Music Visualization Using Projections to 2D Maps

A Thesis Proposal
Presented to
the Faculty of the College of Computer Studies
De La Salle University Manila

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science in Computer Science

by

CRUZ, Edwardo
DIONISIO, Jefferson
FUKUOKA, Kenji
PORTALES, Naomi

Fritz Kevin FLORES
Adviser

July 31, 2018

Abstract

Symphonies are musical compositions for orchestras that consist of several large sections called movements. There are five major musical periods namely the Baroque Period, Classical Period, 19th Century, Romantic Period, and the 20th Century that played a big role during the height of symphonies. This research will compare pairs of symphonies to determine their similarity, and create a visualization method to represent the comparison. From a previous research work by Azcarraga & Flores (2016) that compared symphonies though self-organizing maps (SOM), this research work will compare symphonies through visualization in a 3D SOM. By having a visual representation, the research provides an interactive and straightforward way to identify which parts of the symphonies are most similar and by adding the concept of time series on the clustering of the 3D SOM, more accurate results can be made. Quantitative data will be gathered through frequency cluster analysis and other statistical techniques to measure the musical trajectories of each musical segment of the symphony to produce the overall 3D SOM. T-SNE will also be experimented upon to see if it also produces optimal visualization for symphonies like with SOM.

Keywords: Machine Learning, Music, Time Dimension, Self-Organizing Map, K-Means Clustering.

Contents

Chapter 1

Research Description

1.1 Overview of the Current State of Technology

Music has been a part of peoples culture for hundreds of years, classical music being one of the oldest genres of music. Classical music is rooted in the traditions of early western music and to this day, many people refer to classical music as serious music. Musicians, however, use classical music to refer to music composed during 1750 to 1825, otherwise known as the Classical Era (Bernstein, 1959). The central norms of classical music became established between 1550 and 1900, which is known as the common-practice period. The common-practice period contains the majority of what we now know as classical music. Under this period there are 3 musical eras: Baroque, Classical and Romantic. Music from the Baroque period are decorated and elaborate, with little to no expression. Works from the Classical era contain repetitive dynamics and clean transitions. In contrast to music from the Baroque period, music from the Romantic period are expressive and emotive, having the ability to paint a vivid picture in the minds of the listeners (Grout & Palisca, 1996); however, Dahlhaus (1981) points out that another musical era existed between the Classical and the Romantic period and he refers to this as the 19th century era. This era serves as the transition period for the classical and romantic period, thus having similarities in style with both eras. After the common-practice period comes the 20th century era, which explores modernism, impressionism, neoclassicism and experimental music.

It was in the common-practice era when symphonies began to be composed. Libin (2014) describes symphonies as lengthy forms of musical compositions which are almost always written for orchestras and are consisting of several large movements. They are composed of three to five movements, depending on the time period and are constructed by many different composers (Libin, 2014). There are five major musical periods namely the Baroque Period, Classical Period, 19th Century, Romantic Period, and the 20th Century. Musical pieces from each era share certain characteristics and styles that are representative of the era. With a history of almost 300 years, symphonies today are viewed as the very pinnacle of classical music where Beethoven, Brahms, Mozart and other renowned composers were able to find a venue for transcending their creativities and overall influencing them heavily on their music. During the course of the 18th century, the tradition was to write four-movement symphonies (Hepokoski & Darcy, 2006).

Throughout time, different styles have developed, each having features unique to themselves. Tilden (2013) notes the historical influence of composers with each other and how similar the methods of composing classical music are with pop music. As a result, symphonies written in the early 20th century may be influenced by the great composers and compositions of the previous eras. Analyzing these musical relationships and comparing one to another is a research area that could be done through both manual and machine learning methods.

McFee, Barrington & Lanckriet (2012) compare the use of context-based manual semantic annotation versus their proposed optimized content-based similarity learning framework. With machine learning, the use of high-quality training data without active user participation and the analysis of more data is possible than with feedback or survey data from active user participation. Human error in the analysis process can also be minimized with machine learning since human-supervised training is minimal. Corra & Rodrigues (2016) shows the analysis of music features using machine learning techniques. According to the MIR community (Silla & Freitas, 2009), the two main representation of music feature content are either audio-recorded or symbolic-based. The former employs the explicit recording of audio files while the latter uses symbolic data files such as MIDI or KERN.

SOMphony, a research paper by Azcarraga & Flores (2016), aims to understand the relationship of symphonies between the same composer to denote style as well as to determine the similarities between symphonies of different periods of music to denote influences between time periods. Their research showed the relationships and influences between composers from 5 major musical periods, namely the Baroque Period, Classical Period, 19th Century music, Romantic Period and the 20th century. Their research focused on self-organizing maps (SOM) that are trained using 1-second music segments extracted from the 45 different symphonies. The trained SOM is then further processed by doing a k-means clustering of the node vectors, allowing quantitative comparison of music trajectories between symphonies.

They used frequency counting in their research work for evaluation of each individual symphonies. Each time a 1 second music segment has a BMU inside a cluster, the frequency count for that cluster is incremented. In this way, only the clusters that are frequently visited or symphony will have a high frequency count. The frequency counts are then normalized by dividing the counts of a certain symphony by its total number of music segments. Once these normalized frequency counts are summarized, the resulting percentages can then be used to perform pairwise comparisons between symphonies.

Their research showed that using SOM is indeed helpful in visualizing the musical features of a symphony, making it easier to create insights about the relationships within the different pieces and composers. Their research concluded that a larger dataset would be needed to confirm whether the approach is indeed valid.

SOMphony, however, did not take into consideration the notion of time. In time dimension, each instance represents a different time step and the attributes give values associated with that time (Witten & Frank, 2005). To be able to generate time sensitive musical analysis, this research will add in the time dimension variable to the SOM and a new visualization in 3D space would need to be created.

In a research work by Maaten & Hinton (2008), they introduce a new visualization technique for assigning data points in a two dimensional or three dimensional map called t-Distributed Stochastic Neighbor Embedding or t-SNE, which is a variant of Stochastic Neighbor Embedding or SNE. Since t-SNE produces almost very distinct visualizations based on the experiment in their research work, this technique for visualizing individual symphonies may be more optimal than SOM in terms of both accuracy and efficiency.

1.2 Research Objectives

1.2.1 General Objective

To visualize symphonies using time dimension

1.2.2 Specific Objectives

The research aims to:

1. Perform feature selection to decrease number of features for faster training time in machine learning;
2. Find time-dependent distance measures to compare trajectories;
3. Create a 3D visualization model for the music data;
4. Determine feasibility of t-SNE as another form of visualization for comparison of symphonies;

1.3 Scope and Limitations of the Research

To be able to generate a self-organizing map, the proponents will use jAudio to extract 436 audio features from musical segments generated from the symphony (See Appendix C). The 436 audio features would be trimmed down through feature selection. By selecting only the n most influential features, the time it would take to train the SOM would not be as time consuming compared to using all 436 features, however, this may be at the cost of some of its accuracy in plotting the symphonys musical trajectory. In extracting the features, the musical piece is divided into 1 second music segments in order to be uniform all throughout the piece and to avoid incomplete notes. A 0.5 second overlap is used to be able to consider transitions between each second.

Different metrics for comparison will be used to evaluate which symphonies are similar. Together with the visual maps produced by either SOM or t-SNE, the proponents will then evaluate which metric shows results that reinforce the results seen from looking at the visual map. By doing this, the metrics that show more accurate results can be determined.

To create a 3D model to represent the symphony, the proponents will assign each generated SOM to a point in time and will be used to create a graph representing each map in a series. As a result of using the time dimension, this research will be able to better differentiate symphonies that use similar themes but at different periods of time in the composition.

Similar to SOMphony, the proponents will focus on representation of symphonies using SOMs for the purpose of comparison to other symphonies. T-SNE will also be used as a visualization technique for the musical features of each symphony. The proponents will then decide which among the two to be used for metric evaluations depending on the results.

1.4 Significance of the Research

As the study focuses on qualitative comparison and analysis of different symphonies, the results of the study will prove that music is indeed quantifiable as opposed to being solely qualitative in nature.

The results of this study will also prove that the simplification of hours-long symphonies into a single visual representation in 3D space is possible. It can prove that visualization can be achieved, allowing comparison of data along the time dimension. The results of the study may also help prove the benefits and possibilities of SOM when transitioning from 2D to 3D using time. The results may also verify if t-SNE is a good visualization technique for comparison of musical data.

Some possible future application of the results of this study would include the improvement of existing music information retrieval (MIR) techniques used by music databases. Similarly, this research can also be used to further improve the algorithms used by playlist managers for the retrieval of similar songs from music databases using the comparison of the trained SOMs.

1.5 Research Methodology

This section contains phases and activities that will be performed to accomplish the research. The phases listed here will be arranged sequentially unless otherwise stated.

1.5.1 Concept Formulation and Review of Related Literature

This phase will concern the consolidation of the thesis requirements such as the objective of the research, the research problem to be tackled, and the scopes and limitations of such research. Research related to music comparison, machine learning algorithms in music and music visualization will be part of the Review of Related Literature.

1.5.2 Data Gathering

This phase will concern the gathering of the additional symphonies to be used for the research. The original music dataset for SOMphony is composed of 75 symphonies spread across 5 periods each having 3 composers. To expand the dataset, 2 symphonies will be added to each composer, summing up to a total of 5 symphonies per composer and 2 composers will also be added for each era summing up to a total of 125 symphonies. The proponents have decided to maintain 5 symphonies per composer so that the data set will an equal number of symphonies per composer. The process of selecting which symphonies to be added would be by random to have a better grasp of the general style of the composer. The audio files would be retrieved from online sources and physical means. The researchers would not take into consideration the file type and bitrate of the audio files since music data that is free for use is limited.

1.5.3 Data Preparation

To prepare the data, the symphony audio files are converted into wav files in preparation for splitting. WaveSplitter will be used in splitting the audio file into 1 second segments at intervals of 0.5 second. These segments would undergo feature extraction using jAudio. The result would be an xml file containing all the features determined for each segment. The researchers would then run a RegEx script to remove the unnecessary text and format it into comma-separated values (CSV) file in preparation for labeling. Since the proponents would conduct supervised learning, the data needs to be labelled. The labelling scheme is as follows: period, composer, symphony and audio number (P1C1S1-0001).

The resulting CSV file was further cleaned by removing features that have the feature value 0 for all samples. Features that have at least one NaN value among its data was also removed. These features were removed since they were deemed to not be representative of the data. This leaves 276 features, from the original 534, to be used for training. Once completed, the the entire dataset is then normalized and clustered through Self-Organizing Maps.

1.5.4 Feature Selection

In this phase, the proponents will trim down the 436 features that jAudio has extracted. With decision trees, the top n nodes will be selected as the top n features. Principal component analysis (PCA), on the other hand, can be used to further reduce the number of features by merging columns that have similar data, until the dataset is only represented by a smaller number of features while still retaining the essence of the original data.. Aside from decision trees, the proponents will explore other statistical techniques such as PCA or Pearson correlation for feature selection which may prove more efficient or optimal than decision trees. By doing feature selection, the data set would have a uniform number of features for all symphonies and it would also enhance the efficiency of training the SOM.

1.5.5 Machine Learning and Similarity Metrics

This phase will concern training the SOM and t-SNE using the data with feature-selected features to reduce the dimensionality. The resulting maps can already be used to see similarities between composers, symphonies, or even musical periods. Different similarity metrics will be experimented and the results will be compared to the visual maps to see which metrics produce consistent results and to also be able to validate the similarities between the symphonies.

1.5.6 Visualization

The proponents will assign each BMU to a point on the generated SOM and this will be used to create a graph representing each map in a time series. As a result of using time dimension, the proponents will be able to better differentiate symphonies that use similar themes but at different periods of time in the composition. T-SNE will also be used as a secondary visualization technique for visualizing the symphonies to see if this technique will also provide good visualization for symphonies like with SOM.

1.5.7 Documentation

This phase will be done all throughout the whole research timeframe. The previously mentioned stages and their corresponding findings would also be documented duly.

1.6 Calendar of Activities

Table 1.1 shows the time table for the activities involved with the research for 2017 and Table 1.2 shows the activities for 2018. The numbers represent the number of weeks worth of activity. The # symbol represents the number of weeks allotted for the month.

Calendar of Activities							
Activities for 2017	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Concept Formulation and RRL	###	####	#				
Data Gathering			#	###	##		
Data Preparation				##	##	####	##
Feature Selecion							
Machine Learning and Similarity Metrics				##	##	##	#
Visualization							
Documentation	##	###	##	###	####	####	##

Table 1.1 Timetable of Activities for 2017

Calendar of Activities							
Activities for 2018	Jan	Feb	Mar	Apr	May	Jun	Jul
Concept Formulation and RRL							
Data Gathering							
Data Preparation							
Feature Selecion	###	####					
Machine Learning and Similarity Metrics			####	####	####		
Visualization					####	####	
Documentation	###	####	####	####	####	####	##

Table 1.2 Timetable of Activities for 2018

Chapter 2

Review of Related Literature

This chapter discusses existing research on musical data representations. It also discusses the application of machine learning in music and visualization techniques for musical compositions. A summary of each section in this chapter is presented prior to the discussion of each section.

2.1 Musical Data Representation

Musical Data Representation and Interpretation			
Authors &Year	Title	Research Problem	Approach
Correa & Rodrigues (2016)	A survey on symbolic data-based music genre classification	Expanding music database needs more accurate tools for music information retrieval	Symbolic-based music feature are used to train system for genre classification.
McEnnis, McKay, Fujinaga, & Depalle (2005)	JAudio: A Feature Extraction Library	Solving existing problems in feature extraction systems	They developed jAudio to make extracting features a lot more convenient for researchers.
Cambouropoulos & Widmer (2000)	Automated Motivic Analysis via Melodic Clustering	Finding similarity in music patterns.	Their method uses differences in pitch-intervals and rhythm as basis for splitting one musical motive (small bits of music) from another.

As music grows continuously over time, a constant need for an upgrade to satisfy the number and size of music databases causes the development of more accurate tools for music information retrieval (MIR). MIR is the research field responsible for the development of algorithms or other computational means for the retrieval of useful information from music and the classification of music based on their categories. According to Corra & Rodrigues (2016), the ever increasing research on machine learning, the ever expanding abundance of digital audio formats, the growing quality and availability of online symbolic music data, and availability of tools for extracting musical properties motivate this study on machine learning and MIR. One of the main problems in MIR involves the classification of music based on their genre which this research work tackles. The automatic genre classification of music plays a key role in online music databases where websites or device music engines manage and label music content for retrieval. The main goal of this research work is to be able to compare music samples and give them their own groups or tags in the database so that they can be easily retrieved whenever needed.

Symbolic-based data are music features extracted from symbolic data formats such as MIDI and KERN. In the MIR community, two main representations of music content for MIR research are followed, either the audio-recorded or the symbolic content. Audio-recorded content produce low-level and middle-level features, whereas symbolic content produce high-level features. When analyzing music content, it is preferable to extract more features with the high-level feature of the symbolic content since it is closer to the human perception of music. Due to these reasons, symbolic-based content is used for the research. This research further provides overviews of important approaches regarding music genre classification with the use of symbolic-based music features. The research, as a result, reveals that pitch and rhythm are the best musical aspects to be explored in symbol-based music feature classification that lead to accurate results. Some limitations for further improvement on future works however are present such as the small amount of music dataset used in the research, the bias of using western culture music, and the lack of comparison means for the result of the research due to the lack of previous research works regarding symbolic-based music genre classification.

McEnnis, McKay, Fujinaga, & Depalle (2005) introduced a feature extraction software for audio files called jAudio. jAudio provides an easy to use GUI and a command line interface for selecting which features to select/deselect from the list of features in jAudio's current library of feature extraction algorithms which can be found in Appendix C. The software accepts any audio file as input and outputs ACE XML or ARFF format for the features extracted from the audio file. The proponents in this research encountered many problems with regards to existing feature extraction softwares at the time of their research such as there was great difficulty in extracting perceptual features such as meter or pitch from a signal. Another problem was that there was no existing repository of feature extraction algorithms and researchers would have to implement their own feature extraction algorithm whenever they need it and there will be a big chance that they implement the algorithm incorrectly. There was also no existing feature extraction software that produced a standard output format. Feature extraction code was also restricted and not made available to users, thereby denying researchers from developing more feature extraction algorithms.

JAudio tackles these problems by being a Java-based software, making it easy to acquire and making it compatible with any platform. It produces a standard output format and handles dependencies well by executing all dependencies of a feature extraction algorithm before executing it. For example, the magnitude spectrum of a signal is used by a lot of other features so jAudio would prioritize extracting this first before the others to avoid repeating any extraction process.

JAudio also supports metafeatures which are just features that are used by all other features. Examples of this would be derivatives and mean.

Cambouropoulos & Widmer (2000) stated that music could be categorized into small bits called "motives". These motives are extracted from a musical piece by determining which clusters of musical data can be grouped together while maintaining melodic and rhythmic coherence. This is achieved by representing a melodic segment as a series of notes while minding musical closeness.

Their paper outlines a method that uses differences in pitch-intervals and rhythm as basis for splitting one musical motive from another. For example, two segments can be considered similar if they share a certain number of component notes or intervals using approximate pattern matching. The segments can also be considered similar if they contain shared elements at different pitches. However, this would require a more advanced pattern matching and data structure.

2.2 Machine Learning

Machine Learning			
Authors & Year	Title	Research Problem	Approach
Raphael (2010)	Music Plus One and Machine Learning	Computer driven musical accompaniment	Hidden Markov Models and Gaussian Graphical Models
Dubnov, Assayag, Lartillot, & Bejerano	Using Machine-Learning Methods for Musical Style Modeling	Predicting and determining musical context based on relevant past sample is very difficult because the length of the musical context varies widely	Two approaches, incremental parsing (IP) and the prefix suffix trees (PST), are used in designing predictors that can handle data with very large length.

Comparing trends in musical scores and generating a seemingly new work based on the past works of a certain composer has been the focus of another study. In Dubnov, et. al. (2003)'s research, they stated that predicting and determining musical context based on relevant past samples is very difficult because the length of the musical context varies widely. The proponents formulated then that by using statistical and information theoretic tools, one can capture important trends present in the musical scores for further analysis with machine learning to derive mathematical models for inferring and predicting a seemingly new work from this particular composer. Large contexts make it very difficult to estimate because the number of parameters, computational costs, and data requirements for reliable estimation increases exponentially. To address this problem, the usage of predictors that can handle data with very large length is necessary. Two algorithms are used to design such a predictor for generating new works from old music scores, namely the incremental parsing (IP) and the prefix suffix trees (PST).

The IP algorithm was first suggested by Ziv & Lempel (1978). Given a string as input, the algorithm first builds a dictionary of distinct patterns by traversing from left to right of a sequence once and adding to the dictionary every time a new phrase with a different last character from the longest match that already exists in the dictionary. In representing the dictionary with a tree, every node contains a string in the dictionary and each time the algorithm reaches a node, it means that the string input contains the string assigned to the node but is longer. In this case, a new child node will be added to the tree.

