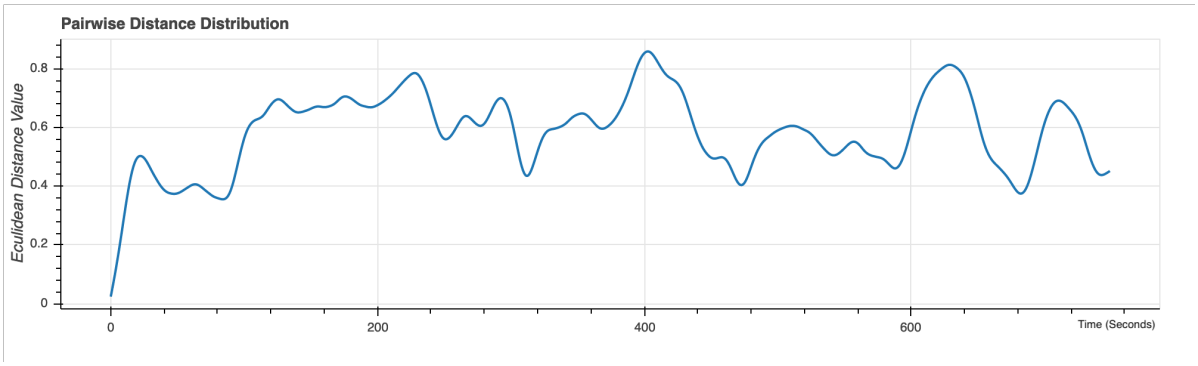

Figure 5.2: The distribution of $d_i[n]$.

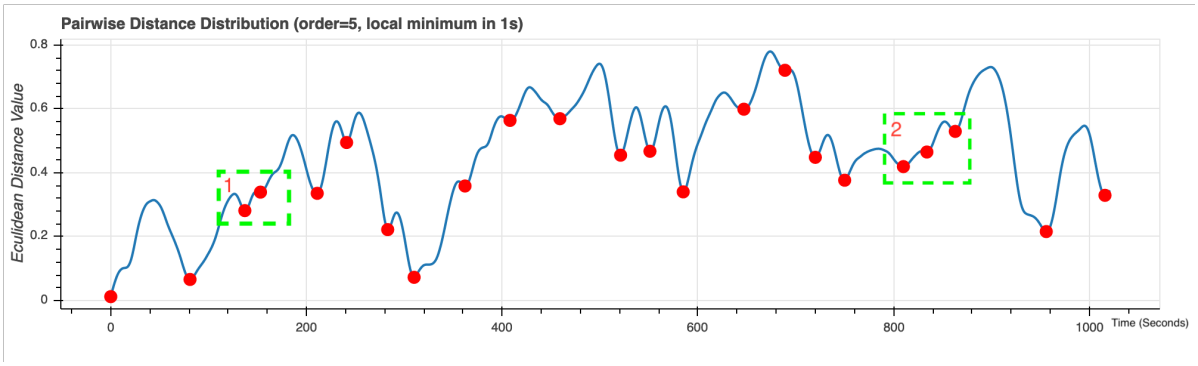

Figure 5.3: The distribution of $d_i[n]$ and red points mean the local minimum points. The missing points in the green box are the points being filtered out by order number.

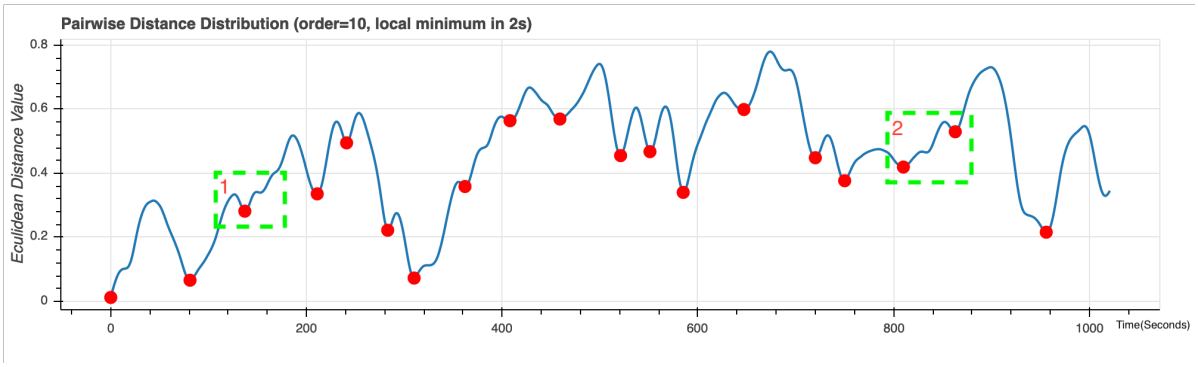

Figure 5.4: The distribution of $d_i[n]$ and red points mean the local minimum points. The missing points in the green box are the points being filtered out by order number.

The pseudo-code of customized clustering method is shown in Algorithm 1. Firstly, if both segments are not assigned into any class, assign both of them with a new cluster. If one segments has been assigned into class, and the other segments has not, assign both segments into the same class. When both segments

Algorithm 1: Customize Clustering Algorithm

Input:
D_p: Dictionary with (Key: Time intervals pairs (t_i, t_j)) and (Value: pairwise distance (d_p))
Output:
C_l: Dictionary with (Key: Time intervals t_i, t_j, ...)
and (Value: tuple(clusters label l, best match distance d_p))

```
Initialize C_l with the set of keys in D and value with the cluster label c_l, each c_l equals to zero
for each items in D_p do
    if D_p[(t_i)] == D_p[(t_j)] ≠ 0 then
        C_l[(t_i)] = C_l[(t_j)] = c_l
        Save d_p in both C_l[(t_i)] and C_l[(t_j)]
        c_l = c_l + 1
        break
    else if D_p[(t_i)] == 0 and D_p[(t_j)] ≠ 0 then
        C_l[(t_i)] = C_l[(t_j)]
        Save d_p in C_l[(t_i)] and C_l[(t_j)]
        break
    else if D_p[(t_j)] == 0 and D_p[(t_i)] ≠ 0 then
        C_l[(t_j)] = C_l[(t_i)]
        Save d_p in C_l[(t_i)] and C_l[(t_j)]
        break
    else if D_p[(t_j)]≠ 0 and D_p[(t_i)] ≠ 0 then
        if D_p[(t_i)] > D_p[(t_j)] then
            C_l[(t_j)] = C_l[(t_i)]
            break
        else
            C_l[(t_i)] = C_l[(t_j)]
            break
        end
    else
        Pass
    end
end
return C_l
```

have already been assigned into a class, compare the distance of the best match of both feature vectors and assigned the feature vectors with smaller value of distance.

5.3. Small To Large Segments Merging

The function of small to large segment merging is to merge short repetition pairs into longer repetition pairs that are listenable in practice.

The segment merging algorithm contains three parts. The first part creates pairwise repetition pairs in the group of repetitions. The second part connected the consecutive pairwise repetition pairs with same semantic label. The third part merges consecutive repetitions into even longer repetitions if the time difference between two repetition pairs is smaller than the tolerance.

To connected the consecutive pairwise repetition pairs, we use time-lag format, a type of data structure that could represent the similarity relationship between two music segments, to speed up the process of the second part of segment merging algorithm. Figure 5.5 represents the repetition pair into a tuple (t_1, t_2, shifting). The shifting of two repetitions are calculated by t_3 - t_1.

By using the time-lag format, two repetition pairs can be merged into longer repetition pair if:

1. Both repetition pairs have the same number of shifting.
2. The time differences between two time-lag format should be the same.

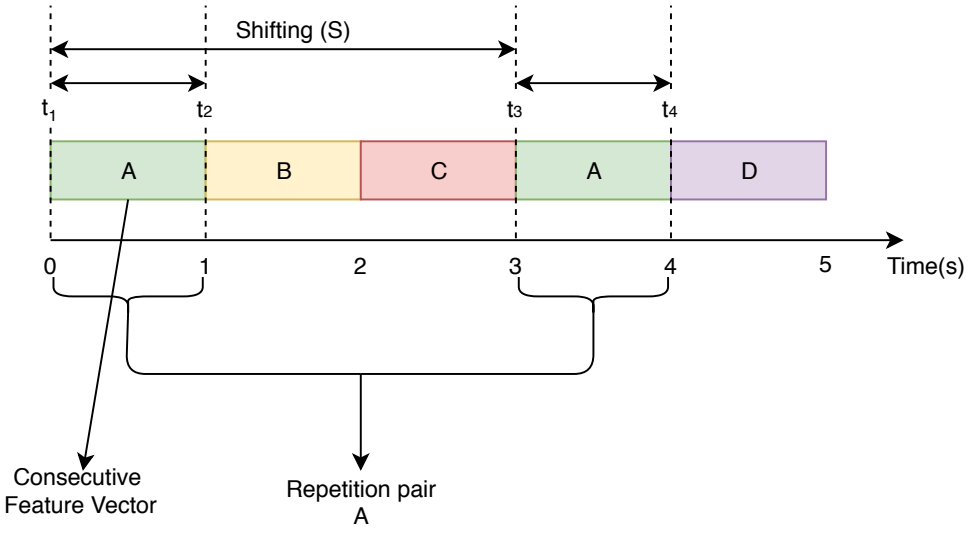

Figure 5.5: The blocks show the feature vectors and the different color mean the relationship between features vectors. The blocks with the same color means the feature vectors are grouped into the same cluster for example repetitions in green box.

3. Repetition pairs should not overlap in the time domain.

Due to the bad structure of real rehearsal data, the result of repetition pairs might still be short and not listenable. This can be solved by merging with tolerance in the third part of segment merging algorithm. The merging with tolerance algorithm is shown in Algorithm 3 and a example of segment merging algorithm is shown in Figure 5.7. There are three cases with different locations of noise. Those noises can appear in both music segments as shown in case *C*, or in single music segment as shown in the case *A* & *B* in Figure 5.6.

Algorithm 2: Connect continuous identical repetition pairs

Input:
l_t: List with the time lag format$((t_1, t_2, S_1), (t_1, t_2, S_1), \ldots$
Output:
l_c: List of consecutive repetition pairs in time lag format

```
Sort l_t descending
for i from 0 to len(l_t) do
    for j from i to 0 do
        Sort l_t descending order
        if Shift value in l_t[i] is equal to l_t[j] and l_t[i][1] equals to l_t[j][0] then
            l_t[j] = (l_t[j][0], l_t[i][1], l_t[j][2])
            l_t[i] is set to None
            break
        else
            Pass
        end
    end
end
Remove element in l_t equal to None
return l_c
```

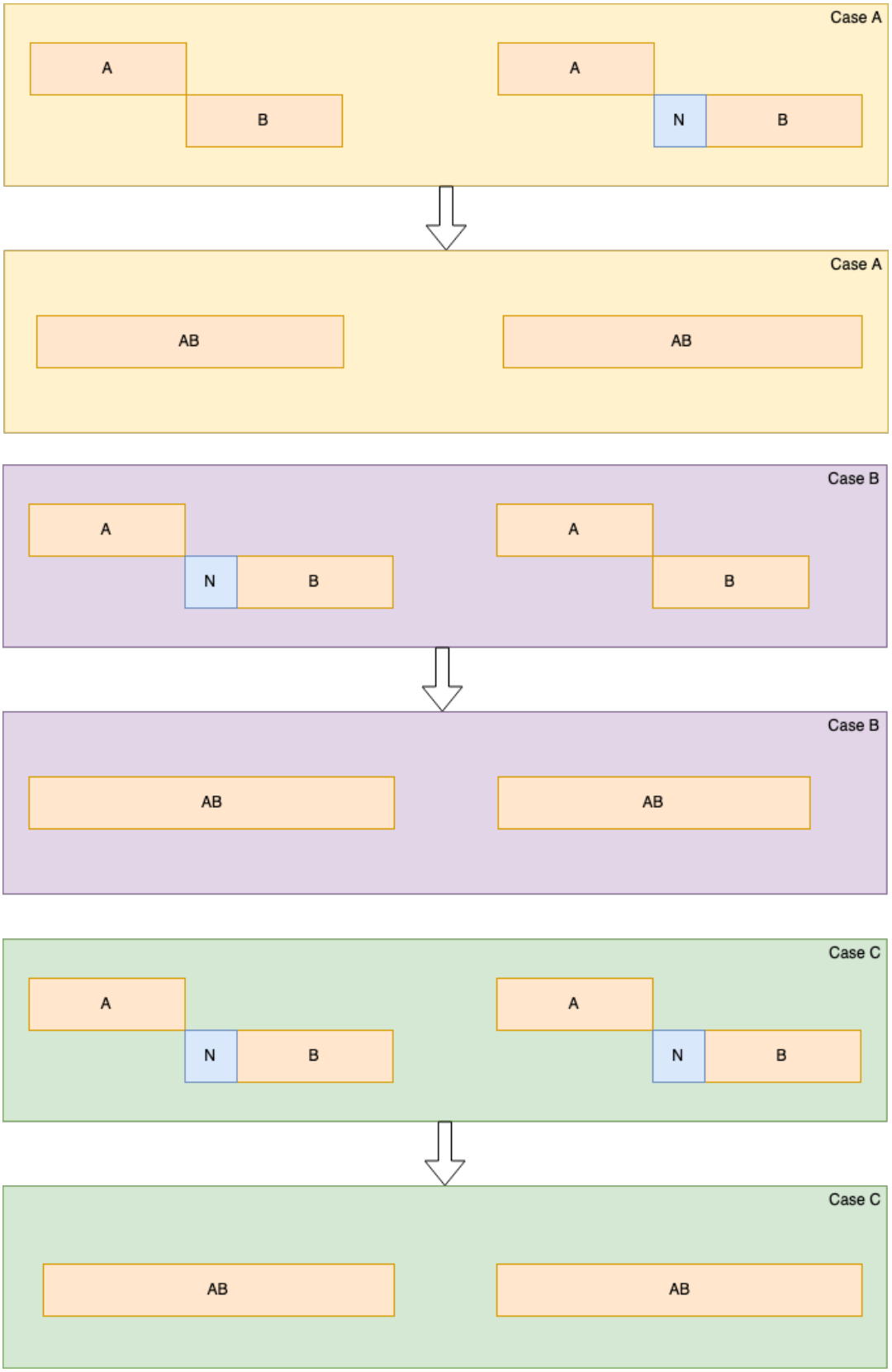

Figure 5.6: This figure shows the process of merging two short repetition pair into longer repetitions with containing noises among repetition pairs. The duration of the noises might different.

