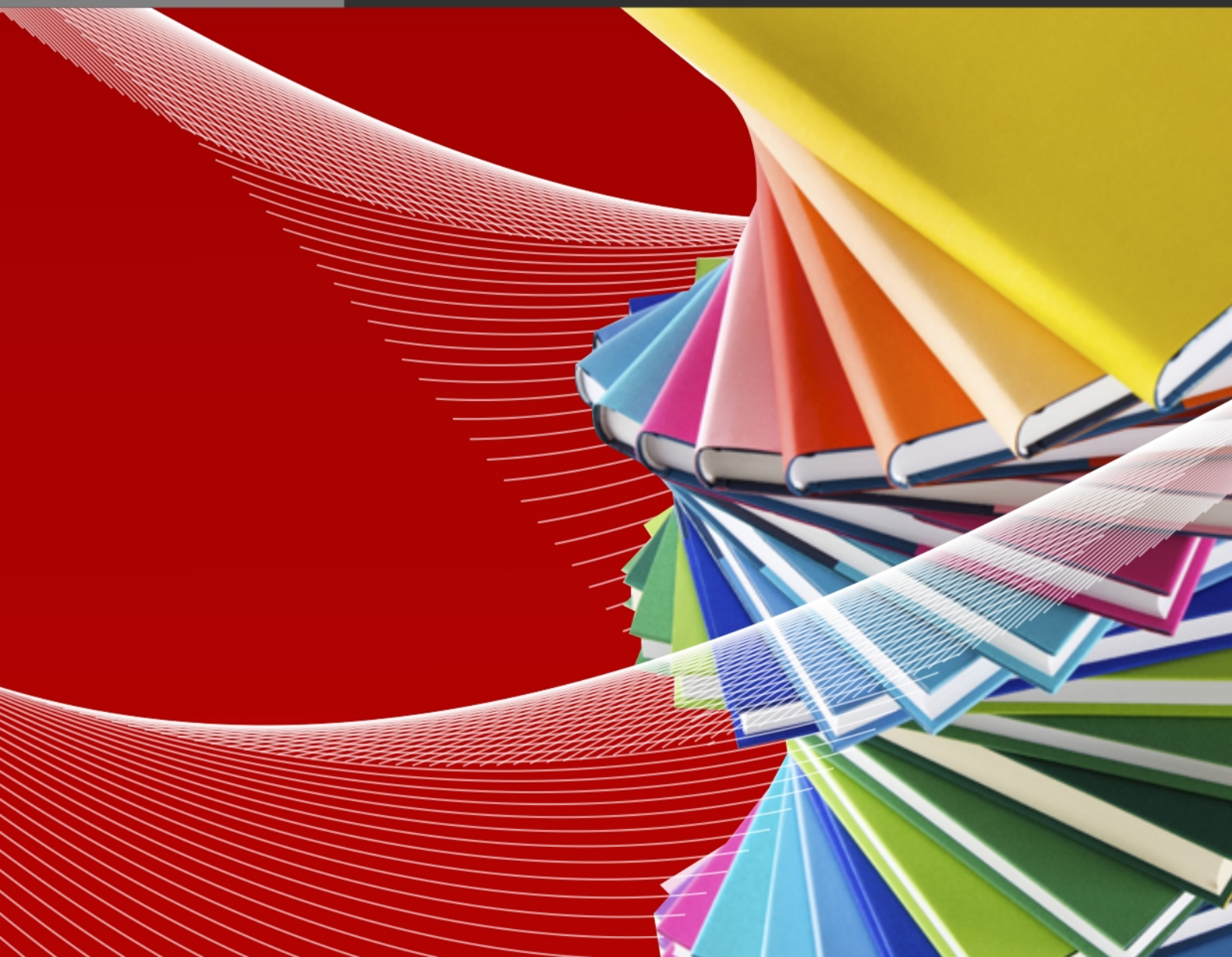
澳門大學
UNIVERSIDADE DE MACAU
UNIVERSITY OF MACAU