PST was developed by Ron, Singer & Tishby (1996). This algorithm is very similar to IP, but it only adds to its dictionary if and only if the pattern or motif appeared a significant number of times in the string input and will prove to be useful in predicting for the future. Due to this, the main advantage that IP has over PST is that IP is a lossless compression algorithm, since in PST, some patterns are not added to dictionary, especially if they are not significant. PST, however, is more efficient that IP as a parsing algorithm.

Aside from music comparison, machine learning is also applied in automatic music accompaniment. These accompaniment systems serve as musical partners for live musicians that are performing music that is centered on the soloist. Raphael (2010) developed an accompaniment system with three modules namely Listen, Predict, and Play. The first module interprets the audio input of the live soloist in real-time, identifying note onsets with variable detection latency using hidden Markov model-based score following. However, there will be some detection latency due to the fact that a note must be heard first before it could be identified. To resolve this issue, the Predict module, implements a Gaussian graphical model that times the accompaniment on the human musician, continually predicting the evolution as more information comes.

2.3 Music Visualization

Musical Visualization			
Authors & Year	Title	Research Problem	Approach
Azcarraga, A., Caronongan, A., Setiono, R., & Manalili, S. (2016)	Validating the Stable Clustering of Songs in a Structured 3D SOM	Will constructing the classic 2D SOM as a 3D map be feasible, with the learning algorithm still the same as the 2D map?	The 3D map is designed as a $3X3X3$ cube with $9X9X9$ nodes. The cube is divided into one core cube and 8 corner cubes. The Euclidean distance from core to each corner represents the quality of the different categories or genres.
Barrington, Chan, & Lanckriet (2010)	Modelling Music as a Dynamic Texture	Addressing the lack of time-dependency between feature vectors	Dynamic Texture to represent a sequence of audio features
Maaten & Hinton (2008)	Visualizing Data Using t-SNE	To construct a dimensionality reduction visualization technique that can outperform other existing visualization techniques	Modify SNE to produce a more optimal visualization technique by replacing some steps in the algorithm like the cost function of SNE and by replacing the Gaussian distribution with Student t-distribution

Musical Visualization			
Authors & Year	Title	Research Problem	Approach
Foote (1997)	Visualizing music and audio using self-similarity	Is it possible to display the acoustic similarity between any two instants of an audio file as a two-dimensional representation	Audio similarity is computed by parameterizing them into MFCC's and getting the autocorrelation of two MFCC feature vectors V_i and V_j that were derived from audio windows.

Modeling music is representing the audio file in a machine-readable form. (Barrington et al., 2010) raises the issue of the lack of time dependency between feature vectors and stresses the need to have the feature vectors ordered in time. When time is ignored, the feature vectors fail to represent the musical dynamics of an audio fragment. The research addresses these limitations and proposes a visualization model for short temporal fragments of music and calls it a dynamic texture.

In another research work regarding the visualization of symphonies using SOM and also the previous research work this research work desires to expand on, Azcarraga & Flores (2016) focused on whether the music of certain composers and centuries are influenced by prior works of other composers. Their approach relied upon SOMs and k-means clustering where each section on the map represented a specific type of sound. When fed the data from a symphony, a line would be drawn and move from section to section which would represent the different types of sound the SOM would encounter during playback. The result would look like a scribble of lines superimposing each other. By comparing whether this signature of the symphony was similar to one of another symphony, the researchers were able to detect the stylistic influence that one composer has with another.

Azcarraga, Caronongan, Setiono, & Manalili (2016) presents a variant of the classical 2D SOM, a 3D SOM, that is stable with the general clusters not moving around on every training phase. A structured 3D SOM is an extension of a 2D Self-Organizing Map to 3D with a predefined structure. The 3D SOM is represented as a 3x3x3 cube with 27 sub-cubes of the same size. Each sub-cube is further divided into 9x9x9 nodes. The structured 3D SOM is a collection of one distinct core cube in the center and 26 exterior cubes surrounding it, hence summing to a total of 27 sub-cubes. Alongside 3D SOM's built in structure, the learning algorithm used in this 3D SOM includes a four-phase learning and labelling phase. The first phase of training involves the semi-supervised training of the core cube. The second phase involves yet another semi-supervised training, but for the eight corner cubes. The third phase involves training the core cube again, but the training will be unsupervised. The fourth and final phase will be the labelling phase. This phase involves the uploading of the music files into the cube and labelling them accordingly. The music dataset used in this research includes songs from 9 genres: blues, country, hip-hop, disco, jazz, metal, pop, reggae, and rock. Each genre has 100 songs, thus summing to a total of 900 songs.

SOM is usually represented as a 2D map with the input elements being similar to the input environment. This research verifies that designing the SOM as a 3D map is very feasible, with the learning algorithm still the same as with the 2D map. By extending the SOM from 2D map to 3D, the map is further distinguished into the sub-cubes: eight corner cubes and one core cube in the center. Each corner cube represents a music genre while the core cube represents the song itself. The 3D SOM will be able to identify the quality of the different categories or genres of music albums based on a measure of distortion values of music files with respect to their respective music genres. Distortion value is measured by the Euclidean distance between the core cube and a corner cube.

Maaten & Hinton presents a visualization technique using dimensional reduction of high dimensional data into a 2D or 3D map called t-SNE as was briefly discussed in the introduction. This technique aims to transform high dimensional data into low dimensional data representation for data plotting in the map using a series of computational steps or algorithm which will be discussed in more detail in Chapter 3.

In their research work, they compared t-SNE with other techniques for dimensionality reduction for visualization of data which includes Sammon mapping, Isomap, and LLE using the MNIST data set, the Olivetti faces data set, and the COIL-20 data set. The resulting visualizations by t-SNE for all three types of data set proved to be superior to the three other techniques as t-SNE was able to cleanly cluster the different data classes together for each data set as compared to the other three techniques.

The main advantage of using t-SNE to other techniques is that t-SNE models dissimilar data points by means of large pairwise distances and models similar data points by means of small pairwise distances. This would result in a visual image that have similar data points grouped together and are far apart from data points that are very dissimilar with them. T-SNE also uses either of two tricks which they label as early compression and early exaggeration. Early compression is when the map points are forced to stay together at the start of the optimization and early exaggeration is when map points are forced to have large gaps between their respective clusters by multiplying all of the high dimensional probabilities with a certain constant value so that the modelled data points will have larger values. When the data points are closely packed together in early compression, the clusters will be able to move much easier. Similarly, when there are large gaps among the different data points, the clusters will also be able to much easier to find a good global optimization.

Foote (1997) presented a paper on Visualizing Music and Audio using Self-Similarity. In this paper, the acoustic similarity between any two instants of an audio file is calculated and displayed as a two-dimensional representation. Structure and repetition is a general feature of nearly all music, with parts resembling certain parts of the song that came before it. This paper presents a method of visualizing the structure of the music by its acoustic similarity or dissimilarity in specific instances of time through grayscale gradation patterns.

Before getting the similarity measures, the two instants are first parameterized into Mel-frequency cepstral coefficients (MFCCs) plus an energy term. The similarity measure $S(i, j)$ is computed by getting the autocorrelation of two MFCC feature vectors V_i and V_j that were derived from audio windows. A simple metric of vector similarity S is the scalar product of the vectors. A better similarity measure can be obtained by computing the vector correlation over a window w. This captures the time dependence ofthe vectors. To have high similarity measure, the vectors must not only be similar, but their sequence must be similar as well.

Given the similarity measures $S(i, j)$ computed for all window combinations, an image is constructed so that each pixel at location (i, j) is given a grayscale value proportional to the measure. The maximum similarity measure is given maximum brightness. Visually, regions of silence or long sustained notes appear as bright squared on the diagonal. Repeated figures such as choruses and phrases will appear as bright off-diagonal rectangles. If the music has a high degree of repetition, it will show up as diagonal stripes or checkerboards that are offset from the main diagonal. Longer audio files would result to larger images due to the rapid rate of feature vectors. To reduce the image size, the similarity is only calculated for certain time indexes and since S is already calculated at window size w, the paper only looks at time indexes that are an integer multiple of w.

Chapter 3

Theoretical Framework

This chapter contains theories and concepts that are related to the research.

3.1 Symphonies

3.1.1 Basic Structure of a Symphony

The Classical and Romantic symphony is mainly written in four movements, namely the fast tempo or sonata allegro form, the slow tempo, the medium/fast tempo or minuet, and the fast tempo again. The sonata form makes up the main form of Classical and Romantic symphonies. It is composed of two contrasting themes, the aggressive and the passive and is further divided into several sections, namely the introduction, exposition, development, recapitulation, and coda. The introduction section is purely optional and is slow and solemn in nature. The exposition section is where the themes of the symphony are exposed or presented for the first time and will consequently be repeated all throughout. The development section is where the themes are altered and manipulated. The recapitulation section is where the themes return to their original forms from before they were altered. The code section finally represents the end of the movement and this is where the original tone from the exposition section is repeated or recapped to form the ending for the movement (Heikkinen, 2017 & BBC, 2014).

3.1.2 Music Features

A feature is a characteristic used to distinguish one entity from another and in a sense defines its uniqueness. Music features, therefore, are what makes music similar to or different from one another. By comparing the values for each music feature and by examining if a feature is present at all or not, comparison of music by mathematical means is very possible (Huron, 2001).

Today, music information retrieval (MIR) has become an important area of research especially because of the ever expanding database for music through the years. The features extracted from music can be used in many areas of MIR research. It can be said that when two songs share closer values for each music feature, then they are more similar than with others (Corra & Rodrigues, 2016).

MFCC

MFCC, also known as Mel-Frequency Cepstral Coefficients, is the most commonly used feature in speech analysis and since speech analysis and music research are closely interrelated as pointed out by Loughran, Walker, O'Neill, & O'Farrell (2008), then MFCC will likely be the most commonly used feature in music feature extraction.

According to Lutter (2014), MFCC is based mainly from experiments on human misconceptions of words such as when a person misunderstands what another person says. This feature extraction method was first developed by Bridle and Brown in 1974 and was further developed by Mermelstein (1976). The MFCC feature extraction method involves mimicking some parts of the human speech production and speech perception. This feature extraction involves five steps, namely the fourier transform, the mel-frequency spectrum, the logarithm, cepstral coefficients, and the derivatives. The first step, fourier transform makes use of the formula $C_{r,k} = |\frac{1}{N}\sum_{j=0}^{N-1} fjexp[-i2\pi\frac{jk}{N}]|$, where $k = 0, 1, ..., (\frac{N}{2}) - 1$ and N is the number of samples within a speech or time frame.

The mel-frequency spectrum closely mimics the sensation of the human ear's auditory system and the process involves filtering the spectrum with different band-pass filters, devices that pass frequencies within a certain range and reject all others, and the power for each band-pass filter is computed accordingly (Agarwal, 2017). The computation makes use of the formula $C_{T,j} = \sum_{k=0}^{\frac{N}{2}-1} d_{j,k}\, C_{T,k}$, where $j = 0, 1, ..., N_d$ and d is the amplitude of the band-pass filters at index j and frequency k, to produce the corresponding filter bank for the spectrum.

The third step, logarithm involves mimicking the perception of loudness by the human ear and is represented by the formula $C_{T,j} = log(C_{T,j})$ where $j = 0, 1, ..., N_d$.

In cepstral coefficients, the main goal here is to remove the speaker or the music dependent characteristics. The computation of cepstral coefficients results in the inverse of the fourier transform of the estimated spectrum of the signal and is represented by the formula $C_{T,j} = \sum_{j=1}^{N_d} C_{T,j} \, cos[\frac{k(2j-1)\pi}{2} N_d]$ where $k = 0, 1, ..., N_{m,c} < N_d$ and $N_{m,c}$ is the chosen cepstral coefficient for further processing.

Lastly, the derivative represents the dynamic nature of speech or the music.

3.2 Preprocessing

3.2.1 Data Collection

In gathering data, a careful lookup for patent or copyright issues must strictly be observed. Symphonies are musical pieces that were generally composed a long time ago and as such copyright on the actual symphonies are nonexistent. The only copyright issues to be possibly encountered here would be the source of the recreated symphony. For example, when the symphony is uploaded by a certain person in Youtube then the standard youtube license or the creative commons would apply (Brown, 2017).

3.2.2 Preparation of Dataset

In Azcarraga & Flores (2016)'s research regarding visualization and comparison of symphonies through SOM, the preparation of dataset was done by first cutting the symphony into multiple 1 second music segments with an overlapping interval of 0.5 second to provide a smoother transition of the segments when represented later visually in addition to taking consideration of sections or notes that have been abruptly cut during the splitting process. In this way, after the feature is extracted and trained in the SOM, the multiple music segments will make up different musical trajectories which makes up the map visualization.

3.2.3 Feature Extraction

Feature extraction is the means of extracting relevant and effective data to train machine learning algorithms. Not all features, however, may be useful and others may be irrelevant individually but can be useful when combined with other features. The input data or raw data often need to be converted into a set of useful features through preprocessing transformations such as, standardization, normalization, signal enhancement, nonlinear expansion, et al. The resulting data may also be pruned of excess features in order to achieve improved algorithm speed and or predictive accuracy (Guyon & Elisseeff, 2006).

Feature extraction for music can be done using jAudio, a Java project/program developed by McEnnis, McKay, Fujinaga, & Depalle (2005). JAudio is a feature extraction system that provides a user friendly GUI and a command line interface to suit user needs for selecting their desired features to be extracted for the audio. The system accepts audio files as input and outputs XML or ARFF files. This output file contains the values for each feature of the audio file selected by the user.

3.2.4 Feature Selection

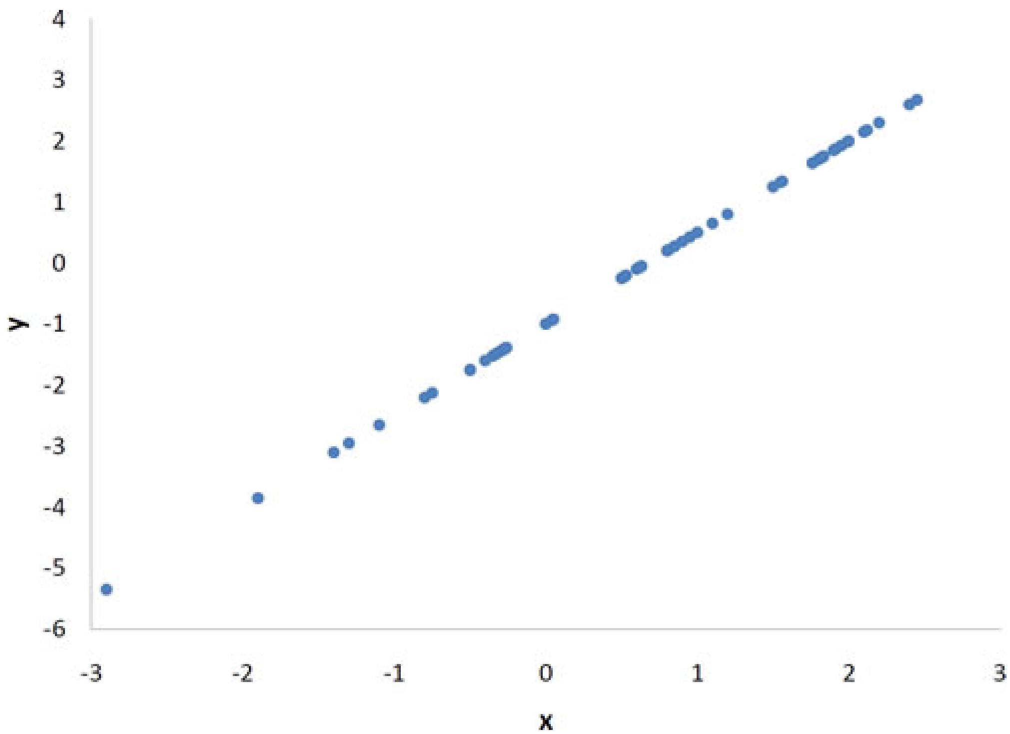
Gupta (2017) defines decision tree as a binary tree that branches down from the root. The term tree-walking is used to continuously make decisions on every level of the tree starting from the root node until the leaf nodes are reached or a satisfactory answer is found. In this way, it can be derived that the nodes found at the top of the tree are more important than its child nodes and all others beneath them. Decision trees are useful for inferring what features of a dataset can greatly or weakly influence its outcome (Mitchell, 1997).

Feature selection is the automatic selection of attributes in the dataset that are most relevant to a specific predictive model. It seeks to identify the out of the ordinary features among all the others and it helps in reducing the number of data attributes being used while still retaining a good or accurate predictive model. Aside from reducing the number of features or data attributes, it can also help in removing unwanted attributes that may decrease the accuracy of the predictive model (Brownlee, J., 2014).

Grabczewski & Jankowski (2005) explains that decision tree algorithms are best used for feature selection because of the inherent characteristic of decision trees that allows them to separate the different features and showcase the more important features since it will appear at the root of the decision tree.

Some simple but successfully tested algorithms for feature selection would include Pearson's correlation coefficient and Fisher-like criterion. Pearson's correlation coefficient or Pearson's R is widely used in the computation of statistics and this involves detecting linear correlation, which is the representation of how close the data points are in making a straight line in a graph, just as shown in Figure 3.1.

Figure 3.1: Pearson Correlation Graph



PCA is another statistical procedure that transforms a set of data points using orthogonal transformation, which scales the set of points by a certain value, into a set of linearly uncorrelated values called principal components. Principal components are ordered such that the variance from the original variable decreases. It will always turn out that the first principal component will have the largest variance that was present in the original variable.

In machine learning, PCA is used to reduce the dimensionality of a set of data points. In implementing PCA on a 2D data set, the mean and the covariance of the data set would first have to be computed first. Mean is computed using $m = \frac{1}{P}\sum_{\mu=1}^{p} x^{\mu}$ and Covariance is measured using $S = \frac{1}{P-1}\sum_{\mu=1}^{P}(x^{\mu} - m)(x^{\mu} - m)^{T}$ (Storkey, 2017).

The eigenvalue and the eigenvector would also have to be computed. Eigenvector is computed using the formula $det(\lambda I - A) = 0$, where λ is the eigenvalue, I is the identity matrix of A, det is the determinant of the matrix, and A is the covariance matrix from the previous step. Eigenvalue is computed using the formula $(\lambda I - A)v = 0$.

For each data point x^n, the lower dimensional representation is $y^\mu = E^T(x^\mu - m)$ and the approximate reconstruction of the original data point x^n is $x^\mu = m + Ey^\mu$. The total squared error over all the training data made is given by $(P-1)\sum_{j=M+1}^{N} \lambda j$ where $\lambda j, j = M+1...N$ are the eigenvalues discarded in the projection.

The last step is to choose the components and to form the feature vector. The number of eigenvectors and eigenvalues is equal to the number of data in the dataset. The eigenvector corresponding to the highest eigenvalue is the principal component of the dataset. To form the feature vector, the top k eigenvectors with top eigenvalues will be used. To compute for the principal component, the transposed version of the feature vector is left-multiplied with the transposed version of the scaled version of the original dataset.