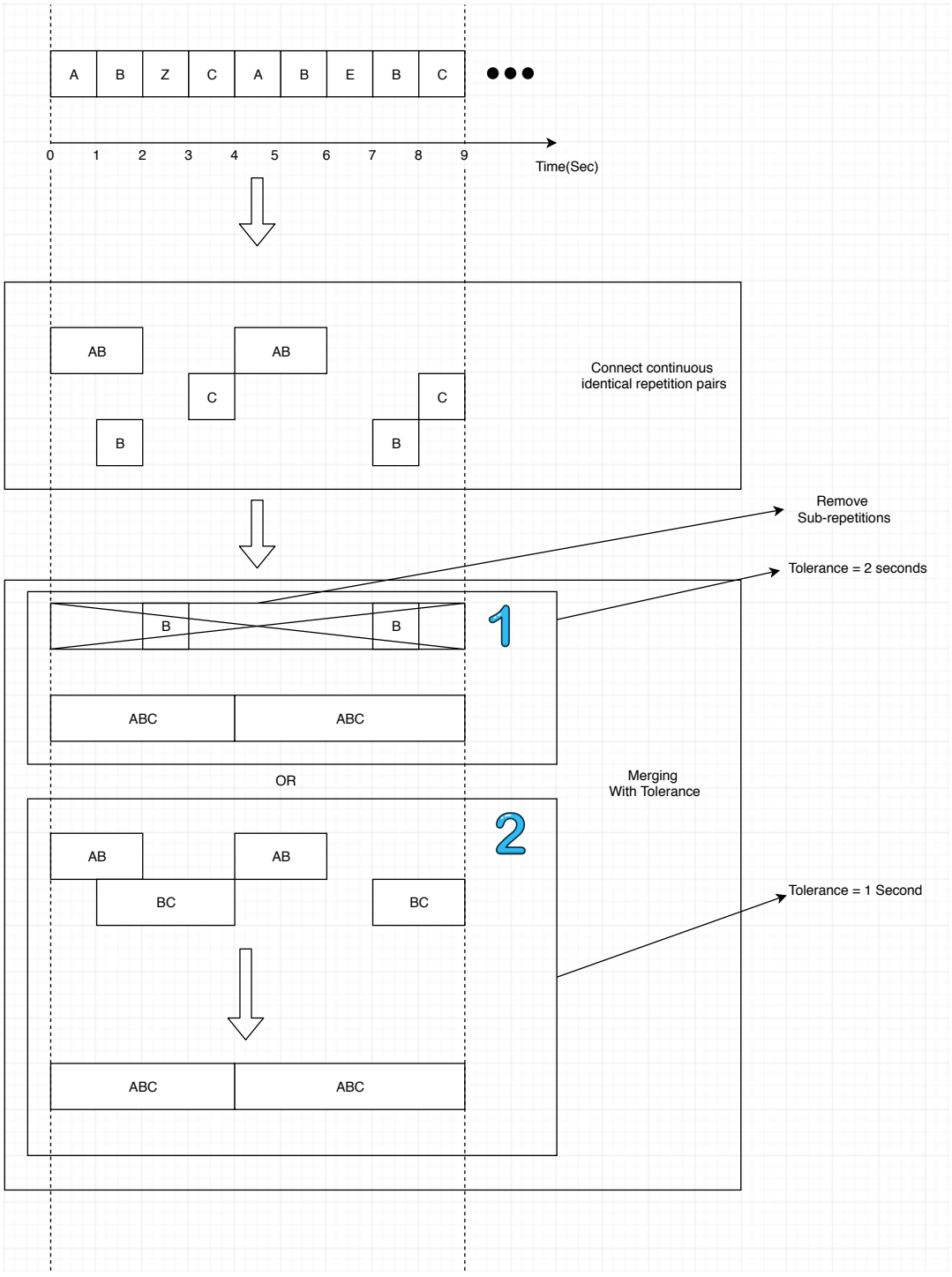

Figure 5.7: Segment merging algorithm.

Algorithm 3: Merging with tolerance algorithm

Input:
L_{rp}: List with the repetition pairs with time intervals(t_{i1}, t_{i2}), (t_{i3}, t_{i4}), ...
T: Number of tolerance to merge two repetition pairs
Output:
L_{rp}: List with connecting the time lag format

```
Sorting L_rp descending
for i From len(L_rp) To 0 do
    for j From i To 0 do
        if t_i1 is not overlap with t_i3 and t_i2 is not overlap with t_i4 then
            if L_rp[i][0][0] - L_rp[j][0][1] <= T and L_rp[i][1][0] - L_rp[j][1][1]<=T and
              L_rp[i][0][1]<=L_rp[j][1][0] then
                L_rp[j] = ((L_rp[j][0][0], L_rp[i][0][1]), (L_rp[j][1][0], L_rp[i][1][1]))
                L_rp[i] is set to None
                Sorting L_rp descending order
                break
            else
                Pass
            end
        else if exist one t_i1 overlap with t_i3 then
            if L_rp[i][1][0] - L_rp[j][1][1]<=T and L_rp[i][0][1]<=L_rp[j][1][0] then
                L_rp[j] = ((L_rp[j][0][0], L_rp[i][0][1]), (L_rp[j][1][0], L_rp[i][1][1]))
                L_rp[i] is set to None
                Sorting L_rp descending order
                break
            else
                Pass
            end
        else if exist one t_i2 overlap with t_i4 then
            if L_rp[i][0][0] - L_rp[j][0][1] <= T and L_rp[i][0][1]<=L_rp[j][1][0] then
                L_rp[j] = ((L_rp[j][0][0], L_rp[i][0][1]), (L_rp[j][1][0], L_rp[i][1][1]))
                L_rp[i] is set to None
                Sorting L_rp descending order
                break
            else
                Pass
            end
        else if t_i1 is overlap with t_i3 and t_i2 not overlap with t_i4 and L_rp[i][0][1]<=L_rp[j][1][0] then
            L_rp[j] = ((L_rp[j][0][0], L_rp[i][0][1]), (L_rp[j][1][0], L_rp[i][1][1]))
            L_rp[i] is set to None
            Sorting L_rp descending order
            break
        else
            Pass
        end
    end
end
Remove sub-repetitions
Remove element in L_rp equal to None
return L_rp
```

6

Synthetic Data Generator

We generate a synthetical group of repetitions which have an identical or similar degree of variations as we expect in reality and the location of repetitions in the synthetic data can be used as ground truth to evaluate the performance of rehearsal analysis framework. This chapter dissembles the process of creating such synthetic data. Section 6.1 introduces the way of managing those musical content into different types of synthetic data. Section 6.2 introduces the musical content that we are going to repeat in the synthetic data. Section 6.3 introduces how to copy and modify those musical content with variations for creating repetition pair that most similar to real rehearsal situations. Section 6.4 introduce the ground truth of repetitions in different types of synthetic data.

6.1. Fully-synthetic & Semi-synthetic Data

The different way of concatenating or inserting copies can create **Fully-synthetic** and **Semi-synthetic** data respectively. We mentioned in the Chapter 3 that **Fully-synthetic** is created by concatenating the modified music copies, however, the well-performance of the rehearsal system in the **Fully-synthetic** audio data does not mean the system could perform as well as in the real rehearsal audio data. To further test the performance of the rehearsal analysis system, an intermediate step with testing on **Semi-synthetic** audio is implemented. To create **Semi-synthetic** data, the copies are firstly extracted from the recordings, then those copies are inserted back to the original recordings. Those inserting copies are considered as ground truth of repetitions in the **Semi-synthetic** data, but we do not have prior knowledge of where repetitions are located in the written music.

6.2. Extracting Feature Vectors

We are going to extract feature vectors from 'Normal' recordings which strictly follow the music score without containing any errors and experimentations. Only one feature vector is extracted for each recording in generating **Fully-synthetic** data. The reason is that, if more than one feature vectors are extracted from one recording, those extracted feature vectors might belong to the group of repetitions in the original recordings. This causes an issue that the repetitions in the original recordings are missing in the automatically generated ground truth of repetitions. In opposite, for the **Semi-synthetic** data, we extract several feature vectors from the same 'Normal' recordings without considering any ground truth issue.

Data	**Fully Synthetic**	**Semi Synthetic**	**Real Rehearsal**
Ground Truth	Yes	Yes but Partially	No
Modification	Yes	Yes	No
Controllable	Yes	Yes but Partially	No
Structure	Highly	Partially	unstructured & messy

Table 6.1: This table shows the correspondence relationship between three different types of data with the ground truth, modifications and controllable characteristic.

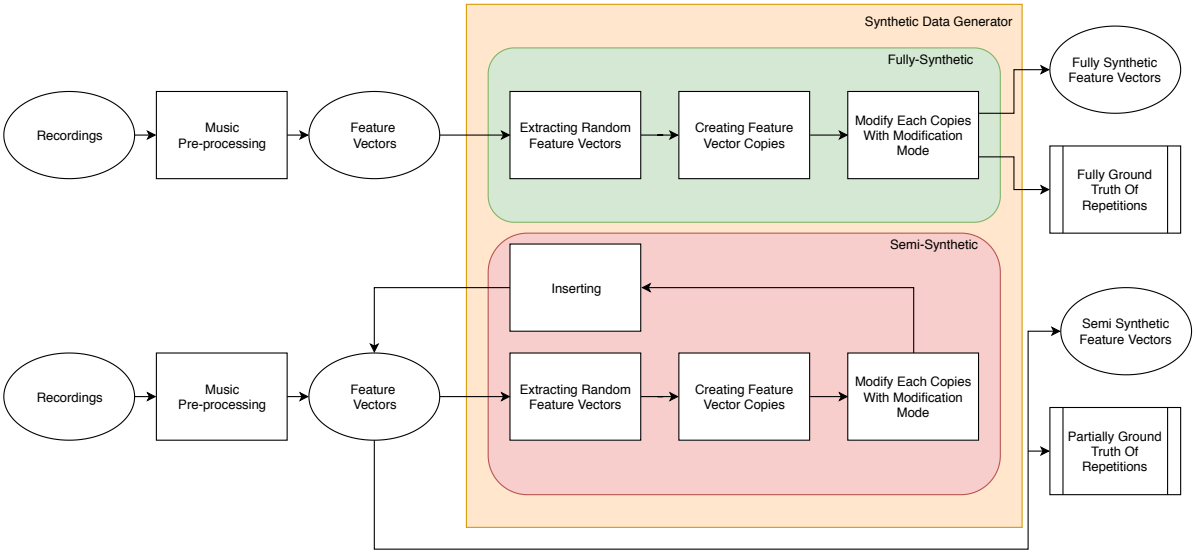

Figure 6.1: Two types of data generated by the synthetic data generator are shown in the orange box. The green box is the workflow of creating fully-synthetic data, and the red box stands for creating the semi-synthetic data. The input of the synthetic data is the commercial recordings.

6.3. Copying & Modification Feature Vectors

Once the feature vectors are extracted from recordings, the next step is creating repetition pairs by copying the extracted feature vectors. Increasing the number of consecutive copies will add the ambiguous ground truth of repetitions. For example, the structure of four copies of feature vectors with semantical label {A} is {AAAA}. The group of repetition pairs can be represented into two different forms, which are {A, A, A, A} or {AA, AA}. However, the ground truth of repetitions are created without considering the patterns {A, AAA} or {AAA, A} that are also considered as repetitions. So that we create two types of **Fully-synthetic** data, one with only 2 copies and the other with 10 copies to investigate the relationship between level of ambiguous ground truth of repetitions and the number of copies.

The goal of the synthetic data generator is to create rehearsal type of repetition pairs which can imitate the real rehearsal recordings. To figure out how to modify the 'Normal' recordings into rehearsal type recordings, we deeply analyzed the reason for noises in the rehearsal recordings. The first type of noise is called 'environmental noise' which exists in rehearsal recordings and have no relation with the musical content in the recordings such as human speaking and white noise. Luckily, the feature extraction can get rid of this type of noise, and it is robustness on environmental variations. Another type of noise is called 'noise in musical practice' caused by different music representations from different musicians, such as the tempo variation or articulation in the music. The 'noise in musical practice' directly influences the characteristic of chroma features; thus, we should focus on simulating those 'noises in musical practice' in synthetic data. From the literature in [24], Winter et al. defined several kinds of noises, and three of them are related to 'noise in musical practice', including **Pause**, **Wrong Pressing Note** and **Tempo Variation**. Our modification modes are created based on these three kinds of noises to create a large amount of synthetic data with the automatically generated ground truth of repetitions. Although modified repetitions in the audio array can maximally restitute the audio data to its original format, it is not able to implement the **Wrong pressing note** and **Tempo Variation** which requires pitch information as input. In contrast, the modifications will be much easier to be done in the chroma feature. **Pause** can be created by adding chroma feature vectors with zero intensity values. **Tempo Variance** can also be easily created by inserting the average feature vectors between two feature vectors to imitate the *tenuto* in the music. **Wrong pressing note** can be created by relocating the intensity value in each of the chroma feature vectors.

We have categorized five modification modes into three modification levels into low, medium and high. The **Clean** mode is in the lowest and the **Mix** mode is the highest modification levels, the **Pause**, **Wrong pressing note** and **Tempo Variation** belong to the medium level of modifications. The following content starts with the **Clean** mode.

Modifications Mode	Clean	Pause	Tempo variance	Wrong Pressing note	Mix
Modifications Level	Low	Medium	Medium	Medium	High

Table 6.2: This table shows the level of different modification modes

6.3.1. Clean mode

Clean mode is the most straightforward mode in the synthetic generator. The repetitions in this mode are created by copying the extracted feature vectors. After that, several copies are concatenated into a long feature vector for further processing.

6.3.2. Pause mode

Although the silence music segments(or called pause repetitions) are removed in the beginning and the end of the recordings, there might still exist some pauses within the recordings that will be detected as repetition pairs. So the function of pause modification mode is to create a pause music feature vector to mimic this occasion. The pause is the feature vector with all the intensity value equal to zero. Like in **Clean** mode, those modified pause feature vectors are concatenated into a long feature vector as synthetic data.