Outstanding Academic Papers by Students
學生優秀作品



University of Macau

Faculty of Science and Technology

澳門大學

UNIVERSIDADE DE MACAU

UNIVERSITY OF MACAU

User Customization for Music Emotion Classification using Online Sequential Extreme Learning Machine

by

WONG WAI KIN, Student No: D-B1-2758-9

Final Project Report submitted in partial fulfilment
of the requirements of the Degree of
Bachelor of Science in Computer Science

Project Supervisor

Dr. VONG CHI MAN

21 May 2015

DECLARATION

I sincerely declare that:

1. I and my teammates are the sole authors of this report,
2. All the information contained in this report is certain and correct to the best of my knowledge,
3. I declare that the thesis here submitted is original except for the source materials explicitly acknowledged and that this thesis or parts of this thesis have not been previously submitted for the same degree or for a different degree, and
4. I also acknowledge that I am aware of the Rules on Handling Student Academic Dishonesty and the Regulations of the Student Discipline of the University of Macau.

Signature : _____

Name : WONG WAI KIN

Student No. : D-B1-2758-9

Date : 21 May 2015

ACKNOWLEDGEMENTS

The author would like to express his utmost gratitude to UM for providing the opportunity to carry out a project as a partial fulfilment of the requirement for the degree of Bachelor of Science.

Throughout this project, the author was very fortunate to receive the guidance and encouragement from his supervisor…

ABSTRACT

Music is an art composed by sound. Music emotion recognition as a research topic stands on different areas such as psychology, musicology. The purpose of this work is to give a recommendation of music to the user by recognizing music emotion using machine learning algorithm. In order to take the music emotion recognition, a set of musical characteristics generated by MIR Tool Box has been used. Several machine learning algorithms are used and compared in this work. For traditional method such as k-nearest neighbour classifier (k-NN classifier) and state-of-the-art neural network such as support vector machine (SVM) and extreme learning machine (ELM). For the recognition result, it cannot get a full accuracy for every user. To improve the result, the online sequential extreme learning machine (OSELM) is used to learn one by one with a fixed size of new data for the user reported result then updating the model using the latest data.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. Introduction

1.1 Overview

Music is an art composed by sound. Composers use pieces of music to tell people their feelings. But users also have a feeling or emotion for a specific music. There may be some different emotion between composer and user somehow. But in most of the time, user just uses his/her own opinion to decide the mood or emotion. Also, we want to make a recommendation of a specific emotion area of music for the user. The recommendation can be helpful for meeting the situation of relaxing user.

1.2 Objectives

In this work, music emotion recognition is done by using serval machine learning algorithm such as k-nearest neighbour classifier (k-NN), support vector machine (SVM) and few of different version of extreme learning machine (ELM) such as ELM with kernel and multi-layer extreme learning machine (MLELM).

Before this approach, a pre-processing and feature extraction need to be done. The pre-processing approach is to change the file type of the music clips because Matlab only can read common file format of music (*.mp3, *.wav etc.). In the feature extraction section, it is going to extract the musical characteristics of all the music clips. Then, the emotion classification will select a combination of feature set to train the model using both of the machine learning algorithms introduced above.

The result can be predicted by after generating the model. If the user thinks that some classification of emotion for music may not be this class, we provide a method to get a small size and quick update for the model using online sequential extreme learning machine (OSELM). It can provide a more acceptable result for the user.

CHAPTER 2. Example Data and Labelling

It is a problem for us to develop an original music dataset for the experiment. It is not easy to collect the music clips and discover their emotion based on our resource and opinion. We need to define if there are copyrights or acceptable for the research use. Also, a reliable data set that has a correct emotion assigned is very important that it can directly affect the result of the experiment.

So, we have searched online and found a MIREX-likehood dataset for music information retrieval research [1]. The dataset offers a number of relevant original contributions:

903 music clips in 30 seconds
Clips with general music tagging such as artist, title, year, genre etc.
Clips with lyrics or MIDI format (partial)

2.1 Category Labelling

There are several defined in the MIREX-likehood dataset:

aggressive, amiable/good, autumnal, bittersweet, boisterous, brooding, campy, cheerful, confident, fiery, fun, humorous, intense, literate, natured, passionate, poignant, quirky, rollicking, rousing, rowdy, silly, sweet, tense/anxious, visceral volatile, whimsical, wistful, witty, wry.