Fisher-like criterion makes use of the formula $\frac{m_0 - m_1}{s_0 - s_1}$, wherein m is the mean value of the feature for the$i$-th element and s is the corresponding standard deviation. This algorithm can only be used, however, when dealing with binary classifications.

Feature selection, in general however, can be classified into three categories, namely the filter methods, wrapper methods, and the embedded methods. Filter method involves labelling each feature with a statistical measure and by comparing these measures, the more important features can be selected. Wrapper method involves grouping different combinations of features together to see which combinations work best. Embedded methods involve learning which features best contribute to the accuracy of the model while the model is simultaneously being created. Some more examples of feature selection algorithms would include best-first search, hill-climbing algorithm, and the usage of heuristics. Best-first search and hill-climb fall under the wrapper methods wherein different combinations are used until the top n features are found. Heuristics falls under the filter method wherein a heuristic score is given to each feature using a statistical measure such as Euclidean distance for example, and the features with the high scores will be the ones selected.

3.3 Machine Learning

Machine learning as defined by Ng (2017) is the science behind computers acting on a certain stimulus without being explicitly programmed to do so. Some examples of impact led by machine learning would be self-driving cars and web search suggestions from Google. Machine learning is also widely used in many different fields of research such as in artificial intelligence, data mining, natural language processing, image recognition, and expert systems (McCria, 2014). In machine learning, the concept of training the system to perform a unique task given a certain amount of data received has two main underlying categories, unsupervised learning and supervised learning.

3.3.1 Supervised Learning

Supervised learning, as defined by Brownlee (2016), is a type of machine learning wherein an input variable and an output variable is defined and an algorithm is used to map the input to the output variable. The goal of this type of learning is to map the input variables to their respective output variables by approximation so that when a new input variable is presented, an output can be predicted by the system. The main difference of supervised learning over unsupervised is that there is no third party that supervises and corrects the training of data in unsupervised but in supervised, intervention of the supervisor is necessary in order to achieve an acceptable level of performance by the system. Supervised learning can be further divided into two groups, namely regression and classification. Regression is used when the output is a real value, for example, weight, height, or age. Classification is used when the output is a category or group, for example, colors, sizes.

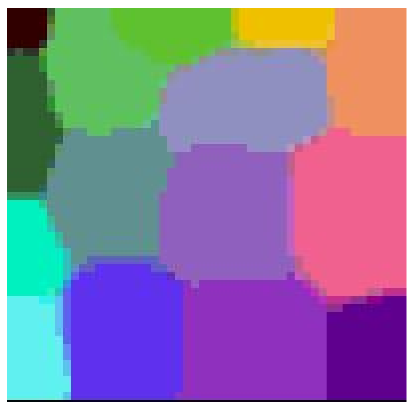
3.3.2 Unsupervised Learning

Brownlee (2016) defines unsupervised learning as having no corresponding output variable. Unsupervised learning is analysing the structure and distribution of the data in order for system to learn. Unsupervised learning can be further classified into two groups of algorithms, namely the clustering and the association. Clustering is used for discovering the groupings of data through clusters and association is used for discovering rules that describe the provided data.

SOM

Teuvo Kohonen (1995) defines SOM as a data visualization technique developed by Professor which reduces the dimension of data through the use of self-organizing neural networks. As SOM reduces the dimension of data, it also groups similar data items together; therefore, it not only reduces the dimension of data but also groups similar ones together. Figure 3.2 shows a basic example of a SOM. Note in this example that the data represented by colors are grouped according to their similarity (eg. yellow is near orange, dark teal is between blue and green).

Figure 3.2: SOM Sample



Rumelhart & Zipser (1985) defines a class under supervised learning called competitive learning. Here, neurons compete among themselves in a winner-takes-it-all scenario wherein only one neuron wins and is activated at any one time. Implementation of this competition is done through the use of lateral inhibition connections, which are structures of a network in which neurons inhibit their neighbors. When neurons are forced to organize themselves through this scenario, then the result would be a map that is self-organized, thus a SOM.

K-Means Clustering Algorithm

K-means clustering algorithm is a type of unsupervised learning algorithm wherein a set of unlabeled data will be grouped together and these groups are defined as the k variable. The algorithm will assign the different data points to their respective k-groups based on the selected features. Data points will then end up being clustered based on their feature similarities. The algorithm has two main iterative steps, , the data assignment step and the centroid update step, that repeats until either data points change clusters, the sum of the distances is minimized, or some maximum number of iterations is reached. Before starting with these two steps, the centroid for each k-cluster is computed first. In data assignment, each data point is placed in their nearest centroid value computed with squared Euclidean distance. In centroid update, the centroid is recomputed by taking the mean of all the data assigned to the cluster of the centroid (Hartigan & Wong, 1979).

t-SNE

Maaten & Hiton (2008) introduces t-SNE as a variant of the Stochastic Neighbor Embedding (SNE) which seeks to visualize high dimensional data by plotting these data points in a two or three dimensional map. Since t-SNE is a variant of SNE, SNE would have to be discussed first before transitioning to t-SNE. SNE starts by transforming the Euclidean distances of the high-dimensional data points into conditional probabilities that represent similarities. For example with $p_{j|i}$, this would represent the similarity from X_j to X_i, that X_i would pick X_j as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at X_i. The conditional probability of $p_{j|i}$ is given with $p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\delta_i^2}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\delta_i^2}$ where δi is the variance of the Gaussian that is centered in X_i. $p_{i|i}$ is set to 0 since only pairwise similarities will be considered. For the low dimensional counterpart of X_i and X_j, the probability is computed by $q_{j|i} = \frac{exp(-||y_i - y_j||^2}{\sum_{k \neq i} exp(-||y_i - y_k||^2)}$.

As with its high dimensional counterpart, $q_{i|i}$ is set to 0. In order for SNE to find a low dimensional data representation that represents the mismatch between $p_{j|i}$ and $q_{j|i}$, a cost function is used, $C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}}$,where KL is the Kullback-Leibler divergences, P_i is the conditional probability distribution over all other data points given data point X_i, and Q_i is the conditional probability distribution over all other data points given data point Yi.

With the variance δi of the Gaussian that is centered over each high-dimensional data point X_i, it is important to note that the density of all data points in a data set are not uniform and it is more appropriate to use a smaller δi value in denser regions than in sparser regions. Any value of δi influences a probability distribution P_i over all other data points. This probability distribution has an entropy value which increases as δi increases. SNE seeks for a value of δi that has a P_i with fixed perplexity specified by the user using binary search. The perplexity is computed by $Perp(P_i) = 2^{H(P_i)}$ where the $H(Pi)$ or the Shannon entropy of P_i is further computed using $H(P_i) = -\sum_j p_{j|i} log_2 p_{j|i}$. The smooth measure of the effective number of neighbors can also be identified as the perplexity and the typical values for this would range from 5 to 50. The cost function shown earlier can also be minimized into $\frac{\delta C}{\delta y_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$.

The gradient can be thought of as a spring force from a map point y_i to any y_j along the map. As the force is computed with $y_i - y_j$, the resulting force can either make the points repel or attract each other depending if the distance between the two is too small or too large. To speed up the optimization and to avoid poor local minima, a gradient update with a momentum term is done using $y^{(t)} = y^{(t-1)} + \eta\frac{\delta C}{\delta y} + \alpha(t)(y^{(t-1)} - y^{(t-2)})$ where y^t indicates the solution at iteration t, η indicates the learning rate, and $\alpha(t)$ represents the momentum at iteration t.

In t-SNE, certain issues such as the crowding problem, which is when too many data points get crowded in a map with a small number of dimension such as with 2D or 3D, are tackled by using a symmetrized version of the SNE cost function with more simple gradients and to use Student t-distribution instead of the Gaussian distribution when computing similarity between two low-dimensional points. The alternative cost function for a symmetrized SNE is $C = KL(P||Q) = \sum_i \sum_j p_{ij} log\frac{p_{ij}}{q_{ij}}$ where $p_{i|i}$ and $q_{i|i}$ are set to 0 just as in SNE and $p_{ij} = p_{ji}$ as $q_{ij} = q_{ji}$ since they constitute the symmetric property. p_{ij} is computed using $p_{ij} = \frac{exp(-||x_i - x_j||^2/2\delta^2)}{\sum_{k\neq 1} exp(-||x_k - x_l||^2/2\delta^2)}$ and q_{ij} using $q_{ij} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k\neq 1} exp(-||y_k - y_l||^2)}$. In order to avoid having p_{ij} with extremely small values which results from outliers, p_{ij} is first set by $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$ so that $\sum_j p_{ij} > \frac{1}{2n}$ for all data points x_i. The gradient of symmetric SNE is computed using $\frac{\delta C}{\delta y_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)$.

In t-SNE, just as discussed above, Student t-distribution with one degree of freedom will take the place of Gaussian distribution as the heavy-tailed distribution in the low-dimensional map. q_{ij} will now be computed using $q_{ij} = \frac{(1+||y_i - y_j||^2)^{-1}}{\sum_{k\neq 1}(1+||y_k - y_l||^2)^{-1}}$. A student t-distribution with one degree of freedom is used because it has the property $(1 + ||y_i - y_j||^2)^{-1}$ wherein it approaches an inverse law for large pairwise distances in the low dimensional map. The resulting map representation of joint probabilities will make the map points invariant to changes in the scale of the map. Large clusters of points that are far apart will also interact just the same way as how an individual point would.

The gradient of the Kullback-Leibler divergence between P and the Student t-based joint probability distribution Q can be computed using $\frac{\delta C}{\delta y_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + ||y_i - y_j||^2)^{-1}$. The resulting gradient from t-SNE will strongly repel dissimilar data points resulting in a visual image that clearly separates each data class as can be clearly seen in the t-SNE results in Appendix E as compared with the results from the other dimensionality reduction techniques used.

In their experiment set-up, they used a total of three data sets as was discussed back in Chapter 2 which includes the MNIST data set, the Olivetti faces data set, and the COIL-20 data set. They first used PCA to reduce the dimensionality of the data to 30 so that the computation of pairwise distances between data points would be faster and at the same time also suppresses the noise without severely distorting the distances between the data points. They then applied different dimensionality reduction techniques which includes Sammon mapping, Isomap, and LLE to compare the resulting visualization later on with the result of t-SNE. The results of each data set for each technique on the MNIST data set can be viewed in Appendix E. The coloring scheme in the visual images represent each class of data. The cost function parameter they used for their experiment was Perp = 40 for t-SNE, none for Sammon mapping, and k = 12 for Sammon mapping and Isomap.

Their results show that t-SNE overall had better visualization because the data classes are more distinct or separated than compared with the results from the other methods. In general, t-SNE outperformed all the other dimensionality reduction techniques used in their research; however, the authors point out that t-SNE has three general weakness to consider. The first one is that it is unclear how t-SNE will perform on the more general task of dimensional reduction. In other words, if dimensionality is increased to more than 3, it is unclear whether t-SNE will still produce optimal visualization results. One way of addressing this problem is to optimize the Student t-distribution to have more than one degree of freedom. The second weakness is the curse of intrinsic dimensionality since t-SNE only reduces the dimensionality of data based on the local property of the data. Data sets that have high intrinsic dimensionality and an underlying manifold that is highly varying causes the local linearity of assumption on the manifold that t-SNE implicitly makes be violated; therefore, applying t-SNE to data sets with high intrinsic dimensionality can produce unreliable results. One way to address this problem is to perform t-SNE on a data representation obtained from a model that represents the highly varying data manifold efficiently in a number of non-linear layers. The last weakness is the non-convexity of the t-SNE cost function. Since the cost function of t-SNE is non-convex, several optimization parameters would first have to be chosen. The constructed solutions would then vary depending on the optimization parameter used each time from an initial random configuration of map data point. They explicitly point out however that the same choice of parameters can be used for different visualization tasks and the result would still be of similar

3.4 Visualization

3.4.1 Single Image

Just as done in Azcarraga & Flores (2016)'s research work and in Maaten & Hiton (2008)'s t-SNE, visualization for the result of the SOM and t-SNE is projected in a single image. With SOM, the BMU or best matching unit, which will be explained further in section 3.5.1, represents the music trajectory of a certain 1 second music segment from the symphony. This sequence of BMUs make up the visual image representation of a certain symphony. A color coding scheme was also used to denote the time sequence of a certain music trajectory in the image, blue representing the start and going to red as the music progresses as shown in Figure 3.3. For t-SNE, sample visual images can be seen in Appendix E.

Figure 3.3: SOM Image from SOMphony

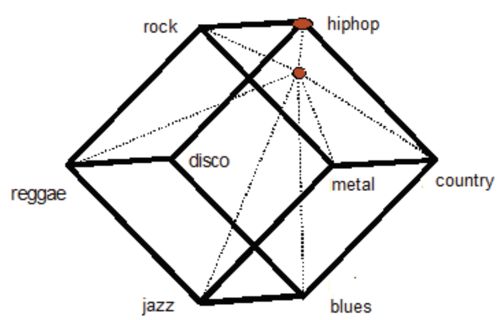
3.4.2 Video

Aside from representing the result of the SOM in a single image, it can also be represented in a video or multiple images. Video can be produced for the results of this research by collating each 1 second segment result in order to show the progression of the musical trajectory grow from the start of the symphony to the end. This allows clearer visualization of the data to have more accurate analysis. Using this kind of visualization also greatly helps the survey user in the outcome of this research.

3.4.3 3D Models

In Azcarraga, Caronongan, Setiono, & Manalili (2016)s research work, they incorporated the use of a structured 3D SOM instead of the regular SOM which will result in a single image. They represented the 3D map as a 3x3x3 dimensional cube with 27 subcubes each of the same sizes. Each subcube is further divided into 9x9x9 nodes. Here, they introduced the concept of a core cube at the center and the other 26 corresponding exterior cubes surrounding it. The training phase of the cube involved a four step labelling phase which was discussed in greater detail back in chapter 2. The resulting 3D SOM was then used to identify the proximity of a certain music to a particular genre. Each genre represented one corner of the cube as shown in Figure 3.4.

Figure 3.4: 3D SOM Cube



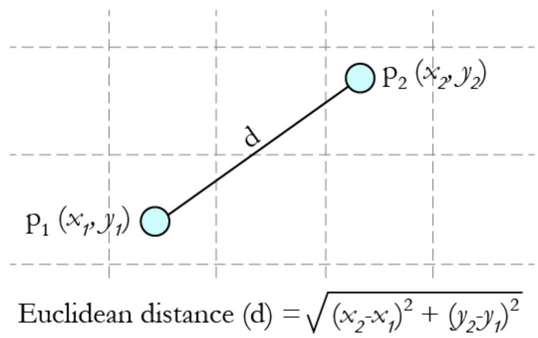
3.5 Metrics

There are two general types of data, qualitative and quantitative data. Qualitative data are data that cannot be measured by numbers while quantitative can be measured by numbers. Since this research will only make use of quantitative measurements, qualitative will no longer be discussed.

3.5.1 Quantitative

When using clustering as the method for machine learning, for example k-means clustering, there will result in k number of clusters after the algorithm is performed. Azcarraga & Flores (2016) used k-means clustering in clustering the 1 second music segments. The best matching unit (BMU) for each 1 second music segment is first computed using Euclidean distance, which is the square root of the square of the difference between the x-axis of the first and second point added to the square of the difference between the y-axis of the first and second point, as shown in Figure 3.5.

Figure 3.5: Euclidean Distance Diagram



Each time a 1 second music segment has a BMU inside a cluster, the frequency count for that cluster is incremented. In this way, only the clusters that are mainly used by the music or symphony will have a high frequency count. The frequency counts are then normalized by dividing the counts of a certain composition by its total number of 1 second music segments. Once these normalized frequency counts are summarized, the resulting percentages can then be used to perform pair-wise comparisons between symphonies as shown in Appendix D.

Chapter 4

Pipeline

This chapter contains procedures that propoenents will follow for the research based from theories and concepts discussed in chapter 3 and the methodologies discussed in chapter 1.

4.1 Data Collection

This phase will concern the expansion of the original music dataset for SOMphony. Each of the 5 eras will have 5 composers with 5 symphonies each. This will sum up to a total of 125 symphonies. The proponents have decided to only include composers that have composed at least 5 symphonies to be able to maintain a balanced data set. The process of selecting which symphonies to be added would be by random to have a better grasp of the general style of the composer. The audio files would be retrieved from online sources and physical means. The researchers would not take into consideration the file type and bitrate of the audio files since music data that is free for use is limited.

4.2 Data Preprocessing

The audio files will be trimmed using Audacity, removing the silent parts usually found at the start and/or end of the composition. This would reduce the amount of empty data, since no features can be extracted when there is no audio. After the music files are trimmed, they will be cut into one second music segments with a half-second overlap as discussed in Section 1.5.3 using Direct WAV MP3 Splitter.

4.3 Feature Extraction

The music segments will then have their features be extracted using jAudio, producing an XML file as an output. The researchers have decided to extract 436 features, refer to Appendix C for the complete list with their descriptions. These researchers have selected all the features that doesnt have variable-sized dimension. The XML file will then be converted to CSV format. The result would be an xml file containing all the features determined for each segment. The researchers would then run a RegEx script from SOMphony to extract the unnecessary text in preparation for labeling. The data needs to be labeled according to their composer, composition and file name.

4.4 Feature Selection

In this phase, the proponents will trim down the 436 features that jAudio has extracted to lessen the training time for the SOM.

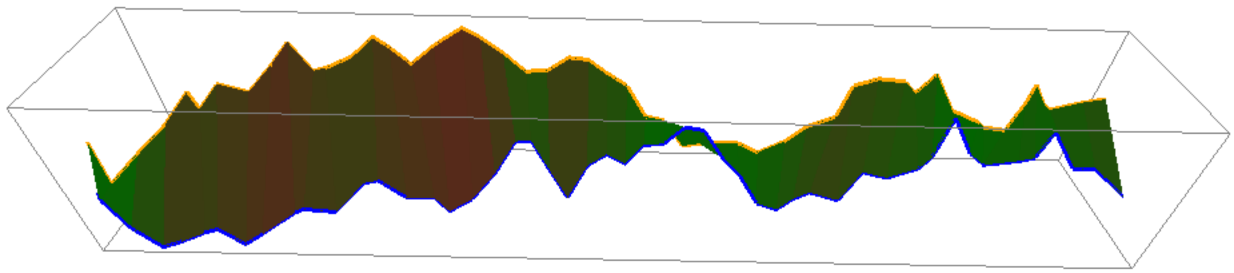
The proponents will explore different algorithms for feature selection. The proponents initially decided on using decision trees. As mentioned in Section 3.2.4, decision tree algorithms are best used for feature selection. However, there are other successfully tested algorithm for feature selection like Pearson's correlation coefficient, principal component analysis (PCA, chi-square and Fisher-like criterion.

4.5 Data Visualization

The proponents will visualize the SOM in 3D by plotting the BMU of each music segment on a 2D plane and then collating the results of all segments in the composition in time series. The result is a line T (x, y, z) where (x, y) denotes the coordinates of the BMU of a particular music segment on the SOM and z being the index of the segment in the time series. Displaying the data of one symphony will plot a line that represents the musical trajectory or progression of the symphony in the SOM from start to finish. Each point on the z-axis (longest axis) represents the position of the BMU on the SOM at a particular interval in the time series. Alternatively, when comparing two symphonies, two lines will be generated representing the musical trajectories of both symphonies.