Algorithm 4: Pause generator function

Input:
F_{cens}: *CENS* feature vector
R_b: Range of the length of pause vectors
Output:
F_{cens}: *CENS* feature with added pause feature vector

1 Generate a random location $R_{loc} \in [0, len(F_{cens})]$ in the chroma feature vectors
2 Create pause feature vector F_p with range $R_b \in (0, 1)$
3 Inserting pause feature vector F_p into the R_{loc} in CENS feature vector F_{cens}
4 **return** F_{cens}

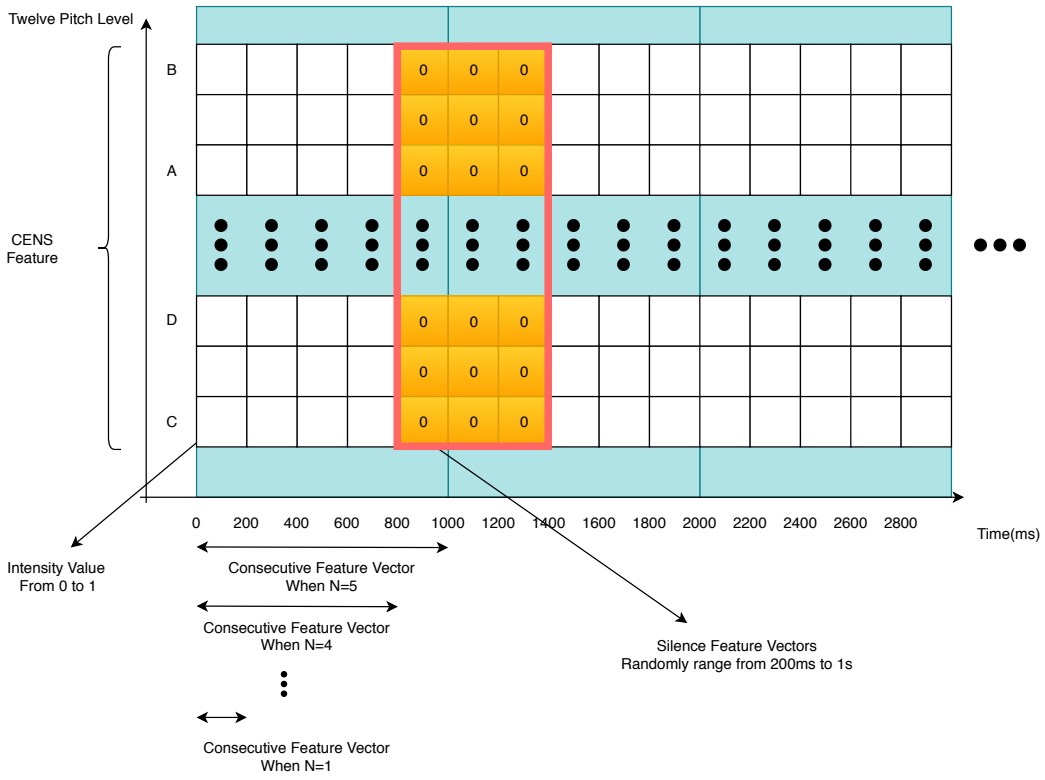

Figure 6.2: Silence Insertions in feature vectors.

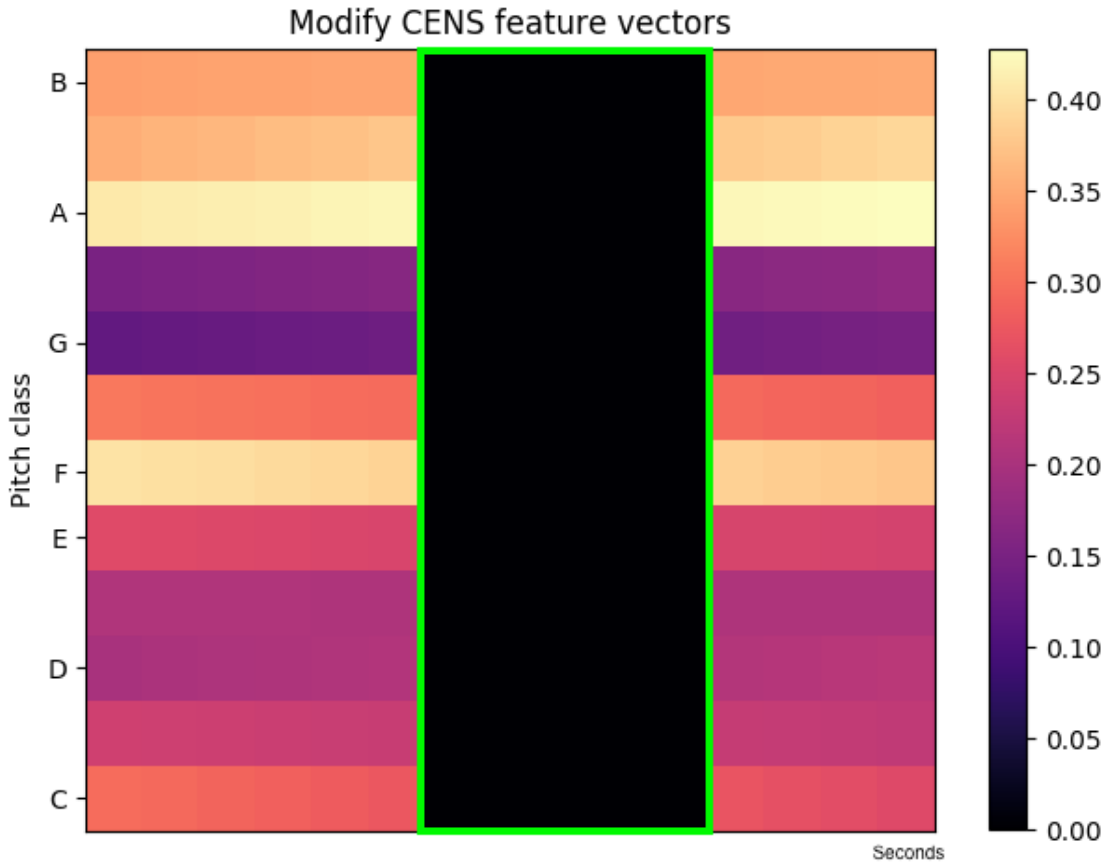

Figure 6.3: Music segments with pause modifications.

6.3.3. Wrong pressing note mode

Due to the large amount of experimentation to refine a music piece, it is inevitable to have wrong note in the rehearsal recordings. To create a wrong note in the feature vectors, we locate the highest intensity value in the single chroma feature as base note and randomly switch it with other intensity value in that single feature vector. By doing this, we successfully generate a different harmony in the feature domain.

Algorithm 5: Create wrong note of feature vectors

Input:
x: Music array
Output:
F_{cens}: *CENS* features with wrong note modifications

1 Choose a random feature vector R_{ram}
2 Shuffling the intensity value in feature vector R_{ram}
3 **return** F_{cens}

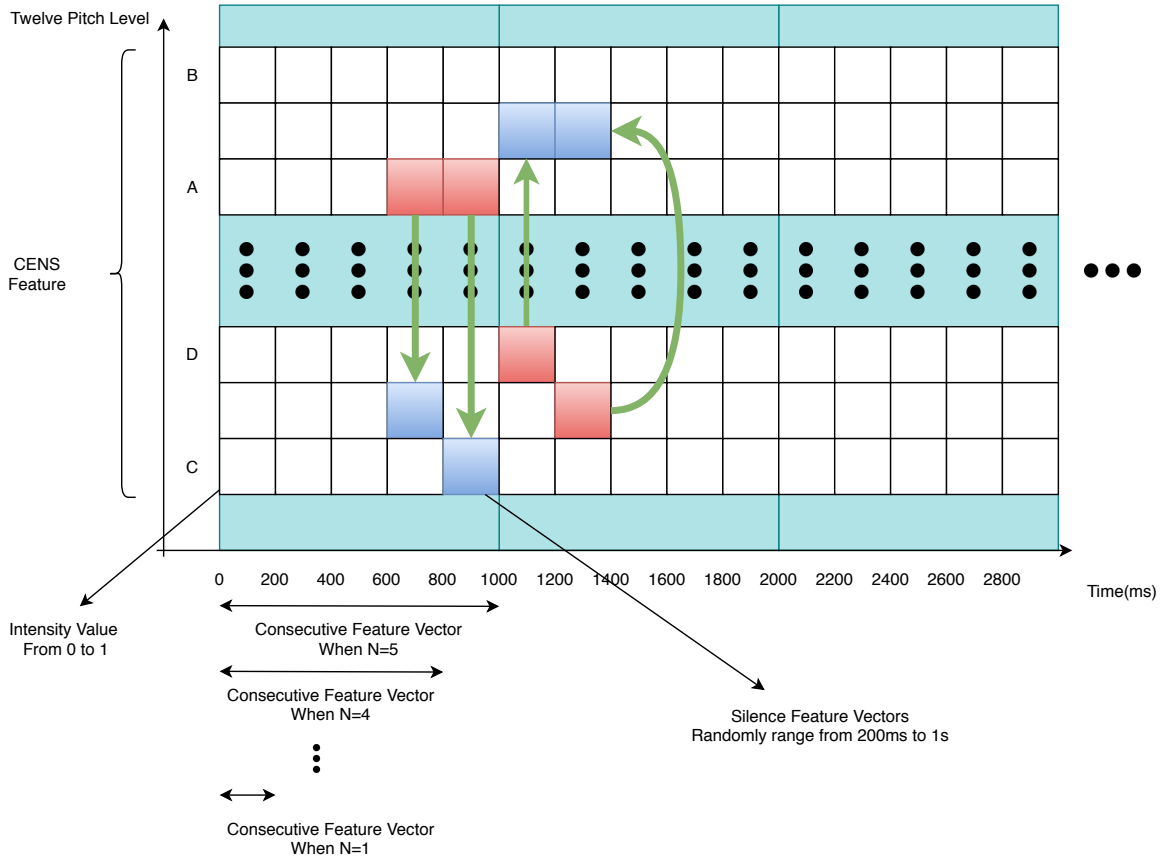

Figure 6.4: Change notes in feature vectors.

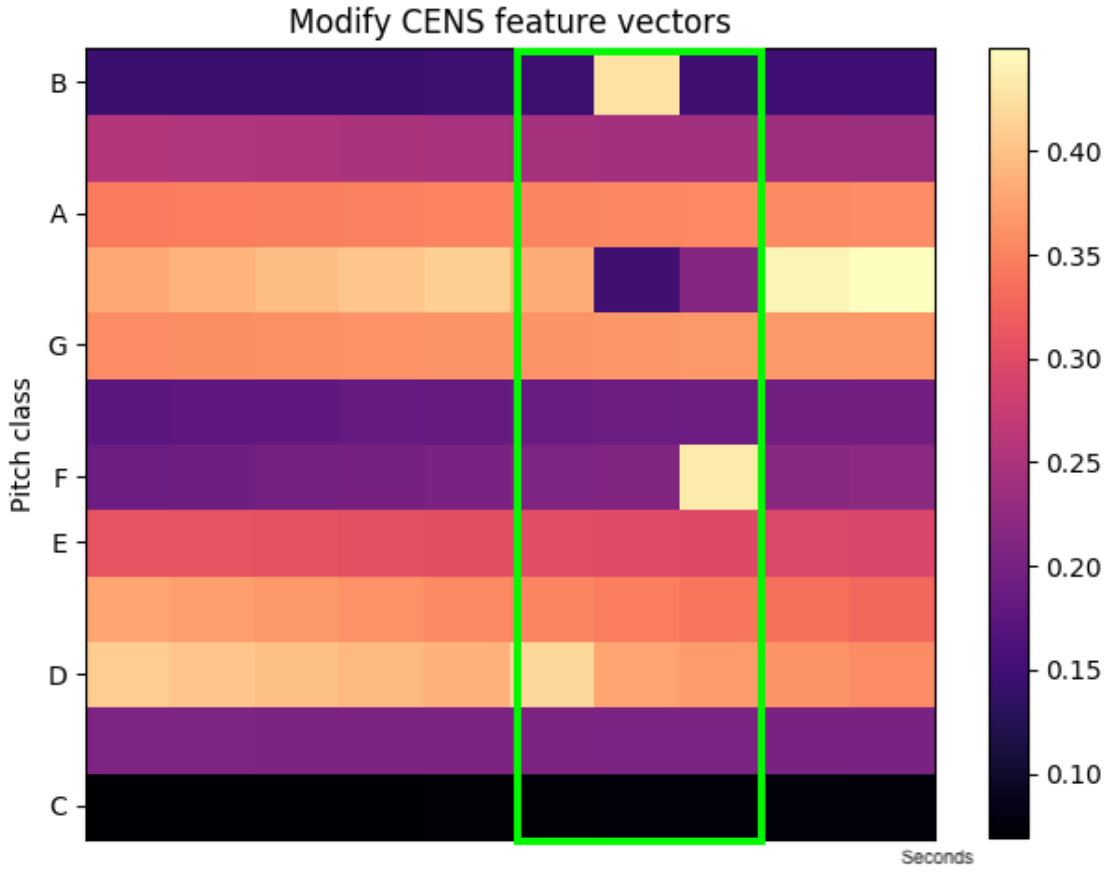

Figure 6.5: Wrong note modification.

6.3.4. Tempo variance mode

The style or representations of music is different from musicians. Those variations are directly shown as the speed or tempo in the music. The idea of creating a slow tempo in the chroma feature is to randomly select a feature vector and insert its copies into the adjacent location to mimic the characteristic of hold keys. In contrast, the way of imitating fast tempo in the chroma feature is to randomly delete feature vectors in the whole feature vectors.