As seen in Figure 4.1, the yellow and blue edges represent the trajectory of

Figure 4.1: 3D Visualization Sample



each of the symphonies as they progress through time. A line is drawn between two points from both symphonies to show the Euclidean distance between the BMUs at the particular time slice. The color of this line would change depending on the euclidean distance, which has a direct implication on the similarity of the two BMUs. Parts of the ribbon that appear to be green would signify that at these time slices, the symphonies sound similar. The parts that appear red would be the parts where the symphony will appear the least similar.

4.6 Evaluation

Frequency cluster counting will be used to determine if a pair of symphonies are alike. Each time a BMU appears inside a cluster, the frequency count for that cluster is incremented. In this way, the clusters frequently visited throughout the length of the musical piece, will have a large frequency count. The frequency counts are then normalized by dividing the counts of a certain composition by its total number of 1-second segments. Once these normalized frequency counts are summarized, the resulting percentages can then be used to perform pair-wise comparisons between symphonies.

The resulting visualization from SOM and t-SNE will then be used to verify both methods accuracy by comparing for example if two symphonies by composer A have a much similar visualization. If the resulting visualizations show that the compositions by the same composer are alike, then it may be able to prove the accuracy of the visualization technique used.

Chapter 5

Results and Analysis

This chapter contains all the results from the research conducted and the corresponding analyses for the results.

5.1 SOM

This phase will concern the expansion of the original music dataset for SOMphony. Each of the 5 eras will have 5 composers with 5 symphonies each. This will sum up to a total of 125 symphonies. The proponents have decided to only include composers that have composed at least 5 symphonies to be able to maintain a balanced data set. The process of selecting which symphonies to be added would be by random to have a better grasp of the general style of the composer. The audio files would be retrieved from online sources and physical means. The researchers would not take into consideration the file type and bitrate of the audio files since music data that is free for use is limited.

5.2 t-SNE

TSNE...

Row Labels	A	B	C	D	E	Grand Total
P1C1S1	383	43	6	152	1	585
P1C1S2	116	1609	2	16	710	2453

P1C1S3	4	3		1	795	803
P1C1S4	2	243				245
P1C1S5	1	352		3	2	358
P1C2S1	223	14			153	390
P1C2S2	4	841	3	31	1	880
P1C2S3	106	100			431	637
P1C2S4	479	18	5	5	154	661
P1C2S5	180	415	48	291	339	1273
P1C3S1	1	41			596	638
P1C3S2	450	73	5	32	71	631
P1C3S3		90		1	227	318
P1C3S4	2	10		387		399
P1C3S5	276	2		1	23	302
P1C4S1	58	99	20	82	22	281
P1C4S2	83	109		2	783	977
P1C4S3	297	286	83	408	57	1131
P1C4S4	104	78		4	1344	1530
P1C4S5	134	690	72	253	44	1193
P1C5S1	33	952	2	1	24	1012
P1C5S2	2	241		5	87	335
P1C5S3	12	338		1	111	462
P1C5S4	7	70			251	328
P1C5S5	268	13		2	133	416
P2C1S1	1369	643	228	414	121	2775
P2C1S2	1055	827	77	398	180	2537
P2C1S3	988	686	107	806	143	2730
P2C1S4	1808	534	22	217	425	3006
P2C1S5	478	148	18	90	232	966
P2C2S1	86	91	40	40	191	448
P2C2S2	180	258	60	270	110	878
P2C2S3	336	1059	104	180	373	2052
P2C2S4	353	287	39	32	454	1165
P2C2S5	545	709	34	121	667	2076
P2C3S1	279	369	12	22	249	931
P2C3S2	72	206	2	8	804	1092
P2C3S3	259	133	1	1	1218	1612
P2C3S4	623	192	1	4	864	1684
P2C3S5	504	123	6	81	599	1313
P2C4S1	1396	330	16	44	1164	2950
P2C4S2	764	612	421	265	845	2907

P2C4S3	1082	293	71	162	913	2521
P2C4S4	881	208	30	11	2112	3242
P2C4S5	1091	546	492	271	771	3171
P2C5S1	694	521	232	703	1446	3596
P2C5S2	700	653	476	1144	80	3053
P2C5S3	1375	373	539	792	281	3360
P2C5S4	1145	1119	629	1408	26	4327
P2C5S5	472	381	129	301	339	1622
P3C1S1	514	411	87	296	46	1354
P3C1S2	246	103	14	84	28	475
P3C1S3	1460	258	65	112	498	2393
P3C1S4	1032	102	15	29	933	2111
P3C1S5	47	116	2	3	1109	1277
P3C2S1	36	3	1653	237	2	1931
P3C2S2	371	70	10	1	554	1006
P3C2S3	461	41	9		531	1042
P3C2S4	1048	102	80	199	574	2003
P3C2S5	487	73	434	444	15	1453
P3C3S1	32		1351	724		2107
P3C3S2	18		1068	598	5	1689
P3C3S3	173		1568	245	4	1990
P3C3S4	54	10	3109	1114	4	4291
P3C3S5	42		1139	521		1702
P3C4S1	1028	139	53	378	462	2060
P3C4S2	414	316	118	250	90	1188
P3C4S3	2275	340	135	192	346	3288
P3C4S4	1021	522	108	145	469	2265
P3C4S5	602	194	6	13	879	1694
P3C5S1	592	311	377	347	317	1944
P3C5S2	1493	740	244	546	101	3124
P3C5S3	1753	286	129	329	22	2519
P3C5S4	713	126	14	28	938	1819
P3C5S5	27	70	2	4	723	826
P4C1S1	429	316	106	186	143	1180
P4C1S2	820	2985	264	1919	58	6046
P4C1S3	1145	1717	357	621	307	4147
P4C1S4	1088	519	54	107	346	2114
P4C1S5	612	215	2	88	424	1341
P4C2S1	438	182	1	100	2682	3403
P4C2S2	1456	575	207	490	115	2843

P4C2S3	2016	1059	103	332	524	4034
P4C2S4	1329	354	17	46	804	2550
P4C2S5	593	94		15	848	1550
P4C3S1	15		2838	1617	21	4491
P4C3S2	7		2373	1664	8	4052
P4C3S3	46		168	5139		5353
P4C3S4	12	253	153	3404	12	3834
P4C3S5	4	3885		62	99	4050
P4C4S1	889	497		15	1337	2738
P4C4S2	1195	566	2	201	910	2874
P4C4S3	1875	66	212	287	342	2782
P4C4S4	138	1	3393	25		3557
P4C4S5	40	7	3044	60		3151
P4C5S1	379	367	243	401	70	1460
P4C5S2	673	250	204	327	140	1594
P4C5S3	1866	331	157	398	408	3160
P4C5S4	672	379	7	104	476	1638
P4C5S5	168	354	33	87	842	1484
P5C1S1	46	471	1397	3291	101	5306
P5C1S2	158	15	3637	1511		5321
P5C1S3	150	21	2394	1556		4121
P5C1S4	472	11	3292	1729		5504
P5C1S5	652	13	2826	2141		5632
P5C2S1	2968	1071	756	822	75	5692
P5C2S2	2594	683	1211	1497	157	6142
P5C2S3	2612	217	63	201	1489	4582
P5C2S4	341	1070	94	114	4240	5859
P5C2S5	1856	434	377	325	207	3199
P5C2S5C	1101	103	231	69	4	1508
P5C2S5D	854	317	82	132	174	1559
P5C3S1	1194	1907	618	1215	493	5427
P5C3S2	2590	1264	417	734	564	5569
P5C3S3	2975	1009	212	786	419	5401
P5C3S4	1460	534	20	730	1418	4162
P5C3S5	2023	707	8	299	2572	5609
P5C4S1	585	1076	911	1186	165	3923
P5C4S2	292	1955	1144	1943	111	5445
P5C4S3	624	2737	428	1329	552	5670
P5C4S4	641	2713	719	1160	286	5519
P5C4S5	1242	2513	299	561	424	5039

P5C5S1	234	2447	6	360	3036	6083
P5C5S2	58	3071	22	26	938	4115
P5C5S3	2613	665	8	333	2554	6173
P5C5S4	251	762		170	4067	5250
P5C5S5	793	745	913	2509	777	5737
Grand Total	86983	65867	51845	60726	65355	330776

Figure 5.1: All Symphonies

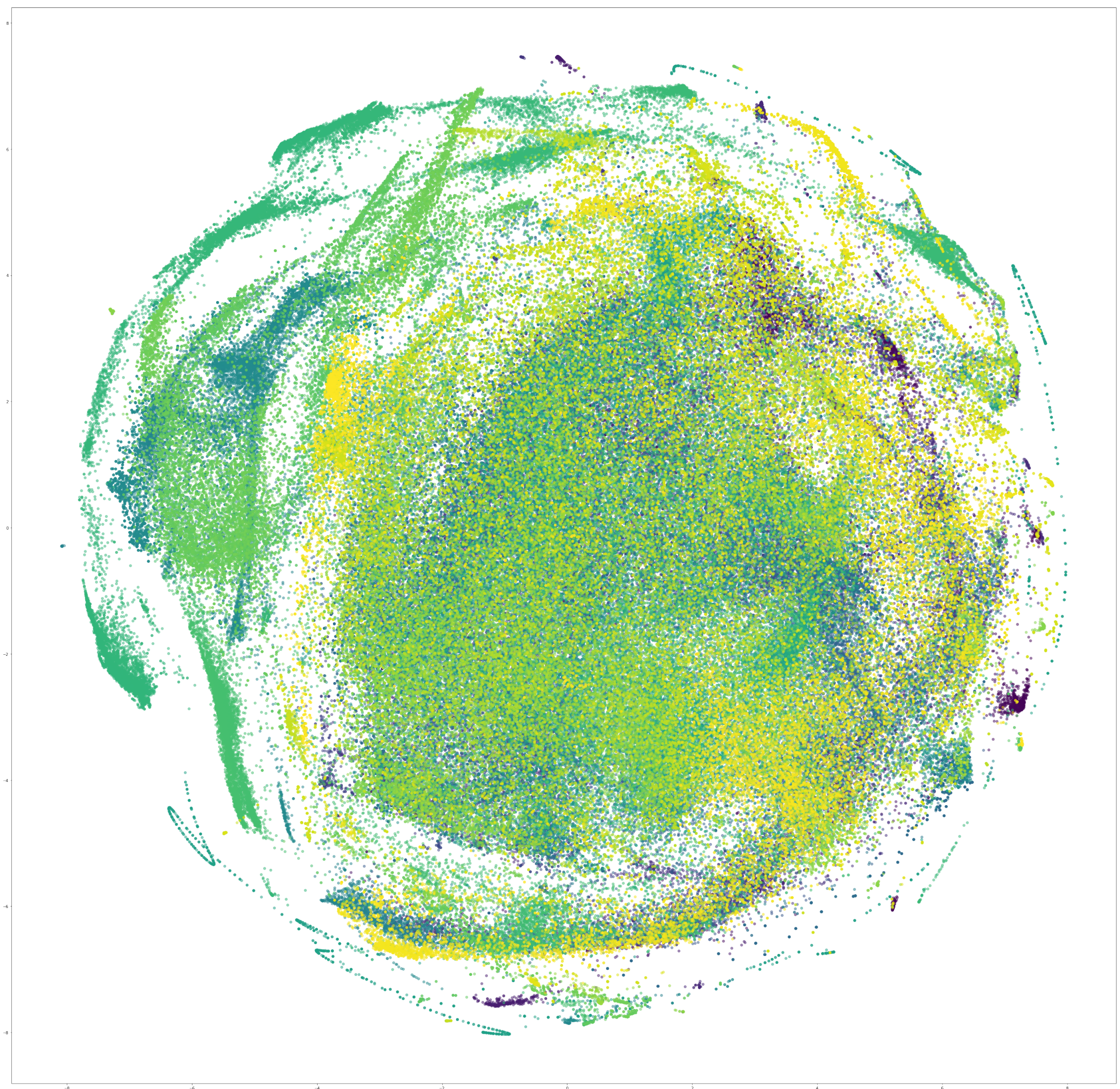


Figure 5.2: Baroque Period

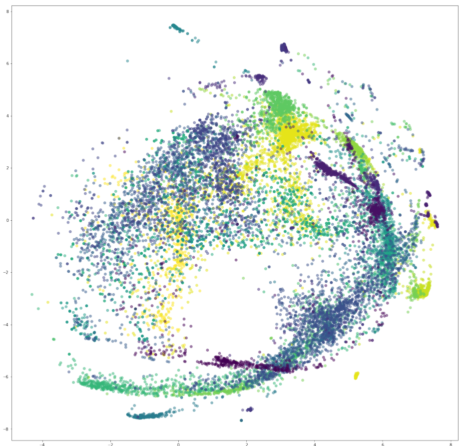


Figure 5.3: Classical Period

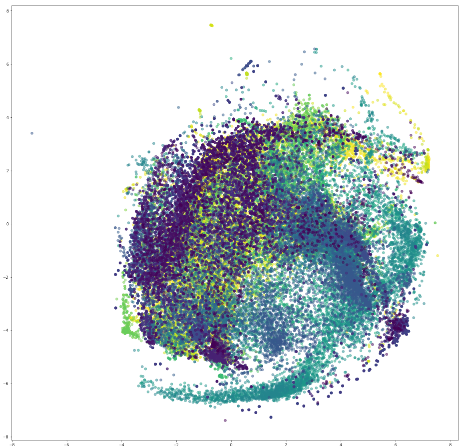


Figure 5.4: 19th Century

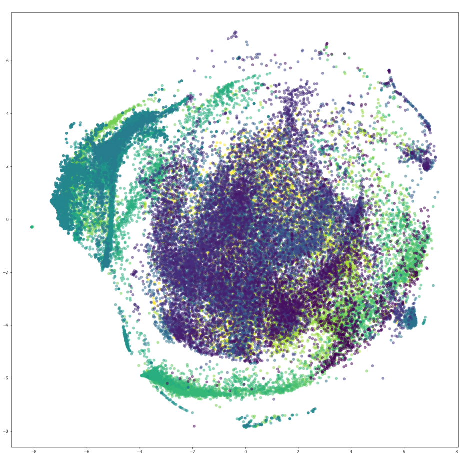


Figure 5.5: Romantic Period

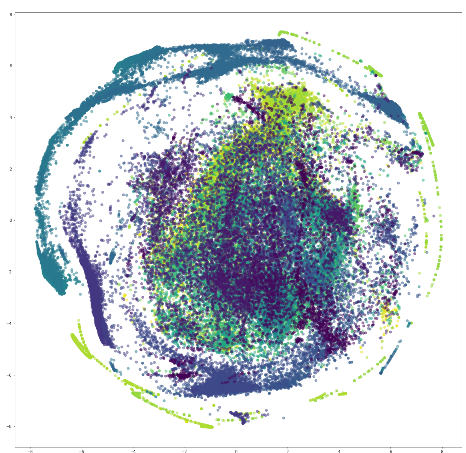


Figure 5.6: 20th Century

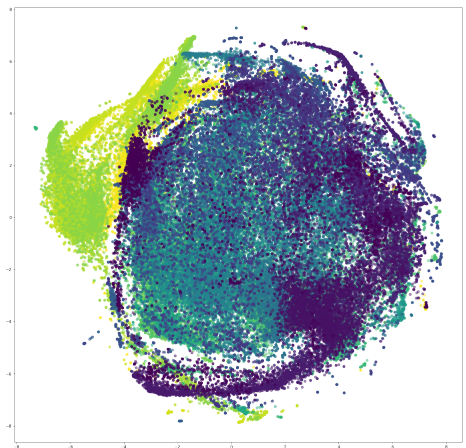


Figure 5.7: Baroque: Bach

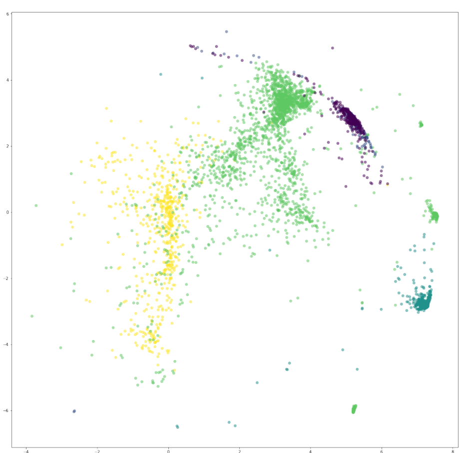


Figure 5.8: Baroque: Boyce

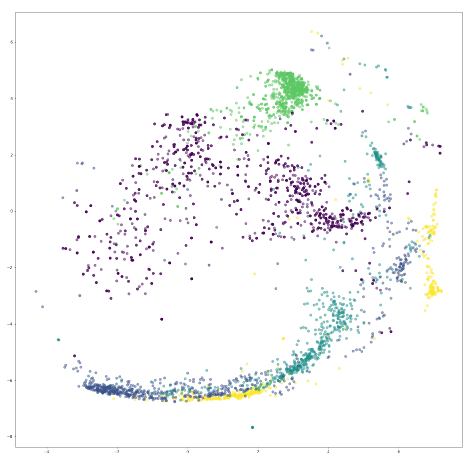


Figure 5.9: Baroque: Gabrieli

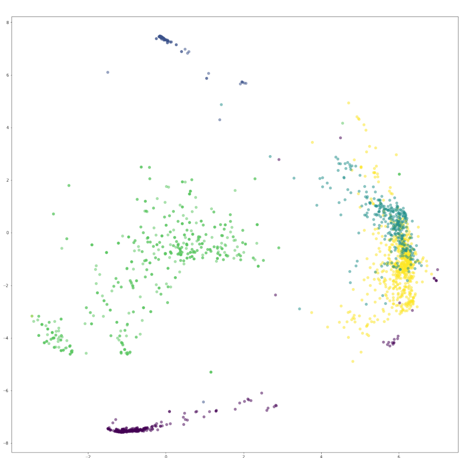


Figure 5.10: Baroque: Sammartini

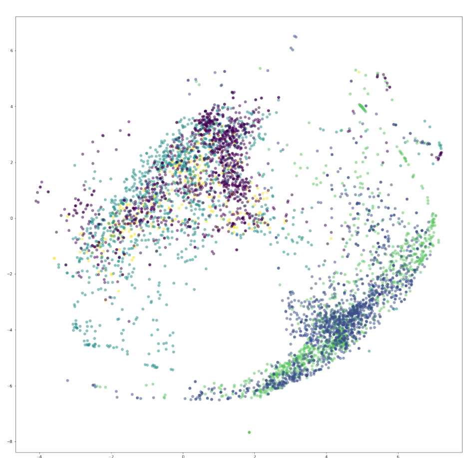


Figure 5.11: Baroque: Viadana

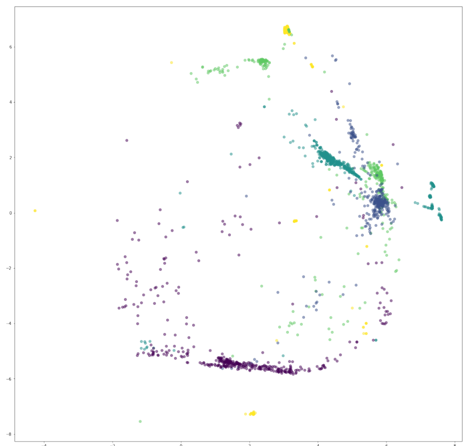


Figure 5.12: Classical: Boccherini

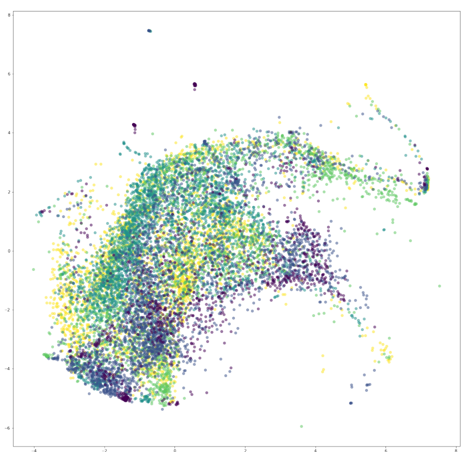


Figure 5.13: Classical: Gluck

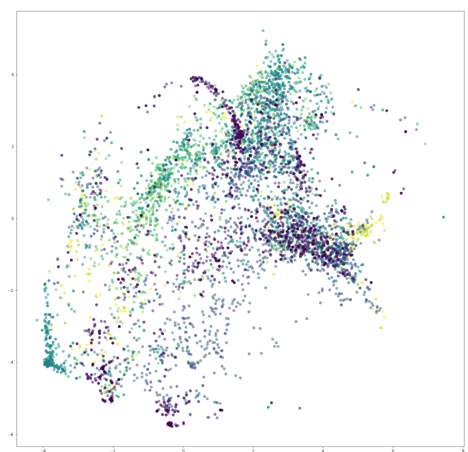


Figure 5.14: Classical: Haydn

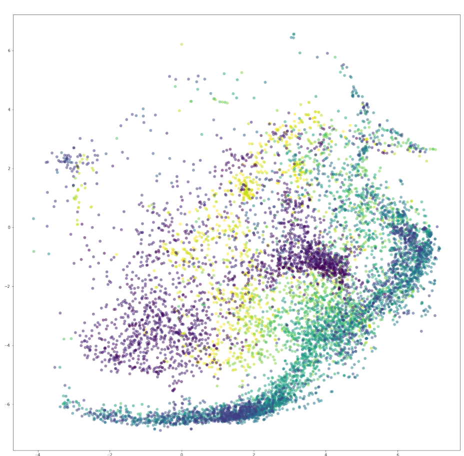


Figure 5.15: Classical: Mozart

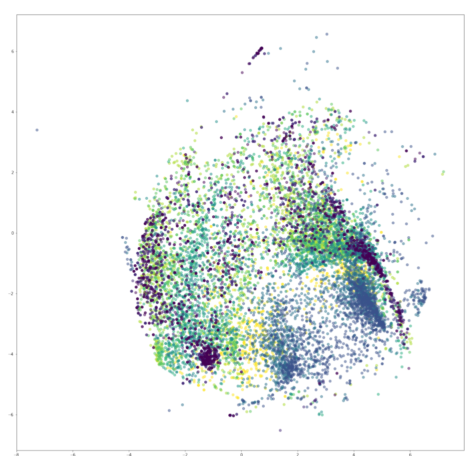


Figure 5.16: Classical: Salieri

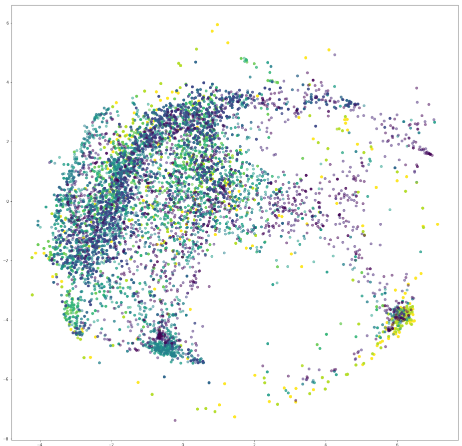


Figure 5.17: 19th Century: Beethoven

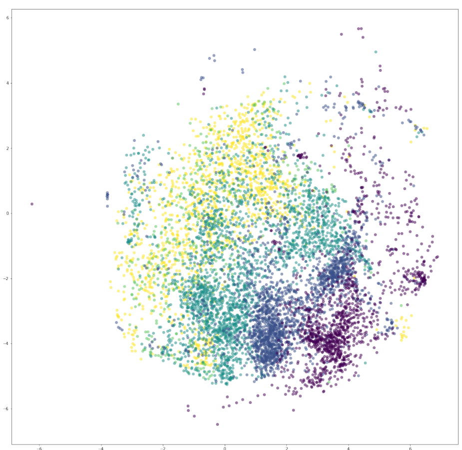


Figure 5.18: 19th Century: Clementi

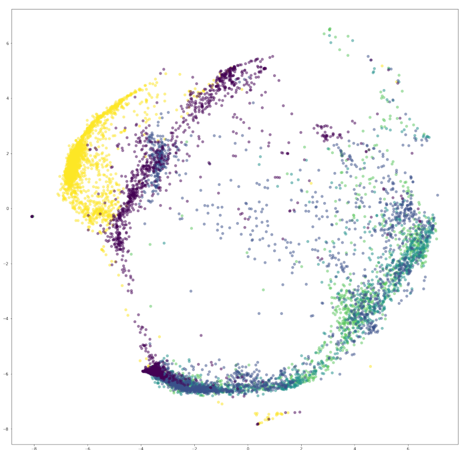


Figure 5.19: 19th Century: Gossec

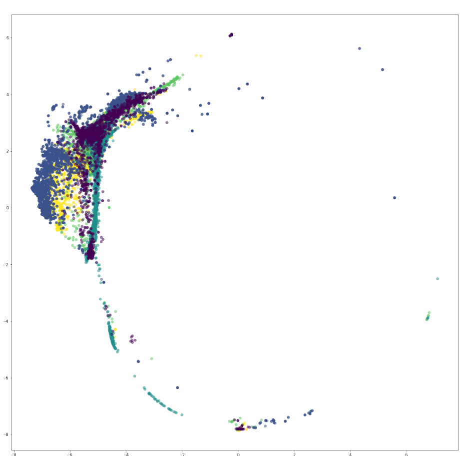


Figure 5.20: 19th Century: Kalliwoda

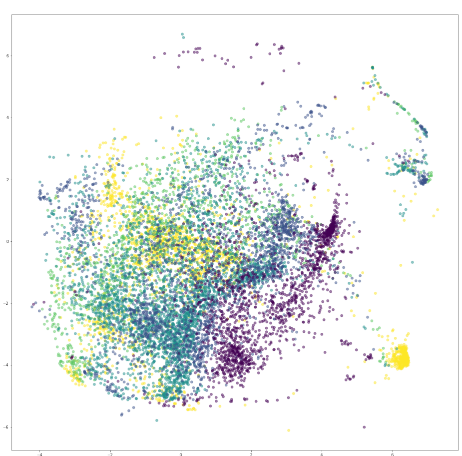


Figure 5.21: 19th Century: Rubinstein

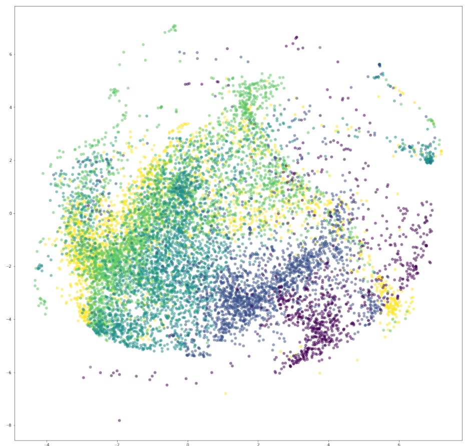


Figure 5.22: Romantic: Dvorak

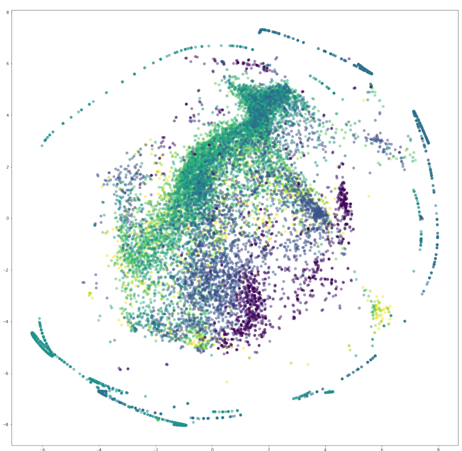


Figure 5.23: Romantic: Mendelssohn

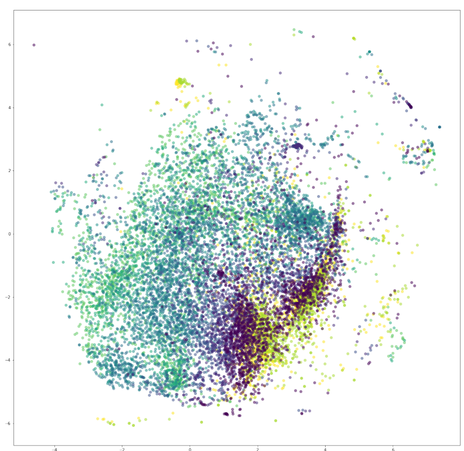


Figure 5.24: Romantic: Schubert

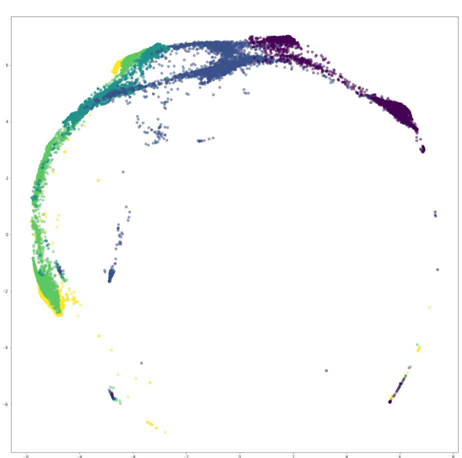


Figure 5.25: Romantic: Schumann

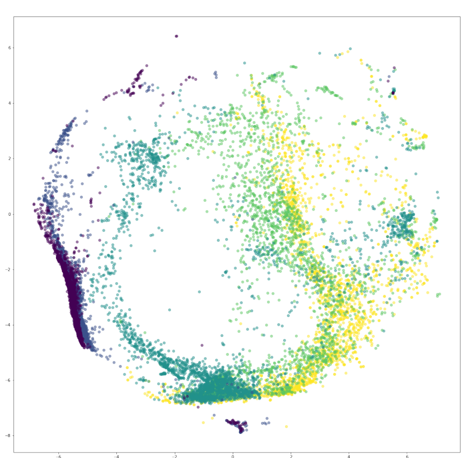


Figure 5.26: Romantic: Tchaikovsky

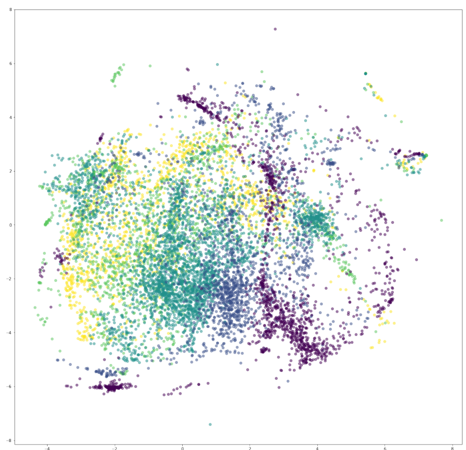


Figure 5.27: 20th Cen: Antheil

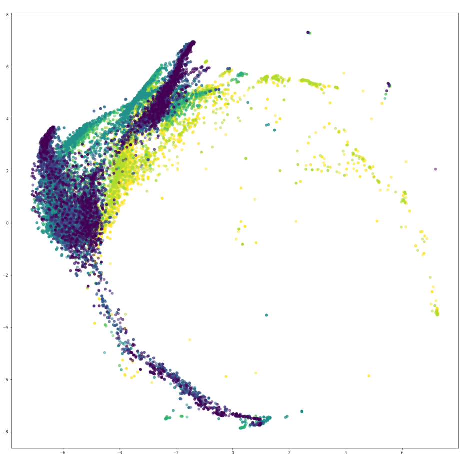


Figure 5.28: 20th Cen: Rachmaninoff

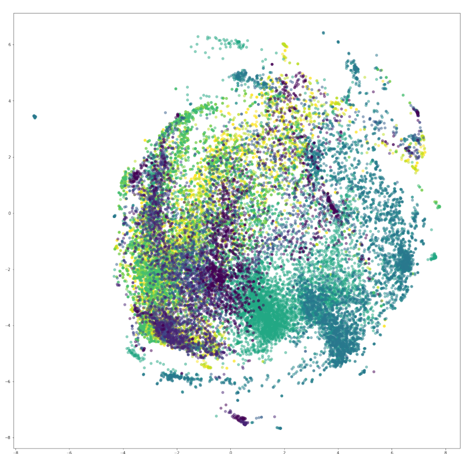


Figure 5.29: 20th Cen: Rubbra

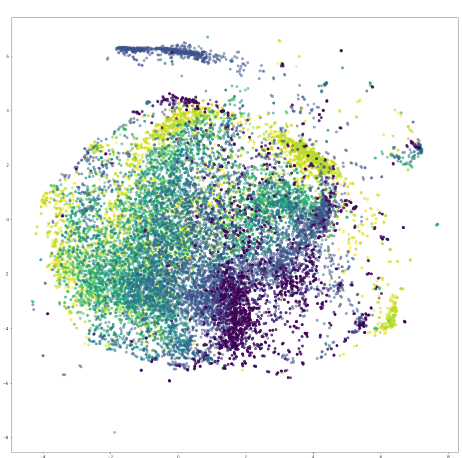


Figure 5.30: 20th Cen: Shostakovich

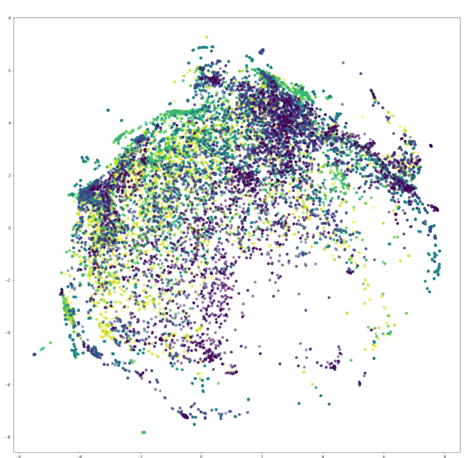
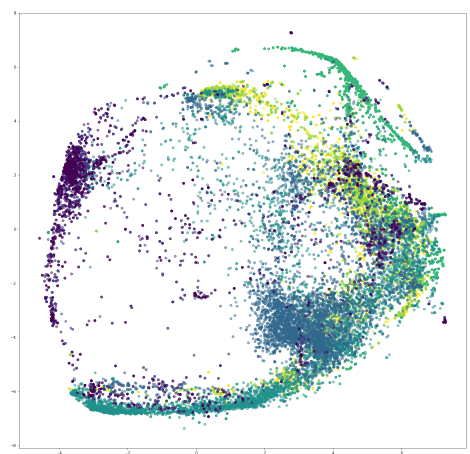


Figure 5.31: 20th Cen: Stravinsky



Appendix A

Research Ethics Documents

This section contains all documents related to research ethics.

DE LA SALLE UNIVERSITY
General Research Ethics Checklist

This checklist is to ensure that the research conducted by the faculty members and students of De La Salle University is carried out according to the guiding principles outlined in the Code of Research Ethics of the University. The investigator is advised to refer to the De La Salle University Code of Research Ethics and Guide to Responsible Conduct of Research before completing this checklist. Statements pertinent to ethical issues in research should be addressed below. The checklist will help the researchers and evaluators determine whether procedures should be undertaken during the course of the research to maintain ethical standards. The University's Guide to the Responsible Conduct of Research provides details on these appropriate procedures.

Details of the Research	
Students	Cruz, Edwardo Dionisio, Jefferson Fukouka, Kenji Portales, Naomi
Thesis Adviser	Flores, Fritz
Department	Software Technology Department
Title of the Research	Music Visualization Using Projections to 2D Maps
Term(s) and Academic year in which research is to be conducted	AY 2017-2018 Term 1,2,3

This checklist must be completed AFTER the* De La Salle University Code of Ethics *has been read and BEFORE gathering data.

Questions	Yes	No
1. Does your research involve human participants (this includes new data gathered or using pre-existing data)? If your answer is **yes**, please answer **Checklist A (Human Participants).**		✔
2. Does your research involve animals (non-human subjects)? If your answer is **yes**, please answer **Checklist B (Animal Subjects).**		✔
3. Does your research involve Wildlife? If your answer is **yes**, please answer **Checklist C (Wildlife)**.		✔
4. Does your research involve microorganisms that are infectious, disease causing or harmful to health? If your answer is **yes**, please answer **Checklist D (Infectious Agents).**		✔

5. Does your research involve toxic/chemicals/ substances/materials? If your answer is **yes**, please answer **Checklist E (Toxic Agents).**		✔

Research with Ethical Issues to address:

If you have a YES answer to any of the above categories, you will be required to complete a detailed checklist for that particular category. A YES answer does not mean the disapproval of your research proposal. By providing you with a more detailed checklist, we ensure that the ethical concerns are identified so these can be addressed in adherence to the University Code of Ethics.

Declaration of Conflict of Interest

[✔] I do not have a conflict of interest in any form (personal, financial, proprietary, or professional) with the sponsor/grant-giving organization, the study, the co-investigators/personnel, or the site.

[ ] I have a personal/family or professional interest in the results of the study (family members who are co-proponents or personnel in the study, membership in relevant professional associations/organizations).

Please describe the personal/family or professional interest:

[ ] I have propriety interest vested in this proposal (with the intent to apply for a patent, trademark, copyright, or license)

Please describe propriety interest:

[ ] I have significant financial interest vested in this proposal (remuneration that exceeds P250,000.00 each year or equity interest in the form of stock, stock options or other ownership interests).

Please describe financial interest:

Declaration

We certify that we have read and understand the De La Salle University Code for the Responsible Conduct of Research and will abide by the ethical principles in this document. We will submit a final report of the proposed study to the DLSU-Research Ethics Office. We will not commence with data collection until we receive an ethics review approval from the College Research Ethics Committee.
Name and Signature of Student 1   Name and Signature of Student 2 Name and Signature of Student 3   Name and Signature of Student 4

Endorsement from thesis adviser to the thesis panel for proposal defense…
Name and Signature of Adviser   Date

Endorsement from thesis adviser to the thesis panel for final defense… This is to certify that the research was conducted in a manner that adheres to ethical research standards. I am thus endorsing the group for final defense.
Name and Signature of Adviser   Date

RESEARCH ETHICS CLEARANCE FORM For Thesis Proposals[1]	
Names of student researcher/s :	*Cruz, Edwardo* *Dionisio, Jefferson* *Fukuoka, Kenji* *Portales, Naomi*
College:	College of Computer Studies
Department:	Software Technology Department
Course:	BS Computer Science with specialization in Software Technology
Expected duration of project:	from: September 2017 to: July 2018

Ethical considerations

None

To the best of our knowledge, the ethical issues listed above have been addressed in the research.