Algorithm 6: Create tempo variance of feature vectors

Input:
x: Music array
N_{op}: Number of operations which correspond to add or delete
Output:
F_{cens}: *CENS* features with tempo variance modifications

```
for i From 0 To N_op do
    Choose a random feature vector F_ram
    Randomize True or False
    if True then
        Copy the feature vectors F_ram to its adjacent location
    else
        Delete the random feature vector F_ram
    end
end
return F_cens
```

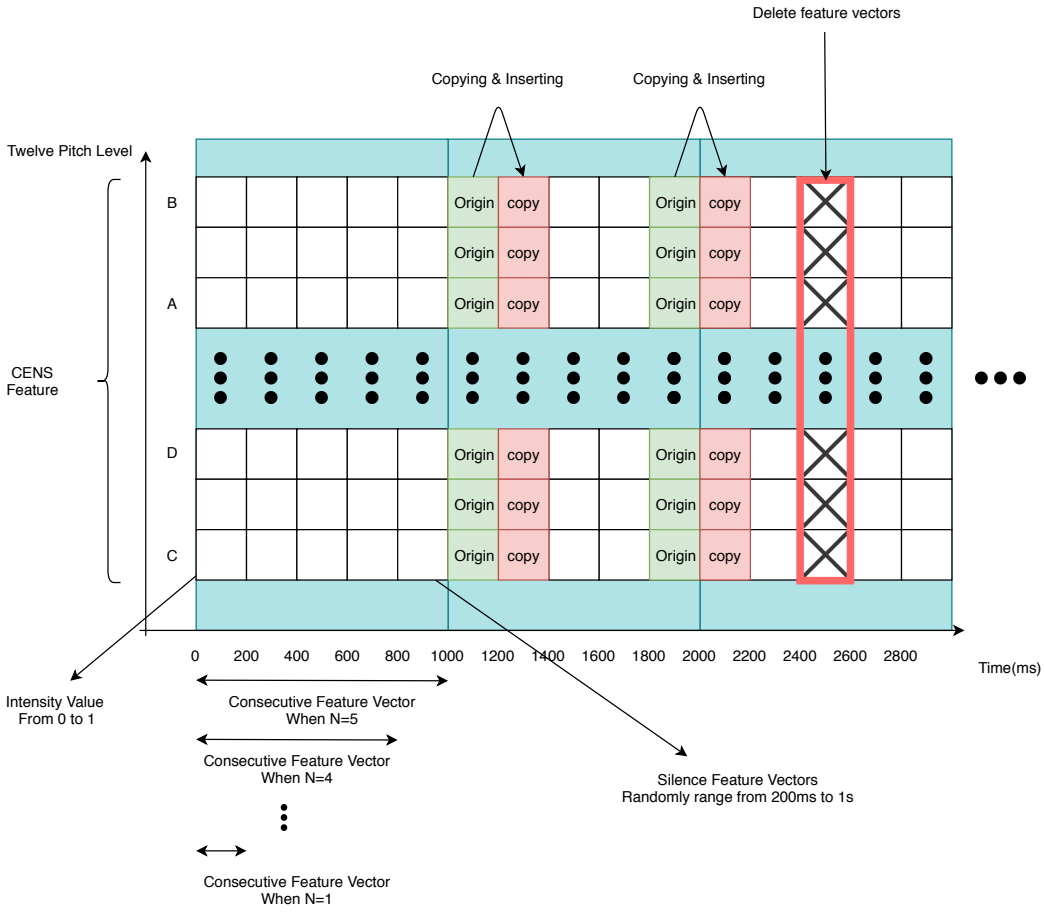

Figure 6.6: Variant the tempo in feature vectors.

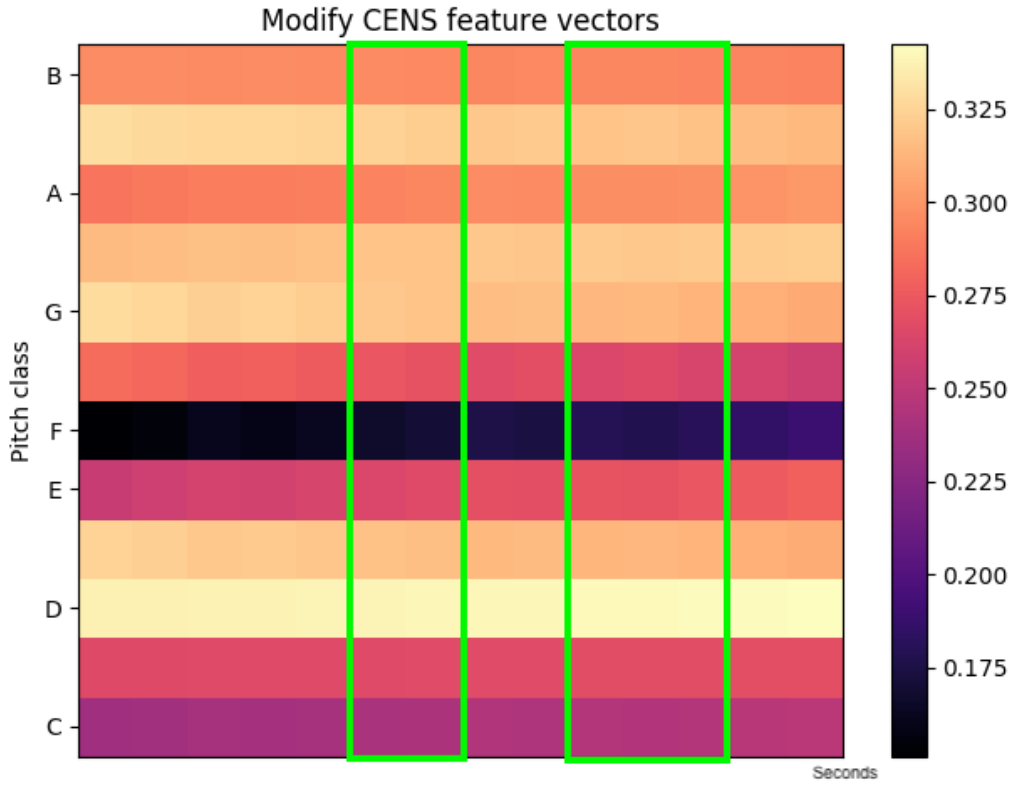

Figure 6.7: Music segments with tempo variance modification.

6.3.5. Mix mode

The last and highest modification level is the mix mode. As mentioned at the beginning of this chapter, this mode can create most diverse modification repetitions pair. The way of creating mix mode is to randomly select modification mode from three different modification mode when creating repetitions.

6.4. Ground Truth Of Repetitions

Two feature vectors are considered as repetition pair if the extracted feature vectors are from the same recording regardless of the modification techniques that are applied to the copies of music segments. The ground truth of repetitions are created by finding all the pairwise combination of the music segments in the same group of repetitions. The Figure 6.8 shows the ground truth of repetitions in **Fully-synthetic** and **Semi-synthetic** ground truth of repetitions. As we could seen from the figure, the **Fully-synthetic** data have fully controllable ground truth while the **Semi-synthetic** data only contain partially ground truth and the ground truth of repetitions in the original recordings (written music) are missing.

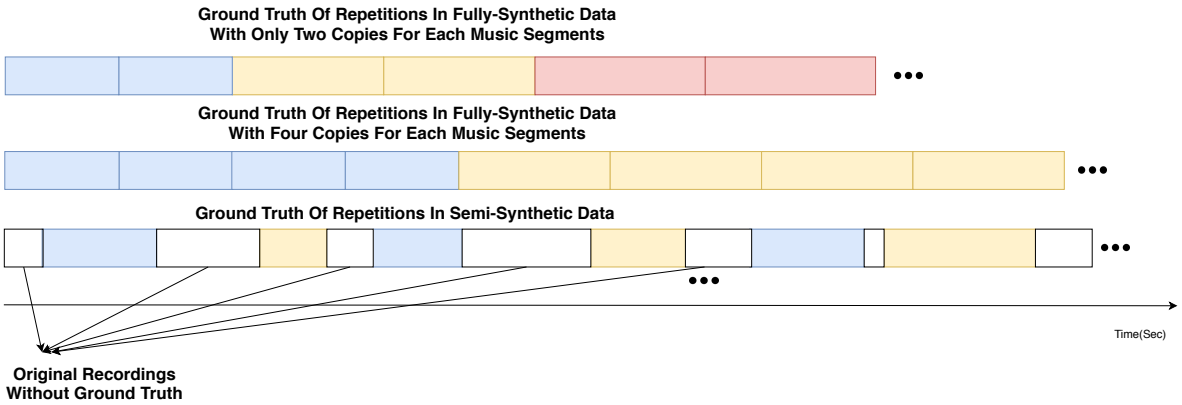

Figure 6.8: Ground truth of repetitions. The blocks with the same color stand for the group of repetitions. The white box stands for the feature vector of the original recordings.

7

Repetition Evaluation

The repetition evaluation is the way of evaluating the quality of repetition pairs extracted from rehearsal analysis framework. We proposed three different evaluation methods to evaluate **Fully-synthetic** and **Semi-synthetic** rehearsal data. Section 7.1 introduces the overlapping calculation algorithm for the purpose of calculating the time overlapping between two music segments. Section 7.2 introduces the evaluation methods used in **Fully-synthetic** and **Semi-Synthetic** data. Section 7.3 introduces the ambiguous ground truth of repetitions in the synthetic data.

7.1. Overlapping calculation algorithm

Calculating the overlapping of repetitions is a vital task to determine whether the predicted repetitions are correctly detected as shown in figure 7.1. The algorithm calculates the percentage of overlapping between two correctly predicted repetition pair. The first part of the algorithm is to check if predicted repetition pair is overlapped with any of the ground truth repetitions. Once the predicted repetition is correctly detected, the percentage of time overlapped between both repetition pairs is calculated. The pseudo-code of overlapping calculation algorithm is shown in 7.

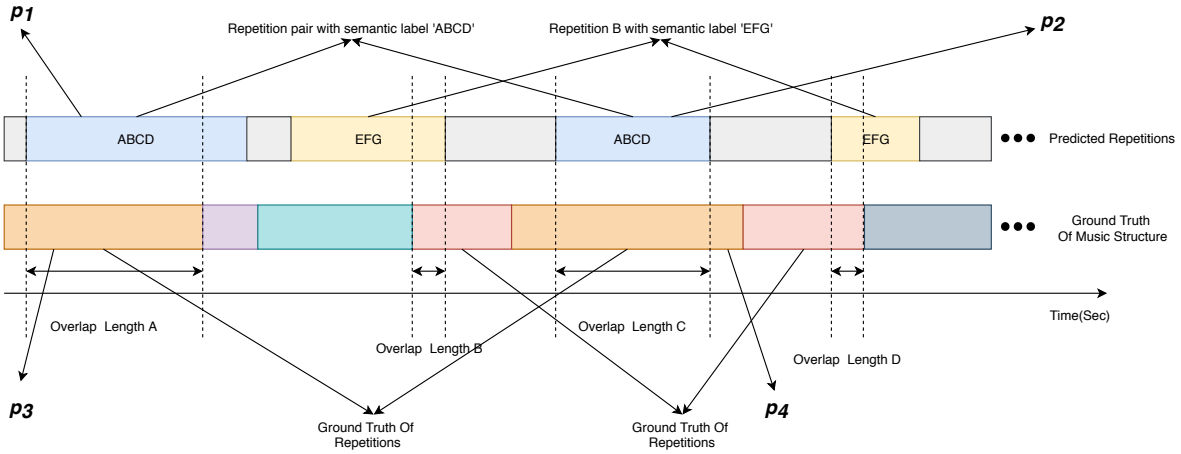

Figure 7.1: The bar on top is the predicted repetitions and the bar on bottom is the ground truth of music structure. There are two repetition pairs predicted with semantic label 'ABCD' and 'EFG' respectively. The block with the same color shows the repetition pairs in the ground truth of music structure. The music segments in one predicted repetition pair can be considered as repetition pair if they have more than 60% overlap with repetitions in the ground truth. For example, the blue block in the predicted repetitions overlaps in time with the orange block in the repetitions in the ground truth.

7.2. Evaluations

The **Fully-synthetic** and **Semi-synthetic** data are evaluated through two evaluation methods. The first evaluation methods are 7.1, 7.2 and 7.3 in traditional information retrieval task used to evaluate the number of

Algorithm 7: Overlap Calculation

Input:
(p_1, p_2): ground truth repetition pair
(p_3, p_4): prediction repetition pair

```
/* If both segments between repetition and ground truth are overlap with each other */
if |p1 ∩ p4| != 0 and |p2 ∩ p4| != 0 then
    return overlap = |p1 ∩ p3| + |p2 ∩ p4|
/* If one of the segments between repetition and ground truth are not overlap with each other, the
   overlap will not be calculated and return false */
else if |p1 ∩ p3| == 0 and |p2 ∩ p4| != 0 then
    return False
/* Same condition as the previous one */
else if |p1 ∩ p3| != 0 and |p2 ∩ p4| == 0 then
    return False
/* Same condition as the previous one */
else if |p1 ∩ p3| == 0 and |p2 ∩ p4| == 0 then
    return False
else
    return False
end
```

correctly predicted repetitions. The true positive (TP) means the number of repetitions that are overlapped with the ground truth of repetitions. The false-negative (FN) represents the number of wrongs detected repetitions that are not overlapped with the ground truth of repetitions or the percentage overlap between predicted and ground truth repetitions are less than 60 %. The false positive (FP) is the number of the ground truth of repetition, which does not overlap with any of the predicted repetitions.

$$precision = \frac{TP}{TP + FN} \tag{7.1}$$

$$recall = \frac{TP}{TP + FP} \tag{7.2}$$

$$F = \frac{2 \times precision \times recall}{precision + recall} \tag{7.3}$$

Second evaluation in 7.4, 7.5 and 7.6 are called purity evaluation methods on measuring the percentage of overlapping between correctly predicted repetitions and ground truth of repetitions. The purity evaluation matrix consists of purity precision (P_p), purity recall (R_p) and purity F scores (F_p). (S_p) means the predicted repetitions and (S_g) is the ground truth of repetitions. $|S_p \cap S_g|$ is related to the total length of overlapping between prediction and ground truth that is measured in seconds. (N_P) in 7.4 corresponds to the number of predictions. (R_p) is purity recall that measures the average between the best predicted repetitions with the total number of predicted repetitions. The (BS_p) in 7.5 is the best predicted repetitions which have the maximum overlap with the ground truth segments.

$$P_p = \frac{\sum \frac{|S_p \cap S_g|}{|S_p|}}{N_P} \tag{7.4}$$

$$R_p = \sum \frac{|BS_p \cap S_g|}{|S_p|} \tag{7.5}$$

$$F_p = \frac{2 \times R_p \times R_p}{R_p + R_p} \tag{7.6}$$

7.3. Ambiguous Of Ground Truth

We have found that the predicted repetitions of **Semi-synthetic** data can be categorized into two groups. The first group of the repetitions belongs to the repetitions that overlaps with ground truth of repetitions. The second group of repetitions belongs to the repetitions that are not overlapped with the ground truth. We can not judge the correctness of this type of repetitions because we do not have the information of repetitions in the original recordings. For the sake of evaluating this type of repetitions, we designed the third evaluation method to evaluate the correctness of those repetitions which are not overlapped with any ground truth of repetitions. The problem of inaccurate evaluation score still exists regarding the ambiguity of ground truth of repetitions in Figure 7.2 that increases the level of difficulties in the evaluations, and this is also mentioned in Chapter 2.