____

Name and signature of adviser/mentor

Date:

______________________________ ______________________________

Name and signature of panelist **Name and signature of panelist**

Date: **Date:**

[1]The same form can be used for the reports of completed projects. The appropriate heading need only be used.

Appendix B

Turnitin Similarity Report

This section consists of the first page of the Turnitin Originality Report.

Music Visualization Using Projections to 2D Maps

ORIGINALITY REPORT

18%	11%	11%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.jmlr.org Internet Source	2%
2	Arnulfo Azcarraga, Arturo Caronongan, Rudy Setiono, Sean Manalili. "Validating the stable clustering of songs in a structured 3D SOM", 2016 International Joint Conference on Neural Networks (IJCNN), 2016 Publication	1%
3	Submitted to De La Salle University - Manila Student Paper	1%
4	"Neural Information Processing", Springer Nature, 2016 Publication	1%
5	Corrêa, Débora C., and Francisco Ap. Rodrigues. "A survey on symbolic data-based music genre classification", Expert Systems with Applications, 2016. Publication	1%
6	netcentric.dlsu.edu.ph Internet Source	1%

Appendix C

List of Features and Definitions

This section consists of the list of features extractable in jAudio and their corresponding definitions.

Spectral Centroid Overall Standard Deviation	The centre of mass of the power spectrum. This is the overall standard deviation over all windows.
Derivative of Spectral Centroid Overall Standard Deviation	Derivative of Spectral Centroid. The centre of mass of the power spectrum. This is the overall standard deviation over all windows.
Running Mean of Spectral Centroid Overall Standard Deviation	Running Mean of Spectral Centroid. The centre of mass of the power spectrum. This is the overall standard deviation over all windows.
Standard Deviation of Spectral Centroid Overall Standard Deviation	Standard Deviation of Spectral Centroid. The centre of mass of the power spectrum. This is the overall standard deviation over all windows.
Derivative of Running Mean of Spectral Centroid Overall Standard Deviation	Derivative of Running Mean of Spectral Centroid. Running Mean of Spectral Centroid. The centre of mass of the power spectrum. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Spectral Centroid Overall Standard Deviation	Derivative of Standard Deviation of Spectral Centroid. Standard Deviation of Spectral Centroid. The centre of mass of the power spectrum. This is the overall standard deviation over all windows.
Spectral Rolloff Point Overall Standard Deviation	The fraction of bins in the power spectrum at which 85% of the power is at lower frequencies. This is a measure of the right-skewedness of the power spectrum. This is the overall standard deviation over all windows.
Derivative of Spectral Rolloff Point Overall Standard Deviation	Derivative of Spectral Rolloff Point. The fraction of bins in the power spectrum at which 85% of the power is at lower frequencies. This is a measure of the right-skewedness of the power spectrum. This is the overall standard deviation over all windows.
Running Mean of Spectral Rolloff Point Overall Standard Deviation	Running Mean of Spectral Rolloff Point. The fraction of bins in the power spectrum at which 85% of the power is at lower frequencies. This is a measure of the right-skewedness of the power spectrum. This is the overall standard deviation over all windows.
Standard Deviation of Spectral Rolloff Point Overall Standard Deviation	Standard Deviation of Spectral Rolloff Point. The fraction of bins in the power spectrum at which 85% of the power is at lower frequencies. This is a measure of the right-skewedness of the power spectrum. This is the overall standard deviation over all windows.
Derivative of Running Mean of Spectral Rolloff Point Overall Standard Deviation	Derivative of Running Mean of Spectral Rolloff Point. Running Mean of Spectral Rolloff Point. The fraction of bins in the power spectrum at which 85% of the power is at lower frequencies.

	This is a measure of the right-skewedness of the power spectrum. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Spectral Rolloff Point Overall Standard Deviation	Derivative of Standard Deviation of Spectral Rolloff Point. Standard Deviation of Spectral Rolloff Point. The fraction of bins in the power spectrum at which 85% of the power is at lower frequencies. This is a measure of the right-skewedness of the power spectrum. This is the overall standard deviation over all windows.
Spectral Flux Overall Standard Deviation	A measure of the amount of spectral change in a signal. Found by calculating the change in the magnitude spectrum from frame to frame. This is the overall standard deviation over all windows.
Derivative of Spectral Flux Overall Standard Deviation	Derivative of Spectral Flux. A measure of the amount of spectral change in a signal. Found by calculating the change in the magnitude spectrum from frame to frame. This is the overall standard deviation over all windows.
Running Mean of Spectral Flux Overall Standard Deviation	Running Mean of Spectral Flux. A measure of the amount of spectral change in a signal. Found by calculating the change in the magnitude spectrum from frame to frame. This is the overall standard deviation over all windows.
Standard Deviation of Spectral Flux Overall Standard Deviation	Standard Deviation of Spectral Flux. A measure of the amount of spectral change in a signal. Found by calculating the change in the magnitude spectrum from frame to frame. This is the overall standard deviation over all windows.
Derivative of Running Mean of Spectral Flux Overall Standard Deviation	Derivative of Running Mean of Spectral Flux. Running Mean of Spectral Flux. A measure of the amount of spectral change in a signal. Found by calculating the change in the magnitude spectrum from frame to frame. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Spectral Flux Overall Standard Deviation	Derivative of Standard Deviation of Spectral Flux. Standard Deviation of Spectral Flux. A measure of the amount of spectral change in a signal. Found by calculating the change in the magnitude spectrum from frame to frame. This is the overall standard deviation over all windows.
Compactness Overall Standard Deviation	A measure of the noisiness of a signal. Found by comparing the components of a window's magnitude spectrum with the magnitude spectrum of its neighbouring windows. This is the overall standard deviation over all windows.

Derivative of Compactness Overall Standard Deviation	Derivative of Compactness. A measure of the noisiness of a signal. Found by comparing the components of a window's magnitude spectrum with the magnitude spectrum of its neighbouring windows. This is the overall standard deviation over all windows.
Running Mean of Compactness Overall Standard Deviation	Running Mean of Compactness. A measure of the noisiness of a signal. Found by comparing the components of a window's magnitude spectrum with the magnitude spectrum of its neighbouring windows. This is the overall standard deviation over all windows.
Standard Deviation of Compactness Overall Standard Deviation	Standard Deviation of Compactness. A measure of the noisiness of a signal. Found by comparing the components of a window's magnitude spectrum with the magnitude spectrum of its neighbouring windows. This is the overall standard deviation over all windows.
Derivative of Running Mean of Compactness Overall Standard Deviation	Derivative of Running Mean of Compactness. Running Mean of Compactness. A measure of the noisiness of a signal. Found by comparing the components of a window's magnitude spectrum with the magnitude spectrum of its neighbouring windows. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Compactness Overall Standard Deviation	Derivative of Standard Deviation of Compactness. Standard Deviation of Compactness. A measure of the noisiness of a signal. Found by comparing the components of a window's magnitude spectrum with the magnitude spectrum of its neighbouring windows. This is the overall standard deviation over all windows.
Spectral Variability Overall Standard Deviation	The standard deviation of the magnitude spectrum. This is a measure of the variance of a signal's magnitude spectrum. This is the overall standard deviation over all windows.
Derivative of Spectral Variability Overall Standard Deviation	Derivative of Spectral Variability. The standard deviation of the magnitude spectrum. This is a measure of the variance of a signal's magnitude spectrum. This is the overall standard deviation over all windows.
Running Mean of Spectral Variability Overall Standard Deviation	Running Mean of Spectral Variability. The standard deviation of the magnitude spectrum. This is a measure of the variance of a signal's magnitude spectrum. This is the overall standard deviation over all windows.

Standard Deviation of Spectral Variability Overall Standard Deviation	Standard Deviation of Spectral Variability. The standard deviation of the magnitude spectrum. This is a measure of the variance of a signal's magnitude spectrum. This is the overall standard deviation over all windows.
Derivative of Running Mean of Spectral Variability Overall Standard Deviation	Derivative of Running Mean of Spectral Variability. Running Mean of Spectral Variability. The standard deviation of the magnitude spectrum. This is a measure of the variance of a signal's magnitude spectrum. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Spectral Variability Overall Standard Deviation	Derivative of Standard Deviation of Spectral Variability. Standard Deviation of Spectral Variability. The standard deviation of the magnitude spectrum. This is a measure of the variance of a signal's magnitude spectrum. This is the overall standard deviation over all windows.
Root Mean Square Overall Standard Deviation	A measure of the power of a signal. This is the overall standard deviation over all windows.
Derivative of Root Mean Square Overall Standard Deviation	Derivative of Root Mean Square. A measure of the power of a signal. This is the overall standard deviation over all windows.
Running Mean of Root Mean Square Overall Standard Deviation	Running Mean of Root Mean Square. A measure of the power of a signal. This is the overall standard deviation over all windows.
Standard Deviation of Root Mean Square Overall Standard Deviation	Standard Deviation of Root Mean Square. A measure of the power of a signal. This is the overall standard deviation over all windows.
Derivative of Running Mean of Root Mean Square Overall Standard Deviation	Derivative of Running Mean of Root Mean Square. Running Mean of Root Mean Square. A measure of the power of a signal. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Root Mean Square Overall Standard Deviation	Derivative of Standard Deviation of Root Mean Square. Standard Deviation of Root Mean Square. A measure of the power of a signal. This is the overall standard deviation over all windows.
Fraction Of Low Energy Windows Overall Standard Deviation	The fraction of the last 100 windows that has an RMS less than the mean RMS in the last 100 windows. This can indicate how much of a signal is quiet relative to the rest of the signal. This is the overall standard deviation over all windows.
Derivative of Fraction Of Low Energy Windows Overall Standard Deviation	Derivative of Fraction Of Low Energy Windows. The fraction of the last 100 windows that has an RMS less than the mean RMS in the last 100 windows. This can indicate how much of a signal is quiet relative to the rest of the signal. This is the overall standard deviation over all windows.

Running Mean of Fraction Of Low Energy Windows Overall Standard Deviation	Running Mean of Fraction Of Low Energy Windows. The fraction of the last 100 windows that has an RMS less than the mean RMS in the last 100 windows. This can indicate how much of a signal is quiet relative to the rest of the signal. This is the overall standard deviation over all windows.
Standard Deviation of Fraction Of Low Energy Windows Overall Standard Deviation	Standard Deviation of Fraction Of Low Energy Windows. The fraction of the last 100 windows that has an RMS less than the mean RMS in the last 100 windows. This can indicate how much of a signal is quiet relative to the rest of the signal. This is the overall standard deviation over all windows.
Derivative of Running Mean of Fraction Of Low Energy Windows Overall Standard Deviation	Derivative of Running Mean of Fraction Of Low Energy Windows. Running Mean of Fraction Of Low Energy Windows. The fraction of the last 100 windows that has an RMS less than the mean RMS in the last 100 windows. This can indicate how much of a signal is quiet relative to the rest of the signal. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Fraction Of Low Energy Windows Overall Standard Deviation	Derivative of Standard Deviation of Fraction Of Low Energy Windows. Standard Deviation of Fraction Of Low Energy Windows. The fraction of the last 100 windows that has an RMS less than the mean RMS in the last 100 windows. This can indicate how much of a signal is quiet relative to the rest of the signal. This is the overall standard deviation over all windows.
Zero Crossings Overall Standard Deviation	The number of times the waveform changed sign. An indication of frequency as well as noisiness. This is the overall standard deviation over all windows.
Derivative of Zero Crossings Overall Standard Deviation	Derivative of Zero Crossings. The number of times the waveform changed sign. An indication of frequency as well as noisiness. This is the overall standard deviation over all windows.
Running Mean of Zero Crossings Overall Standard Deviation	Running Mean of Zero Crossings. The number of times the waveform changed sign. An indication of frequency as well as noisiness. This is the overall standard deviation over all windows.
Standard Deviation of Zero Crossings Overall Standard Deviation	Standard Deviation of Zero Crossings. The number of times the waveform changed sign. An indication of frequency as well as noisiness. This is the overall standard deviation over all windows.

Derivative of Running Mean of Zero Crossings Overall Standard Deviation	Derivative of Running Mean of Zero Crossings. Running Mean of Zero Crossings. The number of times the waveform changed sign. An indication of frequency as well as noisiness. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Zero Crossings Overall Standard Deviation	Derivative of Standard Deviation of Zero Crossings. Standard Deviation of Zero Crossings. The number of times the waveform changed sign. An indication of frequency as well as noisiness. This is the overall standard deviation over all windows.
Strongest Beat Overall Standard Deviation	The strongest beat in a signal, in beats per minute, found by finding the strongest bin in the beat histogram. This is the overall standard deviation over all windows.
Derivative of Strongest Beat Overall Standard Deviation	Derivative of Strongest Beat. The strongest beat in a signal, in beats per minute, found by finding the strongest bin in the beat histogram. This is the overall standard deviation over all windows.
Running Mean of Strongest Beat Overall Standard Deviation	Running Mean of Strongest Beat. The strongest beat in a signal, in beats per minute, found by finding the strongest bin in the beat histogram. This is the overall standard deviation over all windows.
Standard Deviation of Strongest Beat Overall Standard Deviation	Standard Deviation of Strongest Beat. The strongest beat in a signal, in beats per minute, found by finding the strongest bin in the beat histogram. This is the overall standard deviation over all windows.
Derivative of Running Mean of Strongest Beat Overall Standard Deviation	Derivative of Running Mean of Strongest Beat. Running Mean of Strongest Beat. The strongest beat in a signal, in beats per minute, found by finding the strongest bin in the beat histogram. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Strongest Beat Overall Standard Deviation	Derivative of Standard Deviation of Strongest Beat. Standard Deviation of Strongest Beat. The strongest beat in a signal, in beats per minute, found by finding the strongest bin in the beat histogram. This is the overall standard deviation over all windows.
Beat Sum Overall Standard Deviation	The sum of all entries in the beat histogram. This is a good measure of the importance of regular beats in a signal. This is the overall standard deviation over all windows.
Derivative of Beat Sum Overall Standard Deviation	Derivative of Beat Sum. The sum of all entries in the beat histogram. This is a good measure of

	the importance of regular beats in a signal. This is the overall standard deviation over all windows.
Running Mean of Beat Sum Overall Standard Deviation	Running Mean of Beat Sum. The sum of all entries in the beat histogram. This is a good measure of the importance of regular beats in a signal. This is the overall standard deviation over all windows.
Standard Deviation of Beat Sum Overall Standard Deviation	Standard Deviation of Beat Sum. The sum of all entries in the beat histogram. This is a good measure of the importance of regular beats in a signal. This is the overall standard deviation over all windows.
Derivative of Running Mean of Beat Sum Overall Standard Deviation	Derivative of Running Mean of Beat Sum. Running Mean of Beat Sum. The sum of all entries in the beat histogram. This is a good measure of the importance of regular beats in a signal. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Beat Sum Overall Standard Deviation	Derivative of Standard Deviation of Beat Sum. Standard Deviation of Beat Sum. The sum of all entries in the beat histogram. This is a good measure of the importance of regular beats in a signal. This is the overall standard deviation over all windows.
Strength Of Strongest Beat Overall Standard Deviation	How strong the strongest beat in the beat histogram is compared to other potential beats. This is the overall standard deviation over all windows.
Derivative of Strength Of Strongest Beat Overall Standard Deviation	Derivative of Strength Of Strongest Beat. How strong the strongest beat in the beat histogram is compared to other potential beats. This is the overall standard deviation over all windows.
Running Mean of Strength Of Strongest Beat Overall Standard Deviation	Running Mean of Strength Of Strongest Beat. How strong the strongest beat in the beat histogram is compared to other potential beats. This is the overall standard deviation over all windows.
Standard Deviation of Strength Of Strongest Beat Overall Standard Deviation	Standard Deviation of Strength Of Strongest Beat. How strong the strongest beat in the beat histogram is compared to other potential beats. This is the overall standard deviation over all windows.
Derivative of Running Mean of Strength Of Strongest Beat Overall Standard Deviation	Derivative of Running Mean of Strength Of Strongest Beat. Running Mean of Strength Of Strongest Beat. How strong the strongest beat in the beat histogram is compared to other

	potential beats. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Strength Of Strongest Beat Overall Standard Deviation	Derivative of Standard Deviation of Strength Of Strongest Beat. Standard Deviation of Strength Of Strongest Beat. How strong the strongest beat in the beat histogram is compared to other potential beats. This is the overall standard deviation over all windows.
Strongest Frequency Via Zero Crossings Overall Standard Deviation	The strongest frequency component of a signal, in Hz, found via the number of zero-crossings. This is the overall standard deviation over all windows.
Derivative of Strongest Frequency Via Zero Crossings Overall Standard Deviation	Derivative of Strongest Frequency Via Zero Crossings. The strongest frequency component of a signal, in Hz, found via the number of zero-crossings. This is the overall standard deviation over all windows.
Running Mean of Strongest Frequency Via Zero Crossings Overall Standard Deviation	Running Mean of Strongest Frequency Via Zero Crossings. The strongest frequency component of a signal, in Hz, found via the number of zero-crossings. This is the overall standard deviation over all windows.
Standard Deviation of Strongest Frequency Via Zero Crossings Overall Standard Deviation	Standard Deviation of Strongest Frequency Via Zero Crossings. The strongest frequency component of a signal, in Hz, found via the number of zero-crossings. This is the overall standard deviation over all windows.
Derivative of Running Mean of Strongest Frequency Via Zero Crossings Overall Standard Deviation	Derivative of Running Mean of Strongest Frequency Via Zero Crossings. Running Mean of Strongest Frequency Via Zero Crossings. The strongest frequency component of a signal, in Hz, found via the number of zero-crossings. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Strongest Frequency Via Zero Crossings Overall Standard Deviation	Derivative of Standard Deviation of Strongest Frequency Via Zero Crossings. Standard Deviation of Strongest Frequency Via Zero Crossings. The strongest frequency component of a signal, in Hz, found via the number of zero-crossings. This is the overall standard deviation over all windows.
Strongest Frequency Via Spectral Centroid Overall Standard Deviation	The strongest frequency component of a signal, in Hz, found via the spectral centroid. This is the overall standard deviation over all windows.
Derivative of Strongest Frequency Via Spectral Centroid Overall Standard Deviation	Derivative of Strongest Frequency Via Spectral Centroid. The strongest frequency component of a signal, in Hz, found via the spectral centroid. This is the overall standard deviation over all windows.

Running Mean of Strongest Frequency Via Spectral Centroid Overall Standard Deviation	Running Mean of Strongest Frequency Via Spectral Centroid. The strongest frequency component of a signal, in Hz, found via the spectral centroid. This is the overall standard deviation over all windows.
Standard Deviation of Strongest Frequency Via Spectral Centroid Overall Standard Deviation	Standard Deviation of Strongest Frequency Via Spectral Centroid. The strongest frequency component of a signal, in Hz, found via the spectral centroid. This is the overall standard deviation over all windows.
Derivative of Running Mean of Strongest Frequency Via Spectral Centroid Overall Standard Deviation	Derivative of Running Mean of Strongest Frequency Via Spectral Centroid. Running Mean of Strongest Frequency Via Spectral Centroid. The strongest frequency component of a signal, in Hz, found via the spectral centroid. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Strongest Frequency Via Spectral Centroid Overall Standard Deviation	Derivative of Standard Deviation of Strongest Frequency Via Spectral Centroid. Standard Deviation of Strongest Frequency Via Spectral Centroid. The strongest frequency component of a signal, in Hz, found via the spectral centroid. This is the overall standard deviation over all windows.
Strongest Frequency Via FFT Maximum Overall Standard Deviation	The strongest frequency component of a signal, in Hz, found via finding the FFT bin with the highest power. This is the overall standard deviation over all windows.
Derivative of Strongest Frequency Via FFT Maximum Overall Standard Deviation	Derivative of Strongest Frequency Via FFT Maximum. The strongest frequency component of a signal, in Hz, found via finding the FFT bin with the highest power. This is the overall standard deviation over all windows.
Running Mean of Strongest Frequency Via FFT Maximum Overall Standard Deviation	Running Mean of Strongest Frequency Via FFT Maximum. The strongest frequency component of a signal, in Hz, found via finding the FFT bin with the highest power. This is the overall standard deviation over all windows.
Standard Deviation of Strongest Frequency Via FFT Maximum Overall Standard Deviation	Standard Deviation of Strongest Frequency Via FFT Maximum. The strongest frequency component of a signal, in Hz, found via finding the FFT bin with the highest power. This is the overall standard deviation over all windows.
Derivative of Running Mean of Strongest Frequency Via FFT Maximum Overall Standard Deviation	Derivative of Running Mean of Strongest Frequency Via FFT Maximum. Running Mean of Strongest Frequency Via FFT Maximum. The strongest frequency component of a signal, in Hz, found via finding the FFT bin with the highest

	power. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Strongest Frequency Via FFT Maximum Overall Standard Deviation	Derivative of Standard Deviation of Strongest Frequency Via FFT Maximum. Standard Deviation of Strongest Frequency Via FFT Maximum. The strongest frequency component of a signal, in Hz, found via finding the FFT bin with the highest power. This is the overall standard deviation over all windows.
MFCC Overall Standard Deviation	MFCC calculations based upon Orange Cow codeThis is the overall standard deviation over all windows.
Derivative of MFCC Overall Standard Deviation	Derivative of MFCC. MFCC calculations based upon Orange Cow codeThis is the overall standard deviation over all windows.
Running Mean of MFCC Overall Standard Deviation	Running Mean of MFCC. MFCC calculations based upon Orange Cow codeThis is the overall standard deviation over all windows.
Standard Deviation of MFCC Overall Standard Deviation	Standard Deviation of MFCC. MFCC calculations based upon Orange Cow codeThis is the overall standard deviation over all windows.
Derivative of Running Mean of MFCC Overall Standard Deviation	Derivative of Running Mean of MFCC. Running Mean of MFCC. MFCC calculations based upon Orange Cow codeThis is the overall standard deviation over all windows.
Derivative of Standard Deviation of MFCC Overall Standard Deviation	Derivative of Standard Deviation of MFCC. Standard Deviation of MFCC. MFCC calculations based upon Orange Cow codeThis is the overall standard deviation over all windows.
LPC Overall Standard Deviation	Linear Prediction Coeffecients calculated using autocorrelation and Levinson-Durbin recursion. This is the overall standard deviation over all windows.
Derivative of LPC Overall Standard Deviation	Derivative of LPC. Linear Prediction Coeffecients calculated using autocorrelation and Levinson-Durbin recursion. This is the overall standard deviation over all windows.
Running Mean of LPC Overall Standard Deviation	Running Mean of LPC. Linear Prediction Coeffecients calculated using autocorrelation and Levinson-Durbin recursion. This is the overall standard deviation over all windows.
Standard Deviation of LPC Overall Standard Deviation	Standard Deviation of LPC. Linear Prediction Coeffecients calculated using autocorrelation and Levinson-Durbin recursion. This is the overall standard deviation over all windows.
Derivative of Running Mean of LPC Overall Standard Deviation	Derivative of Running Mean of LPC. Running Mean of LPC. Linear Prediction Coeffecients

	calculated using autocorrelation and Levinson-Durbin recursion. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of LPC Overall Standard Deviation	Derivative of Standard Deviation of LPC. Standard Deviation of LPC. Linear Prediction Coeffecients calculated using autocorrelation and Levinson-Durbin recursion. This is the overall standard deviation over all windows.
Method of Moments Overall Standard Deviation	Statistical Method of Moments of the Magnitude Spectrum. This is the overall standard deviation over all windows.
Derivative of Method of Moments Overall Standard Deviation	Derivative of Method of Moments. Statistical Method of Moments of the Magnitude Spectrum. This is the overall standard deviation over all windows.
Running Mean of Method of Moments Overall Standard Deviation	Running Mean of Method of Moments. Statistical Method of Moments of the Magnitude Spectrum. This is the overall standard deviation over all windows.
Standard Deviation of Method of Moments Overall Standard Deviation	Standard Deviation of Method of Moments. Statistical Method of Moments of the Magnitude Spectrum. This is the overall standard deviation over all windows.
Derivative of Running Mean of Method of Moments Overall Standard Deviation	Derivative of Running Mean of Method of Moments. Running Mean of Method of Moments. Statistical Method of Moments of the Magnitude Spectrum. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Method of Moments Overall Standard Deviation	Derivative of Standard Deviation of Method of Moments. Standard Deviation of Method of Moments. Statistical Method of Moments of the Magnitude Spectrum. This is the overall standard deviation over all windows.
Partial Based Spectral Centroid Overall Standard Deviation	Spectral Centroid calculated based on the center of mass of partials instead of center of mass of bins. This is the overall standard deviation over all windows.
Derivative of Partial Based Spectral Centroid Overall Standard Deviation	Derivative of Partial Based Spectral Centroid. Spectral Centroid calculated based on the center of mass of partials instead of center of mass of bins. This is the overall standard deviation over all windows.
Running Mean of Partial Based Spectral Centroid Overall Standard Deviation	Running Mean of Partial Based Spectral Centroid. Spectral Centroid calculated based on the center of mass of partials instead of center of mass of bins. This is the overall standard deviation over all windows.

Standard Deviation of Partial Based Spectral Centroid Overall Standard Deviation	Standard Deviation of Partial Based Spectral Centroid. Spectral Centroid calculated based on the center of mass of partials instead of center of mass of bins. This is the overall standard deviation over all windows.
Derivative of Running Mean of Partial Based Spectral Centroid Overall Standard Deviation	Derivative of Running Mean of Partial Based Spectral Centroid. Running Mean of Partial Based Spectral Centroid. Spectral Centroid calculated based on the center of mass of partials instead of center of mass of bins. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Partial Based Spectral Centroid Overall Standard Deviation	Derivative of Standard Deviation of Partial Based Spectral Centroid. Standard Deviation of Partial Based Spectral Centroid. Spectral Centroid calculated based on the center of mass of partials instead of center of mass of bins. This is the overall standard deviation over all windows.
Partial Based Spectral Flux Overall Standard Deviation	Cacluate the correlation bettween adjacent frames based peaks instead of spectral bins. Peak tracking is primitive - whe the number of bins changes, the bottom bins are matched sequentially and the extra unmatched bins are ignored. This is the overall standard deviation over all windows.
Derivative of Partial Based Spectral Flux Overall Standard Deviation	Derivative of Partial Based Spectral Flux. Cacluate the correlation bettween adjacent frames based peaks instead of spectral bins. Peak tracking is primitive - whe the number of bins changes, the bottom bins are matched sequentially and the extra unmatched bins are ignored. This is the overall standard deviation over all windows.
Running Mean of Partial Based Spectral Flux Overall Standard Deviation	Running Mean of Partial Based Spectral Flux. Cacluate the correlation bettween adjacent frames based peaks instead of spectral bins. Peak tracking is primitive - whe the number of bins changes, the bottom bins are matched sequentially and the extra unmatched bins are ignored. This is the overall standard deviation over all windows.
Standard Deviation of Partial Based Spectral Flux Overall Standard Deviation	Standard Deviation of Partial Based Spectral Flux. Cacluate the correlation bettween adjacent frames based peaks instead of spectral bins. Peak tracking is primitive - whe the number of bins changes, the bottom bins are matched sequentially and the extra unmatched bins are ignored. This is the overall standard deviation over all windows.

Derivative of Running Mean of Partial Based Spectral Flux Overall Standard Deviation	Derivative of Running Mean of Partial Based Spectral Flux. Running Mean of Partial Based Spectral Flux. Cacluate the correlation bettween adjacent frames based peaks instead of spectral bins. Peak tracking is primitive - whe the number of bins changes, the bottom bins are matched sequentially and the extra unmatched bins are ignored. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Partial Based Spectral Flux Overall Standard Deviation	Derivative of Standard Deviation of Partial Based Spectral Flux. Standard Deviation of Partial Based Spectral Flux. Cacluate the correlation bettween adjacent frames based peaks instead of spectral bins. Peak tracking is primitive - whe the number of bins changes, the bottom bins are matched sequentially and the extra unmatched bins are ignored. This is the overall standard deviation over all windows.
Peak Based Spectral Smoothness Overall Standard Deviation	Peak Based Spectral Smoothness is calculated from partials, not frequency bins. It is implemented accortding to McAdams 99 McAdams, S. 1999. This is the overall standard deviation over all windows.
Derivative of Peak Based Spectral Smoothness Overall Standard Deviation	Derivative of Peak Based Spectral Smoothness. Peak Based Spectral Smoothness is calculated from partials, not frequency bins. It is implemented accortding to McAdams 99 McAdams, S. 1999. This is the overall standard deviation over all windows.
Running Mean of Peak Based Spectral Smoothness Overall Standard Deviation	Running Mean of Peak Based Spectral Smoothness. Peak Based Spectral Smoothness is calculated from partials, not frequency bins. It is implemented accortding to McAdams 99 McAdams, S. 1999. This is the overall standard deviation over all windows.
Standard Deviation of Peak Based Spectral Smoothness Overall Standard Deviation	Standard Deviation of Peak Based Spectral Smoothness. Peak Based Spectral Smoothness is calculated from partials, not frequency bins. It is implemented accortding to McAdams 99 McAdams, S. 1999. This is the overall standard deviation over all windows.
Derivative of Running Mean of Peak Based Spectral Smoothness Overall Standard Deviation	Derivative of Running Mean of Peak Based Spectral Smoothness. Running Mean of Peak Based Spectral Smoothness. Peak Based Spectral Smoothness is calculated from partials, not frequency bins. It is implemented accortding to

	McAdams 99 McAdams, S. 1999. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Peak Based Spectral Smoothness Overall Standard Deviation	Derivative of Standard Deviation of Peak Based Spectral Smoothness. Standard Deviation of Peak Based Spectral Smoothness. Peak Based Spectral Smoothness is calculated from partials, not frequency bins. It is implemented accortding to McAdams 99 McAdams, S. 1999. This is the overall standard deviation over all windows.
Relative Difference Function Overall Standard Deviation	log of the derivative of RMS. Used for onset detection. This is the overall standard deviation over all windows.
Derivative of Relative Difference Function Overall Standard Deviation	Derivative of Relative Difference Function. log of the derivative of RMS. Used for onset detection. This is the overall standard deviation over all windows.
Running Mean of Relative Difference Function Overall Standard Deviation	Running Mean of Relative Difference Function. log of the derivative of RMS. Used for onset detection. This is the overall standard deviation over all windows.
Standard Deviation of Relative Difference Function Overall Standard Deviation	Standard Deviation of Relative Difference Function. log of the derivative of RMS. Used for onset detection. This is the overall standard deviation over all windows.
Derivative of Running Mean of Relative Difference Function Overall Standard Deviation	Derivative of Running Mean of Relative Difference Function. Running Mean of Relative Difference Function. log of the derivative of RMS. Used for onset detection. This is the overall standard deviation over all windows.
Derivative of Standard Deviation of Relative Difference Function Overall Standard Deviation	Derivative of Standard Deviation of Relative Difference Function. Standard Deviation of Relative Difference Function. log of the derivative of RMS. Used for onset detection. This is the overall standard deviation over all windows.
Area Method of Moments Overall Standard Deviation	2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Area Method of Moments Overall Standard Deviation	Derivative of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Running Mean of Area Method of Moments Overall Standard Deviation	Running Mean of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Standard Deviation of Area Method of Moments Overall Standard Deviation	Standard Deviation of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Running Mean of Area Method of Moments Overall Standard Deviation	Derivative of Running Mean of Area Method of Moments. Running Mean of Area Method of

	Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Standard Deviation of Area Method of Moments Overall Standard Deviation	Derivative of Standard Deviation of Area Method of Moments. Standard Deviation of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Area Method of Moments of MFCCs Overall Standard Deviation	2D statistical method of moments of MFCCsThis is the overall standard deviation over all windows.
Derivative of Area Method of Moments Overall Standard Deviation	Derivative of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Running Mean of Area Method of Moments Overall Standard Deviation	Derivative of Running Mean of Area Method of Moments. Running Mean of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Standard Deviation of Area Method of Moments Overall Standard Deviation	Derivative of Standard Deviation of Area Method of Moments. Standard Deviation of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Area Method of Moments Overall Standard Deviation	Derivative of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Running Mean of Area Method of Moments Overall Standard Deviation	Derivative of Running Mean of Area Method of Moments. Running Mean of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Standard Deviation of Area Method of Moments Overall Standard Deviation	Derivative of Standard Deviation of Area Method of Moments. Standard Deviation of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Area Method of Moments Overall Standard Deviation	Derivative of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Running Mean of Area Method of Moments Overall Standard Deviation	Derivative of Running Mean of Area Method of Moments. Running Mean of Area Method of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Derivative of Standard Deviation of Area Method of Moments Overall Standard Deviation	Derivative of Standard Deviation of Area Method of Moments. Standard Deviation of Area Method

	of Moments. 2D statistical method of momentsThis is the overall standard deviation over all windows.
Area Method of Moments of Log of ConstantQ transform Overall Standard Deviation	2D statistical method of moments of the log of the ConstantQ transformThis is the overall standard deviation over all windows.
Area Method of Moments of ConstantQ-based MFCCs Overall Standard Deviation	2D statistical method of moments of ConstantQ-based MFCCsThis is the overall standard deviation over all windows.
Spectral Centroid Overall Average	The centre of mass of the power spectrum. This is the overall average over all windows.
Derivative of Spectral Centroid Overall Average	Derivative of Spectral Centroid. The centre of mass of the power spectrum. This is the overall average over all windows.
Running Mean of Spectral Centroid Overall Average	Running Mean of Spectral Centroid. The centre of mass of the power spectrum. This is the overall average over all windows.

Appendix D

Symphony Map from Azcarraga & Flores (2016)'s SOMphony

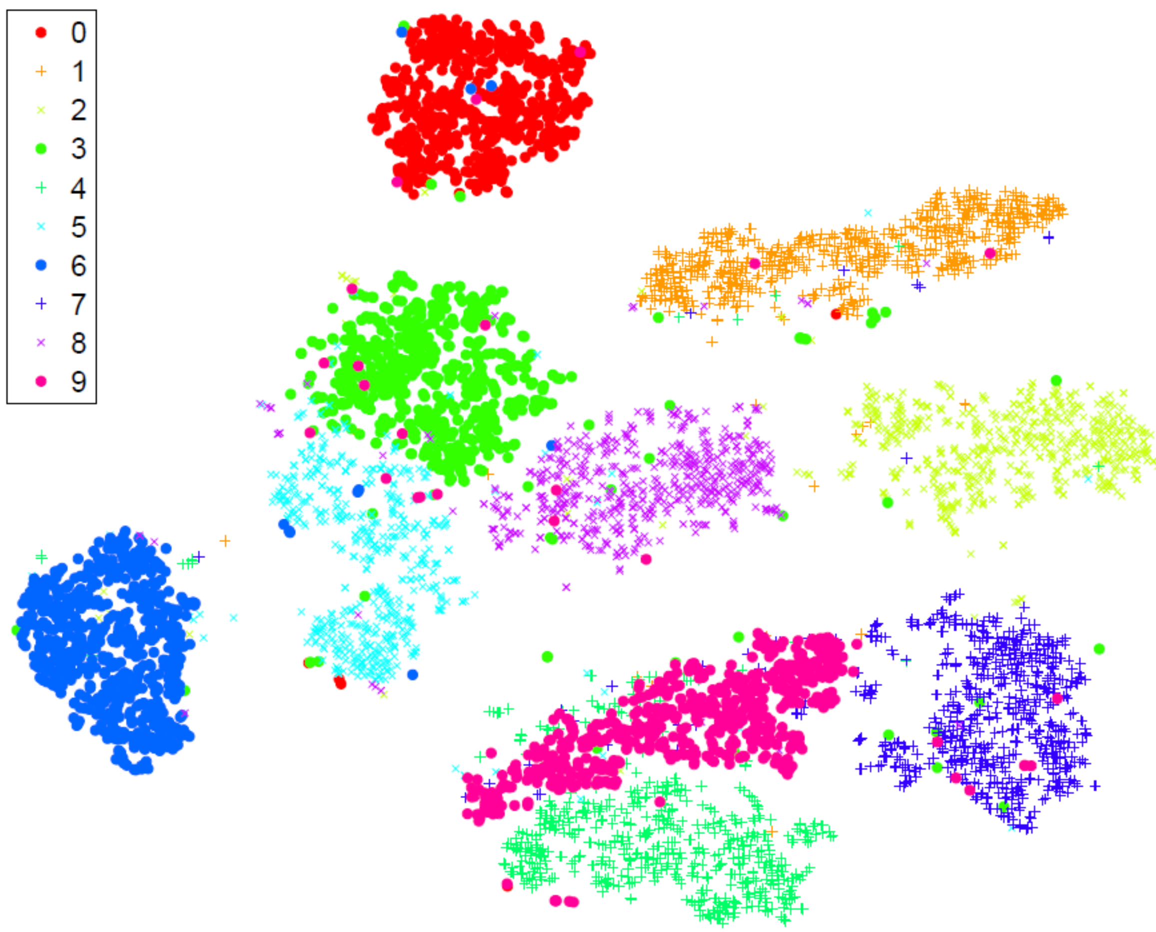
This section contains an image sample of a symphony map