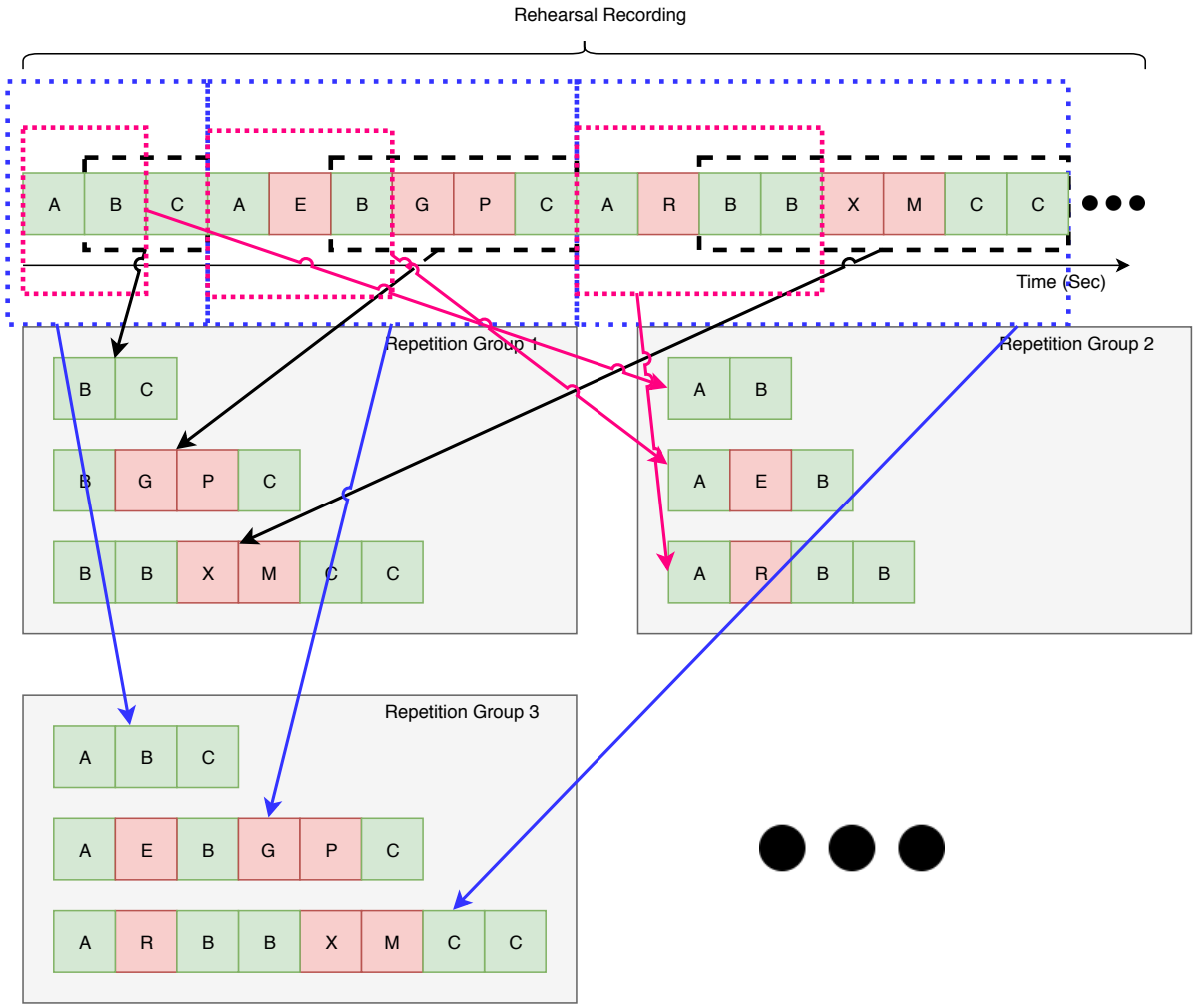

Figure 7.2: The example of different repetition options in rehearsal recordings. As we could seen from the figure, there are more than one repetition pair exist in the recordings and some of them are overlapped. In reality, we can not generate all the repetition pairs from the recordings thus the incomplete ground truth of repetitions influence our evaluation procedure.

8

Experiments On Fully-synthetic & Semi-synthetic Music Data

Using our rehearsal analysis framework, tests have been done on **Fully-synthetic** and **Semi-synthetic** data. In this chapter, those experiments are elaborated upon. Section 8.1 introduces the basic experimental setup. Section 8.2 lists the evaluation results and findings of the two experiments we conduct. In Section 8.3, we further discuss the findings in our experiments.

8.1. Synthesis Setup

To setup the synthetic data generator, we used the Saarland Music Data (SMD) [13] which consists of Western classical music. The recordings in this dataset are free of copyright and consist of performances that strictly follow the music score. To allow for the diversity of our synthetic data, we will vary the static tempo of recordings, modification levels of the variations we allow, and the amount of copies to be made in synthesized rehearsal sessions that are drawn from this data.

Parameter	**Parameter Description**	Value
sr	Sample rate	20480Hz
F_f	Frame Length used in feature extraction	0.2s
l_h	Hop length used in feature extraction	0.2s
L_{fv}	Length of extracted feature vectors from recordings	5sec-10sec
n	Consecutive number of feature vectors	5
L_w	Length of the wrong pressing note	0.2ms - 1sec
L_p	Length of the pause	0.2ms - 1sec
L_{tv}	Length of the tempo variance	0.2ms - 1sec
T_{sm}	Tolerance number in segment merging algorithm	1sec

Table 8.1: Variable used in this experiment.

8.1.1. Static tempo for each recording

In this experiment, we would like to investigate the relationship between music tempo (measure in Beat Per Minutes (BPM)) and the evaluation result. The static tempi in SMD are calculated using funtionality in *Librosa* [9]. However, as listed in Table 8.2, all pieces in SMD have tempo higher than 80 Beat Per Minutes (BPM), meaning all the pieces of music can be considered as fast. Therefore, to ensure we have music with slow tempo, we decided to slow down the slower half of the SMD as shown in Table 8.2.

- *Grave* – slow and solemn (20–40 BPM)
- *Lento* – slowly (40–45 BPM)
- *Largo* – broadly (45–50 BPM)

- *Adagio* – slow and stately (literally, “at ease”) (55–65 BPM)
- *Adagietto* – rather slow (65–69 BPM)
- *Andante* – at a walking pace (73–77 BPM)
- *Moderato* – moderately (86–97 BPM)
- *Allegretto* – moderately fast (98–109 BPM)
- *Allegro* – fast, quickly and bright (109–132 BPM)
- *Vivace* – lively and fast (132–140 BPM)
- *Presto* – extremely fast (168–177 BPM)
- *Prestissimo* – even faster than Presto (178 BPM and over)

8.1.2. Level of modifications

We want our rehearsal analysis framework to be robust to variations as introduced in Chapter 6. The following modifications are chosen:

- Clean: The length of each repetition is chosen between 5 to 10 seconds.
- Pause: The length of the pause is randomly generated to be between 0.2 second to 1 second, with resolution of 0.2 second.
- Tempo variance: The tempo variation is created by randomly adding or deleting between 1 to 5 chroma feature vectors (corresponding to 0.2 to 1 second).
- Wrong note: The wrong note is generated by shuffling the location of the intensity value from 1 up to 5 feature vectors (corresponding to 0.2 to 1 second).
- Mix: We randomly choose between three modification modes (Pause, Tempo Variance and Wrong Note modes).

8.1.3. Synthetic data generator setup

As no ground truth exists in the rehearsal analysis framework, we will generate synthetic rehearsal sessions including ground truth of repetitions. We generate two types of **Fully-synthetic** data and one type of **Semi-synthetic** data based on the parameter shown in Table 8.3.

8.2. Experiment For Fully-synthetic Data

In our first experiment, we test our rehearsal analysis framework by using two types of **Fully-synthetic** data as discussed in the previous section. The experiment is designed to investigate the following three questions:

1. Does the level of ambiguous ground truth increases when more number of copies are created in **Fully-synthetic** data?
2. How do the different tempo of recordings influence the evaluation result?
3. How is the performance of our rehearsal analysis framework in modification methods?

To answer the first question, we create the **Fully-synthetic** data with two levels of ground truth of repetitions by controlling the number of copies. Results of the experiment are displayed in 8.1 and 8.3 and the evaluation scores of **Fully-synthetic** data with two copies are generally higher than **Fully-synthetic** data with ten copies. To conclude, the level of ambiguous of ground truth in 2 copies of **Fully-synthetic** data is higher than 10 copies in the **Fully-synthetic** data.

To answer the second question, we prepare two types of recordings with fast and slow static tempi. The results of evaluation score are displayed in Figure 8.1 and Figure 8.2. There are not much difference of evaluation score when different static tempo are used as base material in the synthetic data generator. However, we guess that the appropriate consecutive number n is related to the tempo of the music (which will be discussed in future work). There are not much difference between the evaluation scores between fast and slow recordings. As a conclusion, the consecutive number n is suitable for both fast and slow recordings and different tempo of recordings do not influence our evaluation result.

SMD	Static tempo	Adjusted tempo
Beethoven_Op031No2-02_002_20090916-SMD.mp3	82	41
Brahms_Op005-01_002_20110315-SMD.mp3	88	44
Bach_BWV888-01_008_20110315-SMD.mp3	96	44
Bartok_SZ080-02_002_20110315-SMD.mp3	96	44
Bach_BWV871-01_002_20090916-SMD.mp3	100	50
Bach_BWV849-01_001_20090916-SMD.mp3	104	51
Ravel_ValsesNoblesEtSentimentales_003_20090916-SMD.mp3	109	50
Rachmaninov_Op039No1_002_20090916-SMD.mp3	109	50
Bach_BWV875-01_002_20090916-SMD.mp3	114	52
Bach_BWV871-02_002_20090916-SMD.mp3	114	52
Bach_BWV875-02_002_20090916-SMD.mp3	114	52
Chopin_Op028-17_005_20100611-SMD.mp3	114	52
Beethoven_Op031No2-03_002_20090916-SMD.mp3	114	52
Beethoven_WoO080_001_20081107-SMD.mp3	114	52
Bartok_SZ080-01_002_20110315-SMD.mp3	120	60
Mozart_KV398_002_20110315-SMD.mp3	120	60
Bartok_SZ080-03_002_20110315-SMD.mp3	120	60
Beethoven_Op027No1-01_003_20090916-SMD.mp3	120	60
Skryabin_Op008No8_003_20090916-SMD.mp3	120	60
Chopin_Op028-03_003_20100611-SMD.mp3	120	60
Beethoven_Op027No1-03_003_20090916-SMD.mp3	126	60
Chopin_Op028-04_003_20100611-SMD.mp	126	60
Bach_BWV849-02_001_20090916-SMD.mp3	126	60
Chopin_Op048No1_007_20100611-SMD.mp3	126	60
Chopin_Op028-11_003_20100611-SMD.mp3	126	60
Liszt_KonzertetuedeNo2LaLeggierezza_003_20090916-SMD.mp3	126	63
Haydn_Hob017No4_003_20090916SMD.mp3	133	N/A
Chopin_Op066_006_20100611-SMD.mp3	133	N/A
Chopin_Op026No1_003_20100611-SMD.mp3	133	N/A
Chopin_Op010-03_007_20100611-SMD.mp3	133	N/A
Liszt_AnnesDePelerinage-LectureDante_002_20090916-SMD.mp3	133	N/A
Bach_BWV888-02_008_20110315-SMD.mp3	133	N/A
Rachmaninoff_Op036-01_007_20110315-SMD.mp3	133	N/A
Brahms_Op010No1_003_20090916-SMD.mp3	141	N/A
Mozart_KV265_006_20110315-SMD.mp3	141	N/A
Rachmaninoff_Op036-02_007_20110315-SMD.mp3	141	N/A
Beethoven_Op027No1-02_003_20090916-SMD.mp3	141	N/A
Haydn_HobXVINo52-02_008_20110315-SMD.mp3	141	N/A
Beethoven_Op031No2-01_002_20090916-SMD.mp3	144	N/A
Chopin_Op028-15_006_20100611-SMD.mp3	150	N/A
Brahms_Op010No2_003_20090916-SMD.mp3	150	N/A
Ravel_JeuxDEau_008_20110315-SMD.mp3	150	N/A
Chopin_Op029_004_20100611-SMD.mp3	150	N/A
Chopin_Op010-04_007_20100611-SMD.mp3	150	N/A
Liszt_VariationenBachmotivWeinenKlagenSorgenZagen_001_20090916-SMD.mp3	150	N/A
Rachmaninoff_Op036-03_007_20110315-SMD.mp3	171	N/A
Haydn_HobXVINo52-01_008_20110315-SMD.mp3	171	N/A
Haydn_HobXVINo52-03_008_20110315-SMD.mp3	171	N/A

Table 8.2: Overview of the SMD recordings used in our experiments, indicating adjusted tempo where it is relevant.

To answer the third question, we analysed the evaluation score in different modification modes and levels. We expected to handle the synthetic variations in synthesize rehearsal data by using segment merging

Data	numRec	numSegPerRec	numCopies	Duration	numTrial
FullySyn (part1)	26 (fast) or 21 (slow)	1	2	<1hrs	10
FullySyn (part2)	26 (fast) or 21 (slow)	1	10	≈ 1hrs	10
SemiSyn	3(random selected)	26	10	≈ 1hrs	10

Table 8.3: This table shows the way of constructing the **Fully-synthetic** and **Semi-synthetic** rehearsal data. The *numRec* means the number of recordings used in the generator, *numSegPerRec* refers to the number of music segments extracted for each recording, the *numCopies* is the number of copies that are created for each extracted music segment and the *Duration* is the duration for two different types of synthetic data. The *numTrial* is the number of trials in this experiment. (PS: The duration does not count the extra time adding in pause and tempo variance modification mode.)

algorithm and F-measure should equals to one, however, due to the ambiguous ground truth and labelling issue in clustering procedure, the F-measure displayed in Figure 8.1 do not reach 1 when testing the **Fully-synthetic** data with two copies. Even worse, the score of F-measure in all modification levels which shows in Figure 8.3 can not reach 0.5 if we increase the number of copies to ten.

In conclusion, the first experiment proved that ambiguous ground truth of repetitions make the rehearsal analysis become an ill-defined problem, and the way of judging whether two music segments belong to repetitions is entirely rely on users' preference. Furthermore, the evaluation method is not sufficient to evaluate the quality of detected repetitions in the condition of ambiguous ground truth of repetitions.

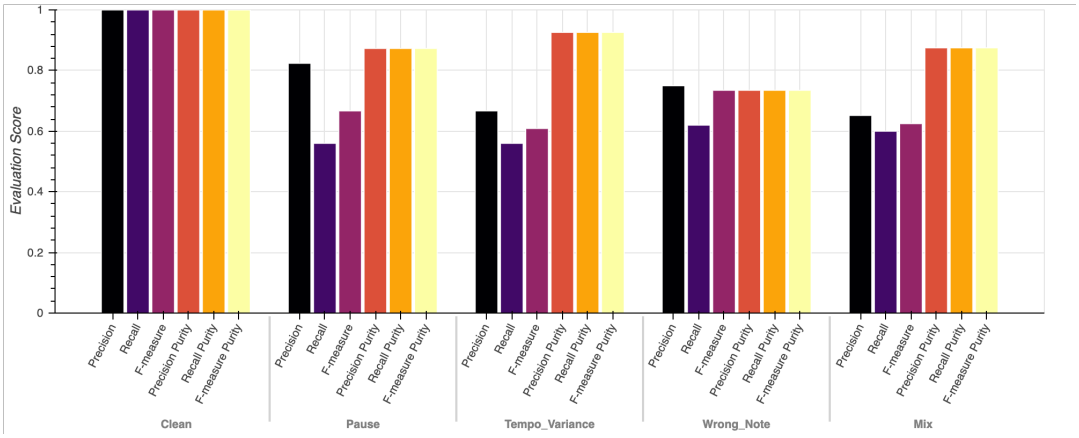

Figure 8.1: Evaluation results on Fully-synthetic data in fast tempo with two copies.

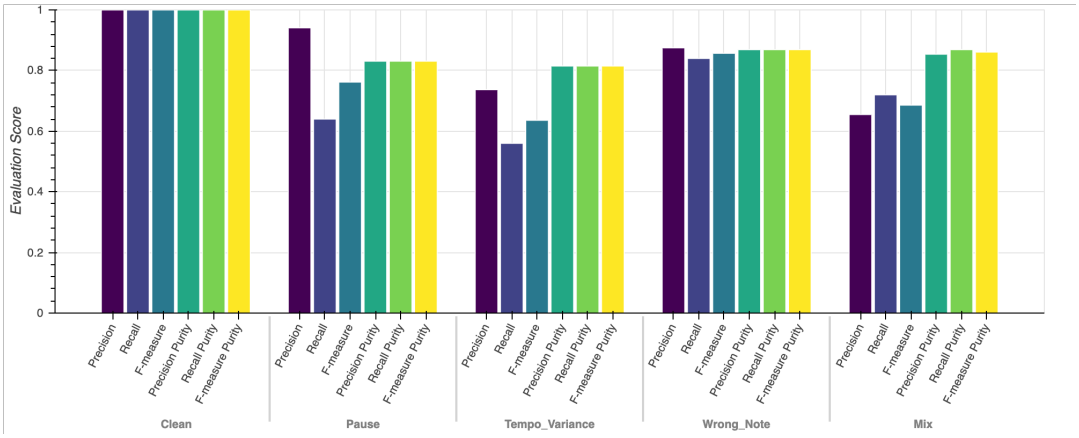

Figure 8.2: Evaluation results on Fully-synthetic data in slow tempo with two copies.

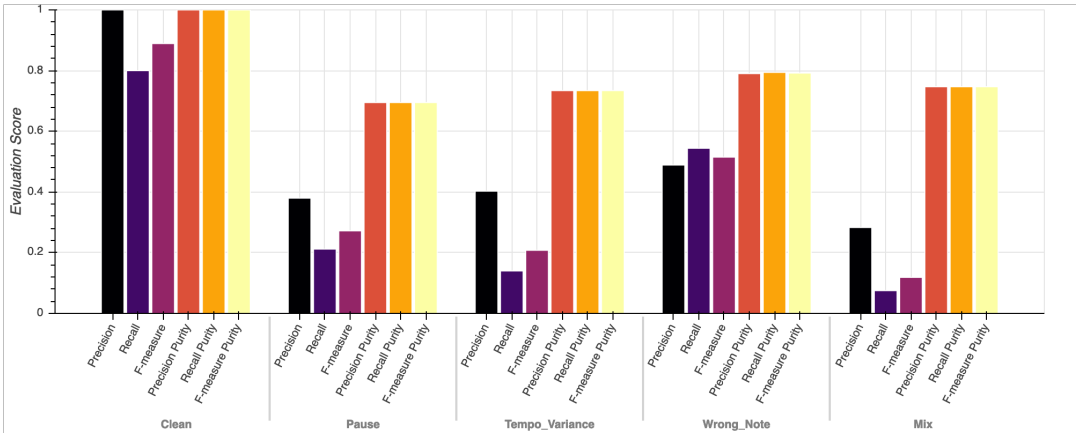

Figure 8.3: Evaluation results on Fully-synthetic data in fast tempo with ten copies.

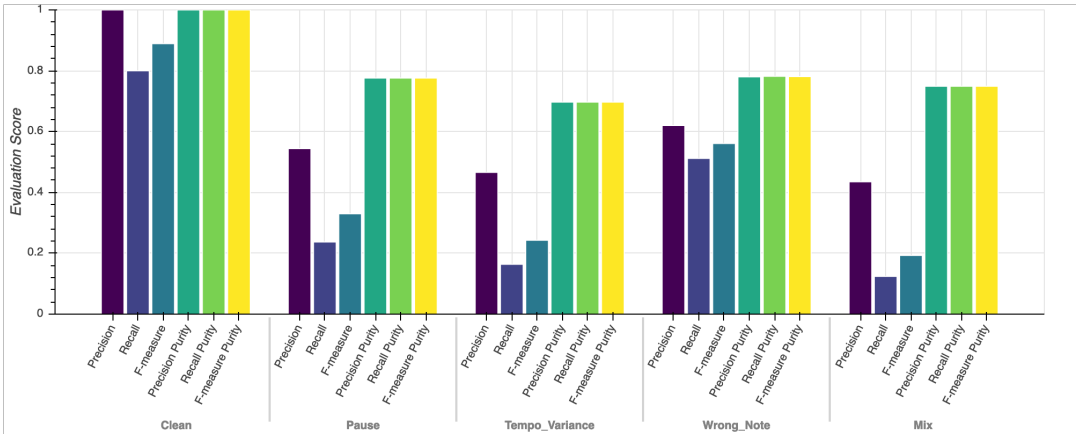

Figure 8.4: Evaluation results on Fully-synthetic data in slow tempo with ten copies.

8.3. Experiment For Semi-synthetic Data

The second experiment is to analysis ground truth conditions between **Fully-synthetic** and the **Semi-synthetic** data. We have seen that the behaviour of evaluation results in first experiment is different from the second experiment. we compare Figure 8.3 and 8.4 that the precision score is higher than the recall in the first experiment, however in the second experiment, the precision score is lower than the recall. One reason is that, the missing ground truth in the written music does not in the ground truth **Semi-synthetic** data. The second reason is that, the ground truth of **Fully-synthetic** data is more than the predicted repetitions extracted from rehearsal analysis system.

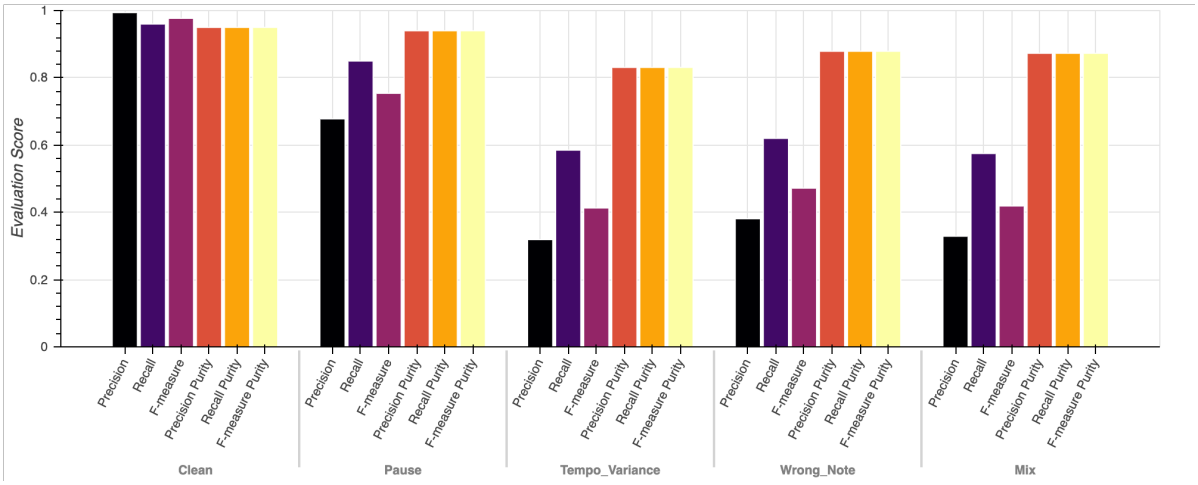

Figure 8.5: Evaluation result of the second experiment in fast tempo.

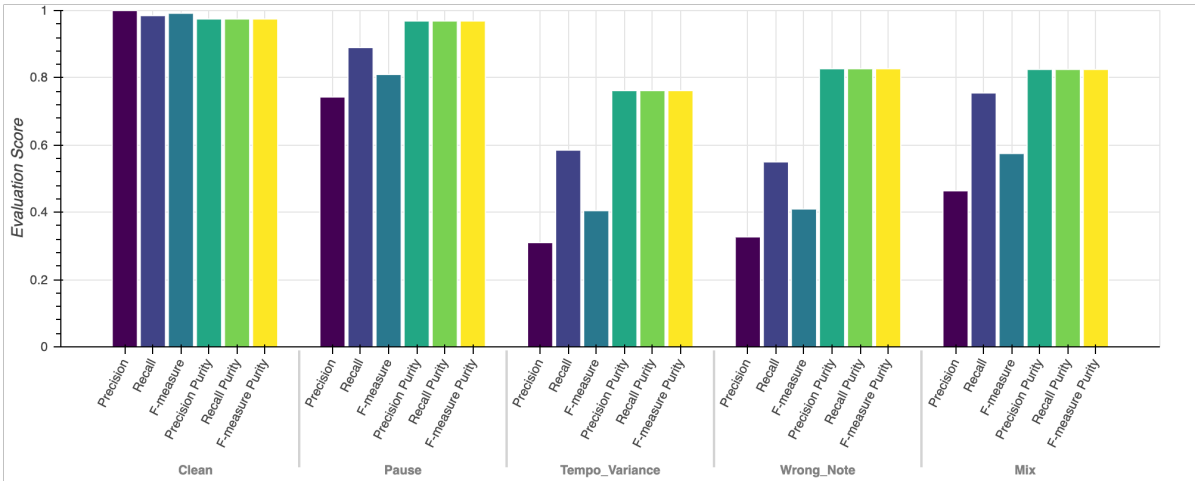

Figure 8.6: Evaluation result of the second experiment in slow tempo.

8.4. Finding Of Experiment Result

The first important finding in the experiment result is the level of ambiguity of ground truth of repetitions varies in different synthetic data. As we have known from 3.3.1, the ground truth of repetitions can be created based on the copies of the music segments. It arises issue that the short repetitions in the ground truth can be merged into longer repetitions that are not in the group of auto-generated ground truth. For instance, music recordings have a structure label "ABCABC" where character A, B, and C represent the semantic label of a single music segment. Based on the structure labels, we found that there exists three kinds of repetitions with either single or combinations of the structure label. The three repetition pairs exist in the structure label are {(A, A), (B, B), (C, C)}, {(AB, AB), (BC, BC)} and {(ABC, ABC)}. According to our merging algorithm, only the longest repetition pairs {(ABC, ABC)} will be detected from the output of our rehearsal analysis framework. Furthermore, due to the ill-defined definition of repetitions, the {(ABC, AB)} or {(AB, ABC)} can also be considered as repetition pairs within the tolerance. These two examples show the high ambiguity of ground truth exists that makes the repetitions evaluation task very difficult than expected.

The second important finding is the different constitution of repetitions in the **Fully-synthetic** and **Semi-synthetic** data. **Fully-synthetic** data contains two types of repetitions. The first type of repetitions are generated by synthetic data generator with automatically generated ground truth that can be used in the evaluation. The second type of repetitions exists in each extracted feature vector from the recordings, which is not in the ground truth of repetitions. The third type of repetitions are the repetitions among the feature vectors from different recordings. There are four types of repetitions in **Semi-synthetic** data. The first three types of repetitions are identical as in **Fully-synthetic** case. The extra type of repetitions are the repetitions originally exist in the recording. Unfortunately, only first type of repetitions is generated through synthetic data generator and can be used in evaluation. We do not have the ground truth of remaining repetitions which aggrandize the inaccuracy score in evaluation procedure. The inner relationship between different repetitions is shown in Figure 8.7.

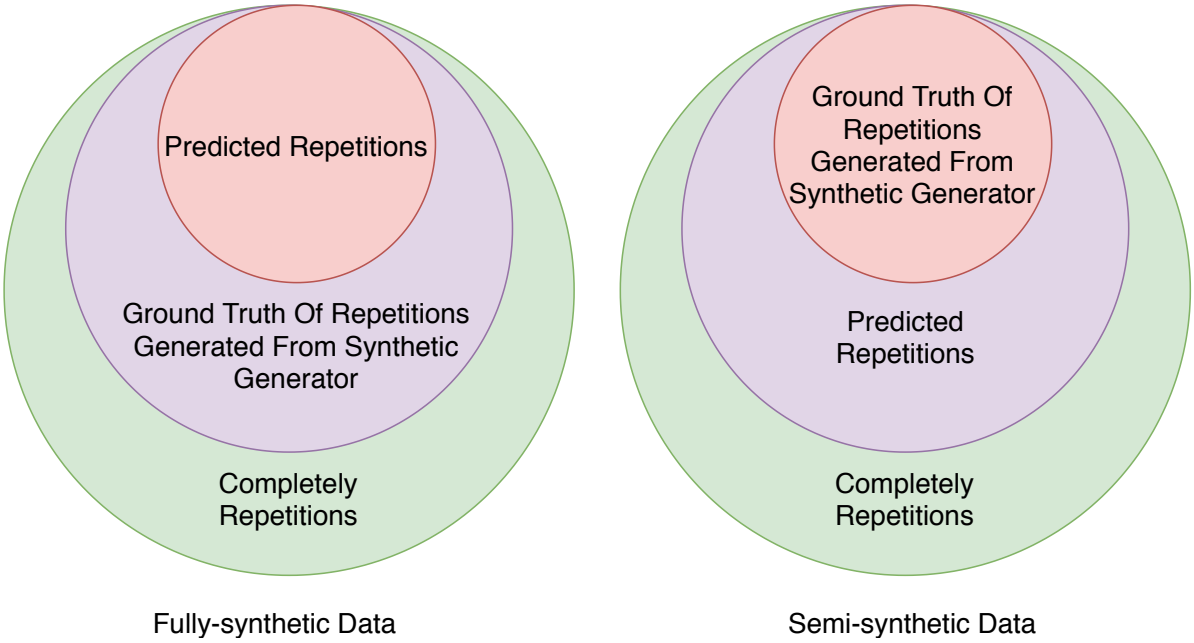

Figure 8.7: The construction of repetitions in fully-synthetic and semi-synthetic data.