Distance		Classical									19th Century									Romantic									20th Century								
		H1	H2	H3	M1	M2	M3	B1	B2	B3	B1	B2	B3	C1	C2	C2	G1	G2	G3	M1	M1	M3	SM1	SM2	SM3	SB1	SB2	SB3	R1	R2	R3	ST1	ST2	ST3	SH1	SH2	SH3
Classical	Haydn 1		14	7	25	25	30	30	76	31	26	27	31	25	27	30	35	29	27	26	28	28	24	22	22	29	33	31	32	34	32	46	48	65	32	34	33
	Haydn 2			15	29	29	32	37	76	36	29	31	34	30	34	38	36	34	31	27	32	32	30	30	29	34	35	36	33	39	38	47	50	65	34	32	33
	Haydn 3				25	24	28	28	75	30	28	30	31	23	24	28	36	29	29	25	27	27	27	22	24	28	32	30	31	33	29	45	47	64	30	32	32
	Mozart 1					15	27	30	79	41	32	36	35	18	31	30	41	23	25	31	27	25	32	18	28	17	33	24	31	27	23	48	51	66	31	34	34
	Mozart 2						16	32	70	42	22	28	23	14	27	29	33	16	15	23	19	18	23	12	20	10	23	11	20	15	21	36	39	55	21	26	25
	Mozart 3							38	58	43	21	27	19	20	29	31	24	20	16	15	12	11	26	24	27	20	14	15	15	19	22	26	28	45	10	16	15
	Bach 1								84	25	44	47	45	30	32	33	53	34	39	42	38	35	40	29	37	31	43	34	42	36	34	55	56	72	40	44	44
	Bach 2									82	58	57	52	74	77	78	57	74	67	57	63	63	67	74	69	75	50	65	53	63	74	38	41	39	52	50	47
	Bach 3										45	47	48	42	41	43	53	43	44	45	44	41	46	42	45	44	47	46	46	48	46	56	57	70	46	48	46
19th Century	Beethoven 1											10	8	29	35	38	25	29	21	19	25	26	16	25	20	29	19	23	16	24	36	25	29	46	21	23	20
	Beethoven 2												12	34	39	42	27	35	27	22	30	31	16	28	18	35	23	29	20	29	41	28	33	49	26	27	24
	Beethoven 3													29	36	38	25	30	21	18	24	25	18	27	21	30	14	22	11	22	36	19	24	41	18	20	16
	Clementi 1														16	16	35	18	19	23	17	16	27	16	25	13	25	17	25	22	13	41	42	60	23	29	28
	Clementi 2															9	41	30	30	28	26	25	33	26	31	28	32	29	33	33	22	47	44	63	30	36	35
	Clementi 3																42	29	30	31	27	25	36	28	35	27	33	30	35	34	18	48	46	66	32	39	37
	Gossec 1																	30	24	21	21	23	29	36	32	36	24	33	28	35	35	33	36	55	26	28	26
	Gossec 2																		11	27	16	16	26	18	27	12	28	18	28	22	20	41	43	61	27	33	32
	Gossec 3																			21	15	15	21	20	24	17	21	17	20	21	24	33	37	55	22	27	25
Romantic	Mendelssohn 1																				15	17	23	27	25	28	13	24	16	27	28	28	29	48	15	15	14
	Mendelssohn 2																					7	25	23	26	19	18	19	21	23	17	34	35	55	17	22	21
	Mendelssohn 3																						27	22	28	18	17	18	21	22	15	33	35	54	16	23	21
	Schumann 1																							19	10	27	26	23	22	24	35	35	39	57	28	31	29
	Schumann 2																								14	13	29	16	26	17	24	42	45	63	27	32	31
	Schumann 3																									25	28	23	24	23	33	39	42	60	28	31	30
	Schubert 1																										28	13	26	16	17	42	44	62	25	31	31
	Schubert 2																											20	9	22	28	19	22	41	11	14	9
	Schubert 3																												18	6	22	31	34	51	17	24	23

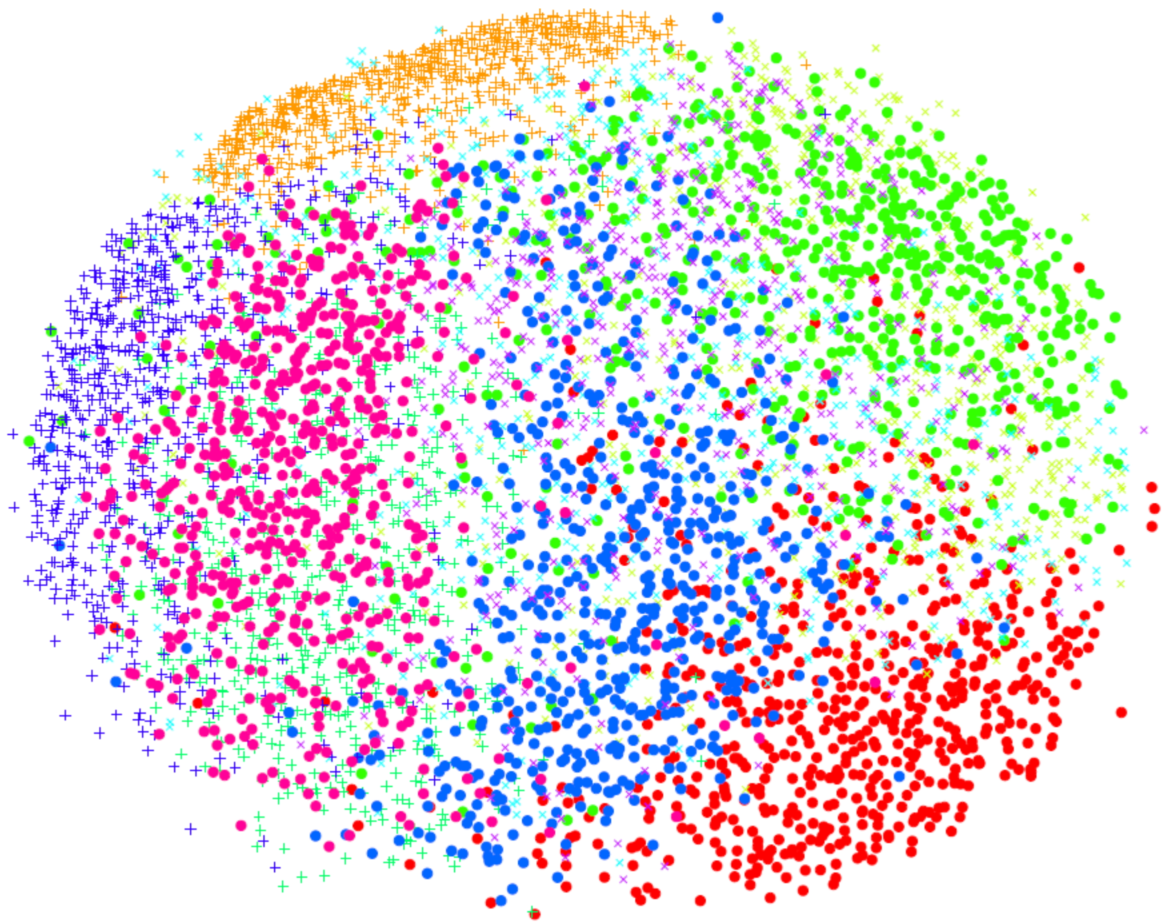
Appendix E

t-SNE vs Other Visualizations on MNIST

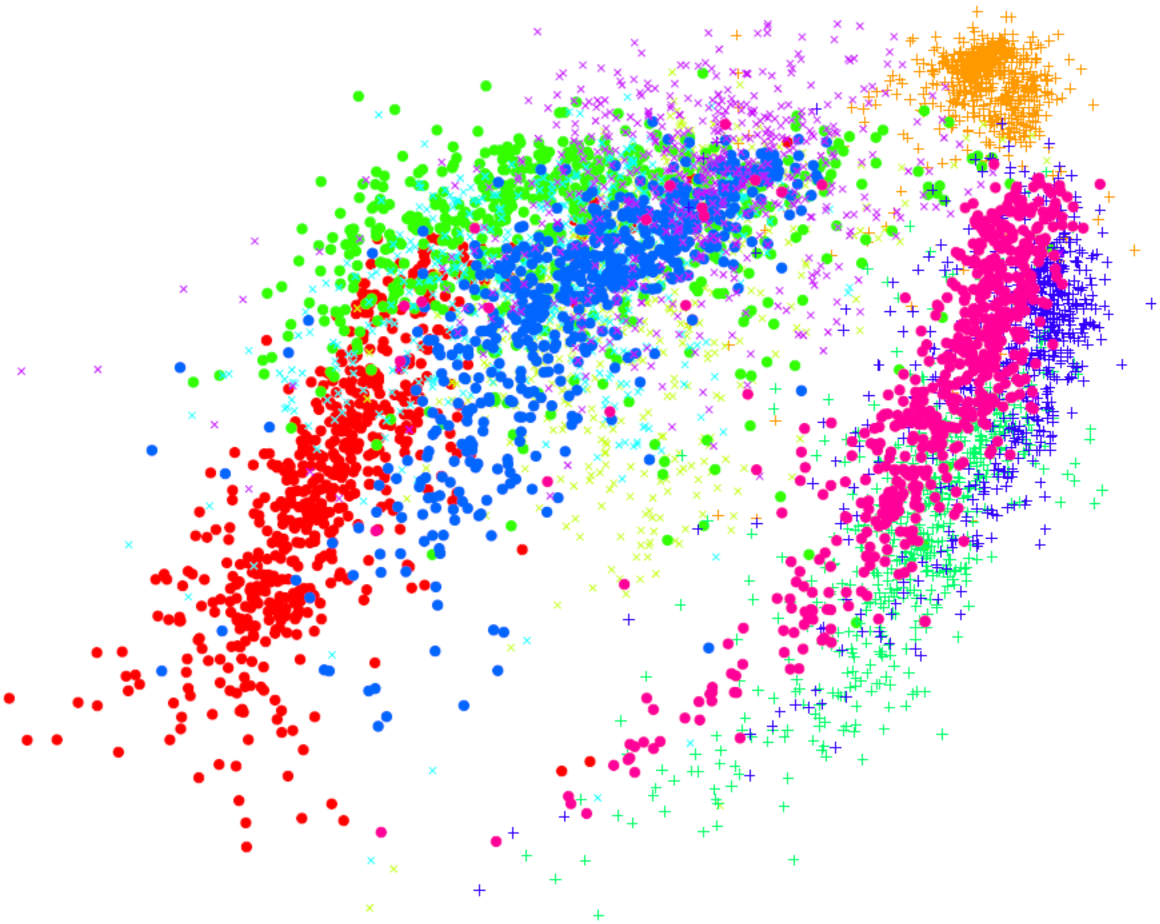
This section contains image samples of t-SNE Visualization vs Other Techniques on the MNIST Data Set from Maaten & Hinton (2008)'s Visualizing Data Using t-SNE



(a) Visualization by t-SNE.



(b) Visualization by Sammon mapping.



(a) Visualization by Isomap.

Appendix F

Resource persons

Mr. Fritz Kevin S. Flores

Adviser

Computer Technology Department

College of Computer Studies

De La Salle University Manila

fritz.flores@dlsu.edu.ph

Dr. Arnulfo P. Azcarraga

Panelist

Computer Technology Department

College of Computer Studies

De La Salle University Manila

arnie.azcarraga@delasalle.ph

Dr. Judith J. Azcarraga

Panelist

Computer Technology Department

College of Computer Studies

De La Salle University Manila

judith.azcarraga@dlsu.edu.ph

Appendix G

Personal Vitae

Mr. Jefferson Dionisio

+639322215642

jefferson_dionisio@dlsu.edu.ph

Ms. Naomi Portales

+639266796391

naomi_portales@dlsu.edu.ph

Mr. Kenji Fukuoka

+639291722475

kenji_fukuoka@dlsu.edu.ph

Mr. Edwardo Cruz

+639260092665

edwardo_cruz@dlsu.edu.ph

References

Agarwal, T. (2017). *A Brief Eplanation of Bandpass Filters.* Retrieved from https://www.efxkits.us/explanation-of-bandpass-filters-types-and-applications/

Azcarraga, A., Caronongan, A., Setiono, R., & Manalili, S. (2016). Validating the Stable Clustering of Songs in a Structured 3D SOM.

Azcarraga, A., & Flores, F. K. (2016). SOMphony: Visualizing Symphonies Using Self-Organizing Maps. In A. Hirose, S. Ozawa, K. Doya, K. Ikeda, M. Lee, & D. Liu (Eds.), *Neural Information Processing: 23rd International Conference, ICONIP 2016, Kyoto, Japan, October 16–21, 2016, Proceedings, Part IV* (pp. 531–537). Cham: Springer International Publishing.

BBC. (2014). *The Symphony.* Retrieved from http://www.bbc.co.uk/schools/gcsebitesize/music/western_tradition/mozart_symphony2.shtml

Brown, L. (2017). *Standard Youtube License vs Creative Commons.* Retrieved from https://filmora.wondershare.com/youtube-video-editing/standard-youtube-license-vs-cc.html

Brownlee, J. (2014). *An Introduction to Feature Selection.* Retrieved from https://machinelearningmastery.com/an-introduction-to-feature-selection/

Brownlee, J. (2016). *Supervised and Unsupervised Machine Learning Algorithms.* Retrieved from https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

Bullinaria, J. (2004). *Self-Organizing Maps: Fundamentals.* Retrieved from http://www.cs.bham.ac.uk/~jxb/NN/l16.pdf

Cambouropoulos, E., & Widmer, G. (2000). Automated Motivic Analysis via Melodic Clustering. *Journal of New Music Research*, *29*(4), 303.

Correa, D. C., & Rodrigues, F. A. (2016). A Survey on Symbolic Data-Based Music Genre Classification. *Expert Systems with Applications*, *60*, 190-210.

Dubnov, S., Assayag, G., Lartillot, O., & Bejerano, G. (2003, October). Using Machine-Learning Methods for Musical Style Modeling. *Computer*, *36*(10), 73–80. Retrieved from http://dx.doi.org/10.1109/MC.2003.1236474 doi: 10.1109/MC.2003.1236474

Foote, J. (1999). Visualizing Music and Audio Using Self-Similarity. *Proceedings of the Seventh ACM International Conference on Multimedia*(1), 77-80.

Germano, T. (1999). *Self-Organizing Maps.* Retrieved from http://davis.wpi.edu/~matt/courses/soms/

Grabczewski, K., & Jankowski, N. (2005). Feature Selection with Decision Tree Criterion. *Hybrid Intelligent Systems, 2005. HIS '05 Fifth International Conference*.

Gupta, P. (2017). *Decision Trees in Machine Learning.* Retrieved from https://medium.com/towards-data-science/decision-trees-in-machine-learning-641b9c4e8052

Guyon, I., & Elisseeff, A. (2006). An Introduction to Feature Extraction. *Guyon I., Nikravesh M., Gunn S., Zadeh L.A. (eds) Feature Extraction. Studies in Fuzziness and Soft Computing*, *207*.

Hartigan, J. A., & Wong, M. A. (1979). A K-Means Clustering Algorithm. *Journal of the Royal Statistical Society*, *28*(1), 100-108.

Heikkinen, D. (2017). *The Classical Symphony - Form and Structure.* Retrieved from http://professordeannaheikkinen.weebly.com/uploads/1/6/8/5/16856420/classical_music_form.pdf

Hepokoski, J., & Darcy, W. (2006). Elements of Sonata Theory : Norms, Types, and Deformations in the Late-Eighteenth-Century Sonata. *Oxford University Press*, 320.

Huron, D. (2001). *What is a Musical Feature? Fortes Analysis of Brahmss Opus 51, No. 1, Revisited.* Retrieved from http://www.mtosmt.org/issues/mto.01.7.4/mto.01.7.4.huron.html

IITG. (2011). *Cepstral Analysis of Speech.* Retrieved from iitg.vlab.co.in/?sub=59&brch=164&sim=615&cnt=1

Kohonen, T. (1995). Self-Organizing Maps. *Proceedings of the IEEE*, *78*, 1464-1480.

Kropotov, J. (2009). Methods: Neuronal Networks and Event-Related Potentials. *Quantitative EEG, Event-Related Potentials and Neurotherapy*.

Libin, L. (2014). *Symphony.* Retrieved from https://www.britannica.com/art/symphony-music

Loughran, R., Walker, J., O'Neill, M., & O' Farrell, M. (2008). The Use of Mel-Frequency Cepstral Coefficients in Musical Instrument Identification. *International Computer Music Association*, 1-4.

Lutter, M. (2014). *Mel-Frequency Cepstral Coefficients.* Retrieved from http://recognize-speech.com/feature-extraction/mfcc

Maaten, L., & Hinton, G. (2008). Visualizing Data Using t-SNE. *Journal of Machine Learning Research*, *19*, 2579-2605.

McCria, N. (2014). *An Introduction to Machine Learning Theory and Its Applications: A Visual Tutorial with Examples.* Retrieved from https://www.toptal.com/machine-learning/machine-learning

-theory-an-introductory-primer

McEnnis, D., & McKay, C. (2010). *JAudio 2.* Retrieved from http://jmir.sourceforge.net/jAudio.html

McEnnis, D., McKay, C., Fujinaga, I., & Depalle, P. (2005). JAudio: A Feature Extraction Library. *Queen Mary, University of London*, 600-602.

McFee, B., Barrington, L., & Lanckriet, G. R. G. (2012). Learning Content Similarity for Music Recommendation. *IEEE Transactions on Audio, Speech, and Language Processing*, *20*(8), 2207-2218.

Mermelstein, P. (1976). Distance Measures for Speech Recognition - Psychological and Instrumental. *Pattern Recognition and Artificial Intelligence*, 374-388.

Mitchell, T. M. (1997). Machine Learning. *McGraw-Hill Science, Engineering, Math.*

Ng, A. (2017). *Machine Learning.* Retrieved from https://www.coursera.org/learn/machine-learning

Ron, D., Singer, Y., & Tishby, N. (1996). The Power of Amnesia: Learning Probabilistic Automata with Variable Memory Length. *Machine Learning*, *25*, 117-149.

Rumelhart, D., & Zipser, D. (1985). Feature Discovery by Competitive Learning. *Cognitive Science*, *9*(1), 75-112.

Silla, C. N., & A., F. A. (2009). Novel Top-Down Approaches for Hierarchical Classification and Their Application to Automatic Music Genre Classification. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, San Antonio, Texas*, 3499-3504.

Storkey, A. (2017). *Machine Learning and Pattern Recognition Principal Component Analysis.* Retrieved from http://www.inf.ed.ac.uk/teaching/courses/mlpr/lectures/mlpr-dim-red.pdf

Tilden, I. (2013). *What Pop Music Owes to the Musical Masters.* Retrieved from https://www.theguardian.com/music/2013/jan/24/what-pop-music-owes-classical-masters

Trevino, A. (2016). *Introduction to K-Means Clustering.* Retrieved from https://www.datascience.com/blog/k-means-clustering

Yang, Y., & Pedersen, J. O. (1997). A Comparative Study on Feature Selection in Text Categorization. *ICML*, *97*, 412-420.

Ziv, J., & Lempel, A. (1978). Compression of Individual Sequences via Variable Rate Coding. *IEEE Trans. Information Theory*, *24*(5), 530-536.