9

Testing Real Rehearsal Data

In this chapter, we present the test of real rehearsal data, including creating and evaluating semi-synthetic data using real rehearsal recordings. Students and music educators in the conservatory have different demands on utilizing the rehearsal analysis framework. Students would like to have a system which can list all the repetitions from the rehearsal recordings and match those music segments with their preference commercial recordings. They could quickly compare the most frequently repeated music segments in the rehearsal recordings with the audio matches in the reference recordings. The music educators might want to keep track of the studying progress of their students, by finding the most frequently played music segments in the rehearsal recordings as the most difficult part students have encountered.

In section 9.1, the **Semi-synthetic** data is evaluated using rehearsal recordings as resources, and the evaluation result is compared with the **Semi-synthetic** data using real rehearsal recording as audio resources. Section 9.2 describes the use case in visualizing the rehearsal recordings matched with its reference recordings.

9.1. Semi-synthetic Real Rehearsal Recordings

In this thesis, the rehearsal data recorded in the topic of 'Measuring progress in music rehearsals' by The in his thesis [22] is used. Those recordings are collected from 5 different students major in piano with different expertise from Bachelor or Master level. Those pieces are recorded from their solo repertoire that were the preparations for the final exam at the end of the academic year. The recordings are made with separating practice session. In this experiment, we are using randomly choose real rehearsal recordings that lasts from 4 to 40 minutes long. We created **Semi-synthetic** data by using real rehearsal recordings as material and the parameters in Table 9.1. This is the best we could do to prove that the parameters used in rehearsal analysis framework are transferable in the real rehearsal recordings.

Data	numRec	numSegPerRec	numCopies	Duration	numTrial
SemiSyn	20 (4 - 40 mins)	26	10	≈ 1hrs	10

Table 9.1: This table shows the way of constructing **Semi-synthetic** rehearsal data by using real rehearsal recordings as base material. The *numRec* means the number of recordings used in the generator, *numSegPerRec* refers to the number of music segments extracted for each recording, the *numCopies* is the number of copies that are created for each extracted music segment and the *Duration* is the duration for two different types of synthetic data. The *numTrial* is the number of trials in this experiment. (PS: The duration does not count the extra time adding in pause and tempo variance modification mode.)

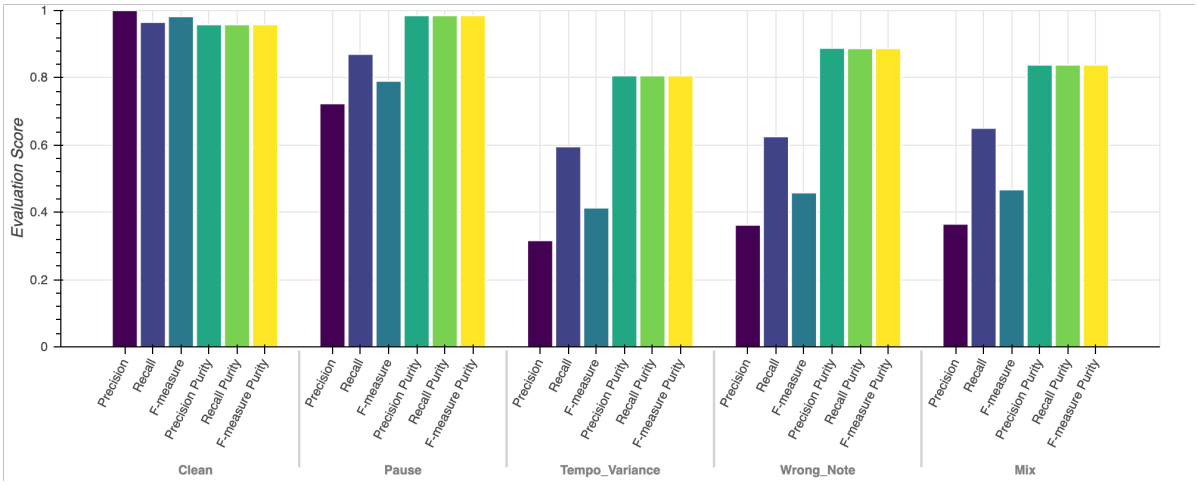

Figure 9.1: Evaluation result of semi-synthetic data generator in real rehearsal recordings.

9.2. Use Case In Visualizing The Rehearsal Recordings

In The's thesis [22], he plots the density distribution of most frequently played music segments in the rehearsal recordings. The music segments with a high value of density are considered as repetitions in the rehearsal recordings. However, we do not know the if the repetition belongs to the repetitions in the written music, or it is because of a musician revisiting the material more often in a rehearsal. In this section, we would like to propose a novelty method to find out the latter type of repetitions in the rehearsal recordings by involving its reference recordings in the experiment. The reference recordings are the commercial version of music recordings that musicians played in their rehearsal recordings.

9.2.1. Visualize repetitions in the rehearsal recordings

One rehearsal recordings named {***Ravel, Sonatine first movement***} have been used to visualize the repetitions. The repetition pairs in the rehearsal recordings are shown as lines in Figure 9.2. The lines with the same color mean the music segments that belong to the same group of repetitions. The horizontal and vertical axis correspond to timeline of the rehearsal recordings. Each line in the Figure 9.2 is the repetition pair in the set of repetition group $RP_n = \{((t_{x1}, t_{x2}), (t_{y1}, t_{y2})), ((t_{x3}, t_{x4}), (t_{y3}, t_{y4}))\ldots\}$ where (t_{x1}, t_{x2}) shows the music segment in x-axis while (t_{y1}, t_{y2}) shows the music segment in y-axis.

We invented **Smart Query** which indicates the key music segments that repeat the most in the rehearsal recordings. The **Smart Query** is the projection of each repetition pair related to the x-axis as (t_{x1}, t_{x2}). At the end, the group of **Smart Query** with x-axis projection is saved as $SQ_n = \{(t_{x1}, t_{x2}), (t_{x3}, t_{x4})\ldots\}$.

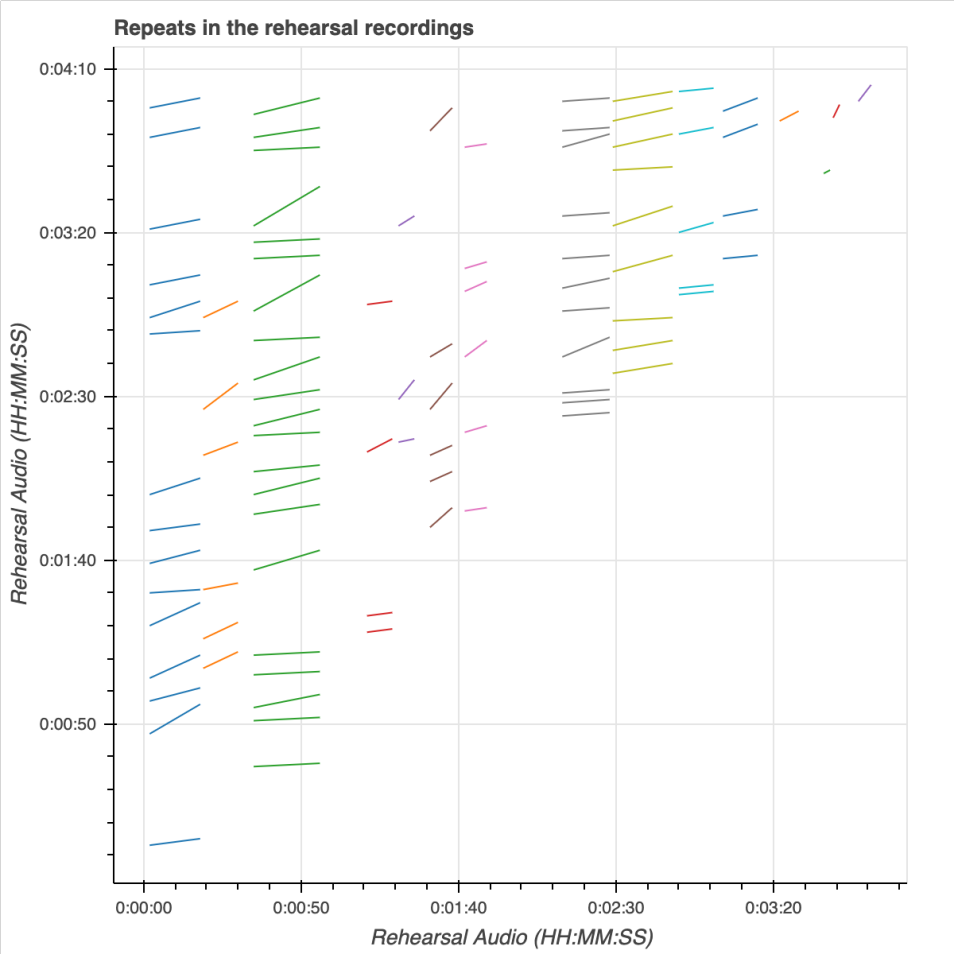

Figure 9.2: Repetitions in the rehearsal audio (also called smart query in this thesis)

9.2.2. Finding repetition groups among rehearsal recordings

Once the **Smart Query** are extracted from repetitions, we can use it to match with reference audio to monitor the similar music segments in the reference recordings. Figure 9.3 shows the matching of smart query with reference audio. Based on the matching between smart query and the reference audio, a density distribution graph is built to analysis the most frequently played music segments in the rehearsal audio. From the Figure 9.3, we could deduct that, when the end of one line is connected to the beginning of the next line, it means there might consist larger music segments pairs.

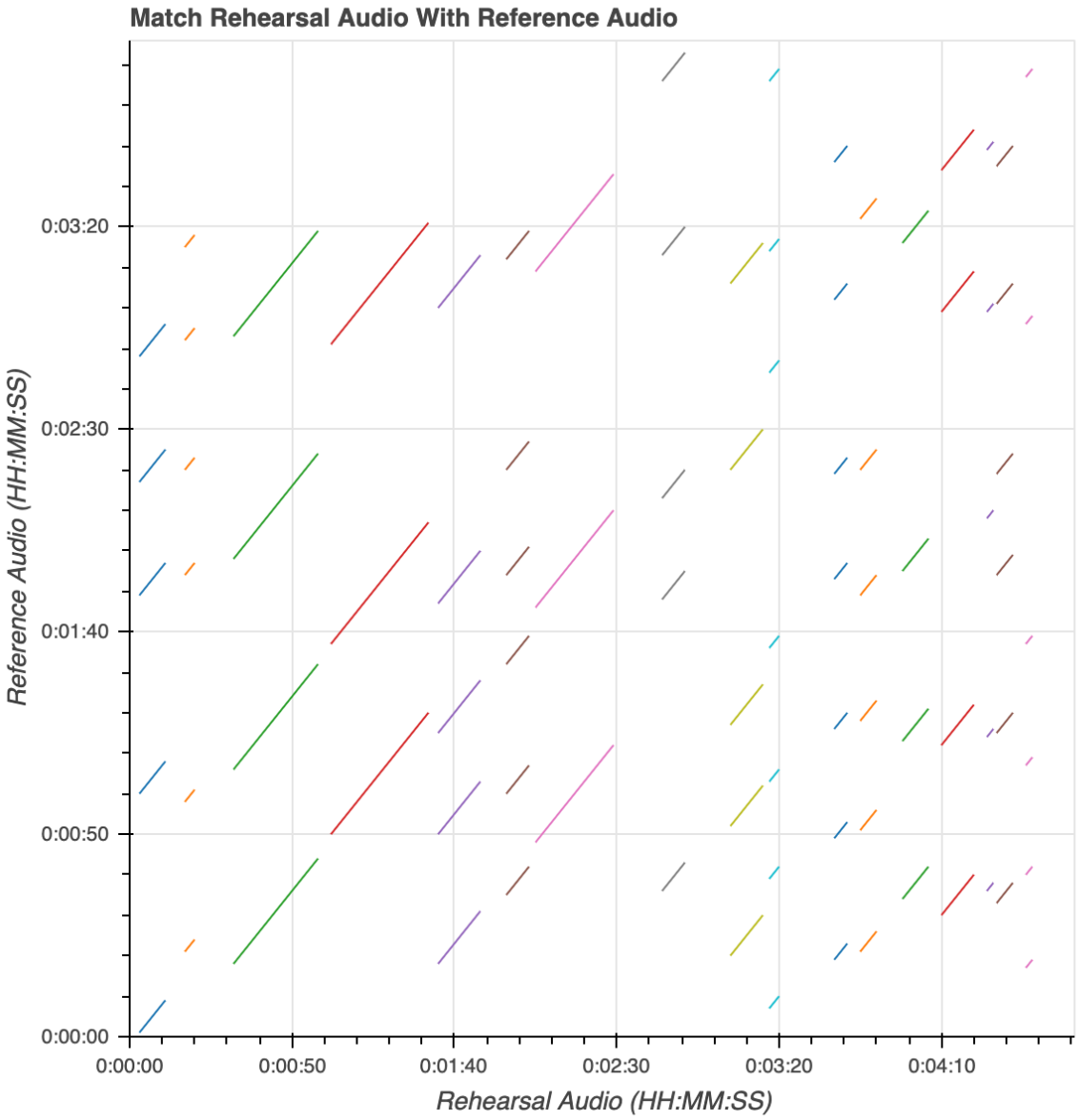

Figure 9.3: Smart queries in rehearsal recordings with matching of its reference recordings. The horizontal axis shows the timeline of rehearsal recordings and the vertical axis shows the reference recordings.

9.2.3. The frequency of repetition distribution in the reference recordings matched with reference recordings

We have found that two types of repetitions exist in the rehearsal recordings. The first type belongs to the reference recordings, and the second type is created by musicians to refine their practice. What is more valuable for us is the latter repetitions, because it positive correlated to the difficulties that students meet in their practice or rehearsal session. In The's thesis [22], the histogram of most frequent played music segments in the rehearsal recordings are created. The distribution does not recognize the repetitions that belong to the repeats in the reference recordings, so we cannot fully distinguish between the types of repeats in the reference music recordings and the repetitions that musicians revisiting the music content more often in a rehearsal.

To tackling this issue, the most intuitive way to get second type of repetitions is to subtract the distribution of repetitions in the rehearsal recordings with the distribution of repetitions in the reference recordings. However, it is unrealistic to get the repetitions in the reference recordings. So we decided to use our rehearsal analysis framework to extract repetitions in the reference recordings with slightly sacrificing the quality of repetitions. With the repetitions from the output of reference recordings, we can remove the first type of repetitions from the rehearsal recordings. The density distribution is shown with green color in Figure 9.4. The second type of repetitions in rehearsal recordings is shown with blue color in Figure 9.4. The distribution in green color without overlapped with the distribution in blue color is the second type of repetitions in the rehearsal recordings. As a consequence, the musicians can figure out which music segments are practiced most frequently in the rehearsal recordings.

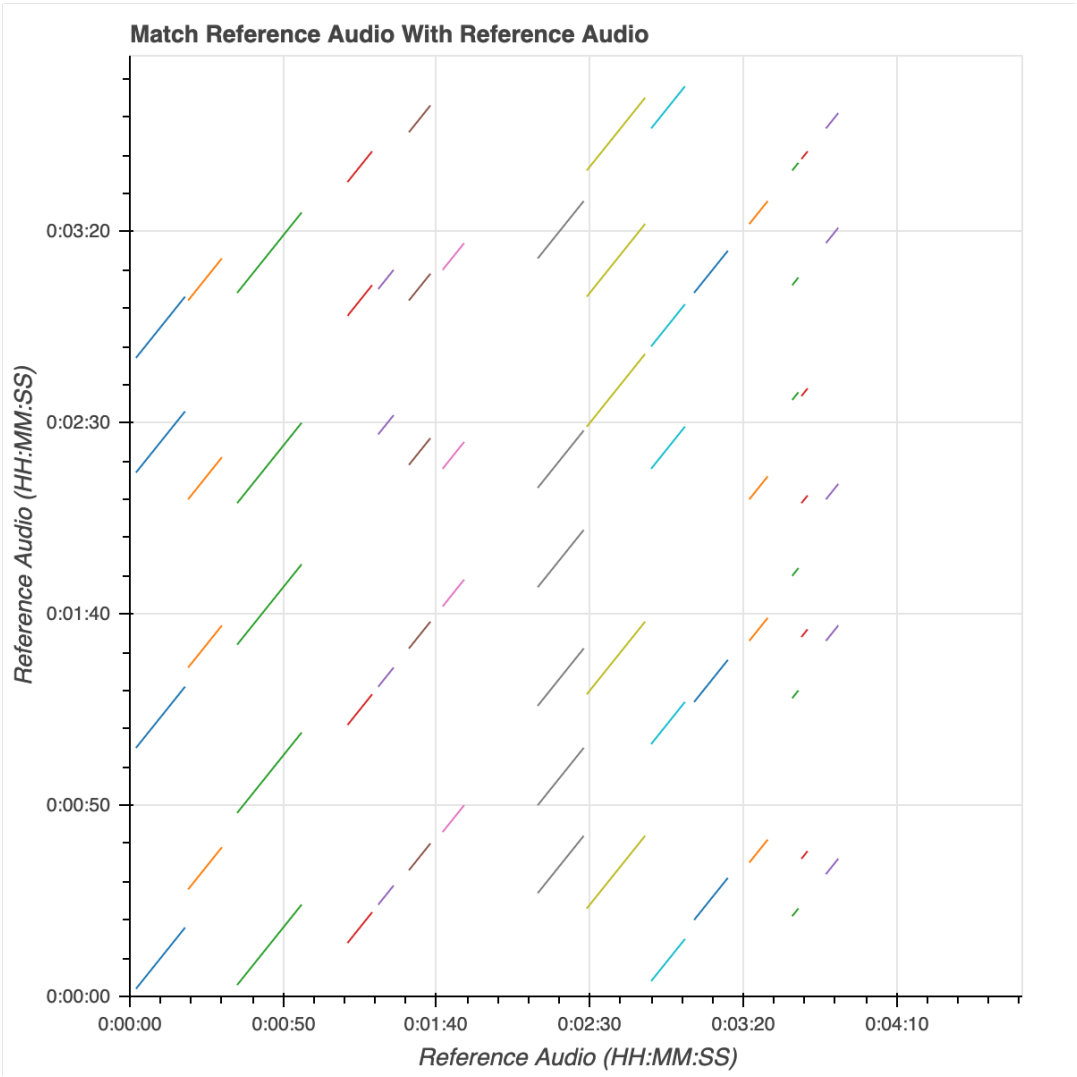

Figure 9.4: Reference recording matches with reference recordings.

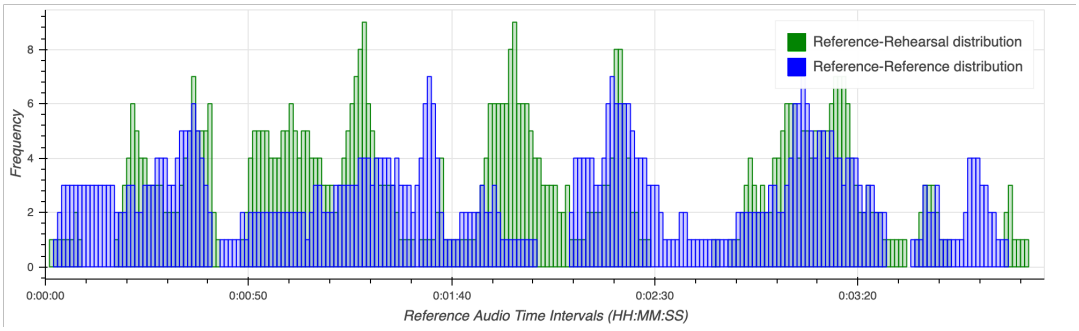

Figure 9.5: Frequency distribution with reference to rehearsal recordings and reference to reference recordings.

10

Conclusion, Discussion and Future works

10.1. Conclusion

The research objective for answering the first research question is to design a rehearsal analysis framework which can automatically extract repetitions in the rehearsal recordings. After realizing the difference of repetitions between commercial recordings and rehearsal recordings, we are the first to define the concept of repetitions in the rehearsal recordings and come up with an unsupervised approach to extract repetitions from recordings.

The research objective for answering the second research question is to prepare testing data with ground truth of repetitions. We design a synthetic data generator to synthesize rehearsal type of data with automatically generated ground truth of repetitions. In synthetic data generator, five modifications modes and three levels of modification have been used to create synthesize data to evaluate the performance of our rehearsal analysis framework.

The research objective for answering the third research question is to to give a comprehensive evaluation for the repetitions from the output of rehearsal analysis framework. To evaluate the framework, we proposed three different evaluation matrices to evaluate **Fully-synthetic** and **Semi-synthetic** rehearsal data. The evaluation methods evaluates the number of correct detected evaluation repetitions and the percentage of overlap between the predicted and ground truth of repetitions. We also found ambiguous ground truth is exist in our synthetic data from the results of our experiments, as a consequence, it can not be easily evaluated accurately by using methods in current literature as well as our evaluation methods.

Except for answering those three research questions, we have tested our rehearsal analysis framework by using real rehearsal recordings and improve use cases in visualizing the density distribution of most played music segments in the rehearsal recordings described in The's thesis [22].

10.2. Discussion & Future Works

Although the rehearsal analysis approach provides a novelty approach by creating synthetic data to test and evaluate the rehearsal recordings, many problems require to be solved in the future.

10.2.1. Silence removal methods

The first extension of this thesis is about the silence removal task in the rehearsal analysis framework. We assume that when the length of silence within the recordings are less than one second, it can be considered as acceptable variations in the repetition pairs, and then it will be merged by the segment merging algorithm. However, if there exists extra noises in the recording that prolong the length of our synthetic variations, the segment merging algorithm might not work in this case. It is because the length of variations is larger than the value of tolerance. To help us create longer and better output of repetitions, future work is needed to have better performance in removing silence from the recordings.

10.2.2. Relationship between consecutive number and different tempo

Theoretically, the smaller number of consecutive number n will have better evaluation result, however, it will lead to a huge computational expense. It is because the consecutive number n might relates to the static

tempo of the music. In the future work, the relationship between the music tempo and the number of consecutive n should be investigated more deeply.

10.2.3. Trainable synthetic data generator

A rehearsal type of synthetic data is generated in our thesis, however, it is hard to say how similar the synthetic data is comparing to the real rehearsal recordings. We expect to investigate more in the future work on this issue, such as designing a synthetic data generator with trainable parameter to create synthetic data that is much likely as the real rehearsal data. In the meanwhile, we can also involve users in the loop of synthetic data generator to create data with user-specific preference.

10.2.4. Evaluation methods for repetitions

Except for the ground truth of repetitions that is generated by synthetic data generator, the rehearsal analysis framework might extract repetitions that exist in the written music and not containing in the ground truth of repetitions. In the future work, it will be useful to design a evaluation method to evaluate the correctness of repetitions outside the repetitions that are generated by synthetic data generator.

10.2.5. Discovering the meaning of variations in repetitions

In this thesis, we extracted and evaluated the repetitions from the rehearsal recordings, but we did not link the variations in the repetitions to the quality of practice. We expect to discover the relationship between variations in the repetitions and the quality of practice.

10.2.6. Ideas From Software Testing

We have noticed that our rehearsal analysis approach is similar to a concept in software testing which called fuzzing. This idea of fuzz testing is first proposed by Miller et al. in his paper [10]. It is a testing method which use its automatic or semi-automated methods to find out the errors or loophole in software, system, or network. Two main types of fuzzers exist in the literature, namely mutation-based fuzzers and generation-based fuzzers mention by Miller et al. in his paper [11] in 2007. The mutation-based fuzzer creates new testing data by modifying the original data such as added or shifted a random bit in the data. This type of fuzzer does not consider any information of input of data format. Another fuzzer is called generation-based fuzzers, and it created the test data from scratch according to the characteristic of the input data. It is very similar to our synthetic data generator which modifies the music copies with different modification modes in feature representations. The difference is that the data edited by generation-based fuzzer are inserted with the hope that a program will raise an error or crash, while we want our rehearsal analysis framework accept a certain degree of tolerance or mistakes. What matches the two approaches between our synthetic data generator and fuzzing is that we automated synthetic data with automatically generated ground truth which is similar to the fuzzing testing, and we evaluate the data through those ground truth. Due to the time limitation, we are not going much deeper into the knowledge of fuzzing but we believe that the ideas of generation-based fuzzers can help us to create a more reliable rehearsal analysis system in the future.

10.2.7. Prospect

Hence, we sincerely hope that more MIR researchers could follow the current progress of rehearsal analysis task, make more contribution to music society, and assist musicians by providing them with more comfortable, simpler and faster way to reflect on their practice.

Bibliography

[1] Mark A Bartsch and Gregory H Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on multimedia*, 7(1):96–104, 2005.

[2] Thierry Bertin-Mahieux and Daniel PW Ellis. Large-scale cover song recognition using hashed chroma landmarks. In *2011 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 117–120. IEEE, 2011.

[3] Wei Chai and Barry Vercoe. Structural analysis of musical signals for indexing and thumbnailing. In *2003 Joint Conference on Digital Libraries, 2003. Proceedings.*, pages 27–34. IEEE, 2003.

[4] Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366, 1980.

[5] Jonathan Foote. Visualizing music and audio using self-similarity. In *ACM Multimedia (1)*, pages 77–80, 1999.

[6] Florian Kaiser, Thomas Sikora, and Geoffroy Peeters. Mirex 2012-music structural segmentation task: Ircamstructure submission. *Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX)*, 2012.

[7] Mark Levy and Mark Sandler. Structural segmentation of musical audio by constrained clustering. *IEEE transactions on audio, speech, and language processing*, 16(2):318–326, 2008.

[8] Cynthia CS Liem and Alan Hanjalic. Cover song retrieval: a comparative study of system component choices. *system*, 3(1), 2006.

[9] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8, 2015.

[10] Barton P Miller, Lars Fredriksen, and Bryan So. An empirical study of the reliability of operating system utilities. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1989.

[11] Charlie Miller, Zachary NJ Peterson, et al. Analysis of mutation and generation-based fuzzing. *Independent Security Evaluators, Tech. Rep*, 2007.

[12] Michael J Moravcsik. *Musical sound: an introduction to the physics of music*. Springer Science & Business Media, 2001.

[13] Meinard Müller, Verena Konz, Wolfgang Bogler, and Vlora Arifi-Müller. Saarland music data (smd). In *Proceedings of the international society for music information retrieval conference (ISMIR): late breaking session*, 2011.

[14] Meinard Müller. *Fundamentals of Music Processing*. 2015. ISBN 978-3-319-21944-8. doi: 10.1007/978-3-319-21945-5. URL `http://link.springer.com/10.1007/978-3-319-21945-5`.

[15] Meinard Müller, Frank Kurth, and Michael Clausen. Audio matching via chroma-based statistical features. In *ISMIR*, volume 2005, page 6th, 2005.

[16] Meinard Müller, Nanzhu Jiang, and Peter Grosche. A robust fitness measure for capturing repetitions in music recordings with applications to audio thumbnailing. *IEEE Transactions on audio, speech, and language processing*, 21(3):531–543, 2012.

[17] Jouni Paulus and Anssi Klapuri. Music structure analysis by finding repeated parts. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, pages 59–68. ACM, 2006.

[18] Jouni Paulus, Meinard Müller, and Anssi Klapuri. State of the art report: Audio-based music structure analysis. In *ISMIR*, pages 625–636. Utrecht, 2010.

[19] Iris Yuping Ren, Hendrik Vincent Koops, Anja Volk, and Wouter Swierstra. In search of the consensus among musical pattern discovery algorithms. In *Proceedings of the 18th International Society for Music Information Retrieval Conference*, pages 671–678. ISMIR press, 2017.

[20] Lior Rokach and Oded Maimon. Clustering methods. In *Data mining and knowledge discovery handbook*, pages 321–352. Springer, 2005.

[21] Jordan BL Smith and Elaine Chew. A meta-analysis of the mirex structure segmentation task. In *Proc. of the 14th International Society for Music Information Retrieval Conference, Curitiba, Brazil*, volume 16, pages 45–47, 2013.

[22] Michael C.G. The. Measuring progress in music rehearsals. 2018.

[23] Li-Chun Wang and Avery. An industrial strength audio search algorithm. In *Ismir*, volume 2003, pages 7–13. Washington, DC, 2003.

[24] R Michael Winters, Siddharth Gururani, and Alexander Lerch. Automatic practice logging: Introduction, dataset & preliminary study. In *ISMIR*, pages 598–604, 2016.

[25] Guangyu Xia, Dawen Liang, Roger B Dannenberg, and Mark J Harvilla. Segmentation, clustering, and display in a personal audio database for musicians. 2011.