Although there are many of emotions in named, they may have some similar characteristics between the others. The dataset also divide them into few clusters base on the five mood clusters proposed by MIREX [2]. The clustered categories and their emotions are listed in table 1 and the number and percentage of the full dataset (see table 2).

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Passionate	Rollicking	Literate	Humorous	Aggressive
Rousing	Cheerful	Poignant	Silly	Fiery
Confident	Fun	Wistful	Campy	Tense/Anxious
Boisterous	Sweet	Bittersweet	Quirky	Intense
Rowdy	Amiable/Good natured	Autumnal brooding	Whimsical Witty Wry	Volatile Visceral

Table 1: The emotions in different clusters

Cluster Group	Number of music clips	Percentage of dataset
Cluster 1	170	18.8%
Cluster 2	164	18.2%
Cluster 3	215	23.8%
Cluster 4	191	21.2%
Cluster 5	163	18.1%

Table 2: Number and percentage of music clips in different clusters

CHAPTER 3. Design and Implementation

There are five steps for the experiment procedure. They are dataset collection, data pre-processing, feature extraction, classification and improvement after classification. Here is the flow showing the experiment steps (see figure 1).

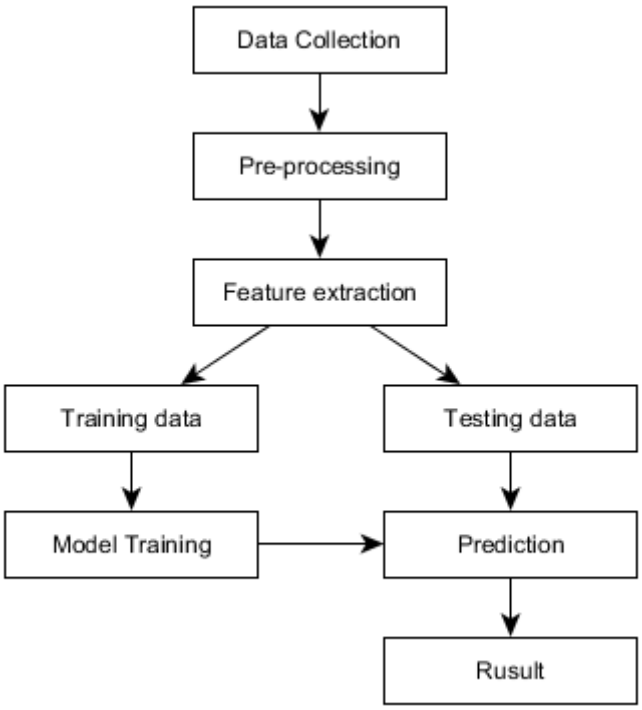


Figure 1: Procedure

3.1 Data Pre-processing

The dataset has converted to '*.mp3' method and the property of each music clips: the sample rate is 22050 Hz, in 16 bits precision and reduced channel to mono. We also normalized the volume of the dataset.

3.2 Feature extraction

We use a toolbox that is implemented in Matlab. It allows users to extract the musical features from some audio files [3]. The features such as rhythm, attacks time and tonality are extracted using different algorithms constructed by the toolbox. Total 189 features extracted by MIRToolbox and divide by 4 fields based on their musical characteristic. They are dynamic, rhythm, spectral and harmony field.

3.2.1 Dynamic Field

In dynamic field, the extracted features are RMS energy, slope, attack and low energy.

RMS energy is the frame-based root mean square energy of the music clips. It can be calculated by the formula,

$$x_{\text{rms}} = \sqrt{\frac{1}{n}(x_1^2 + x_2^2 + x_3^2 + \cdots + x_n^2)} \quad (1)$$

Where n is the number of the total frame of the music file.

Attack time is a part of ADSR (attack, decay, sustain level, release) model (see figure 2) [6]. Attack time is the time taken for initial run-up of level from nil to peak, beginn ing when the key is first pressed [5]. The feature slope and attack describe the slope of the attack and the time of attack.

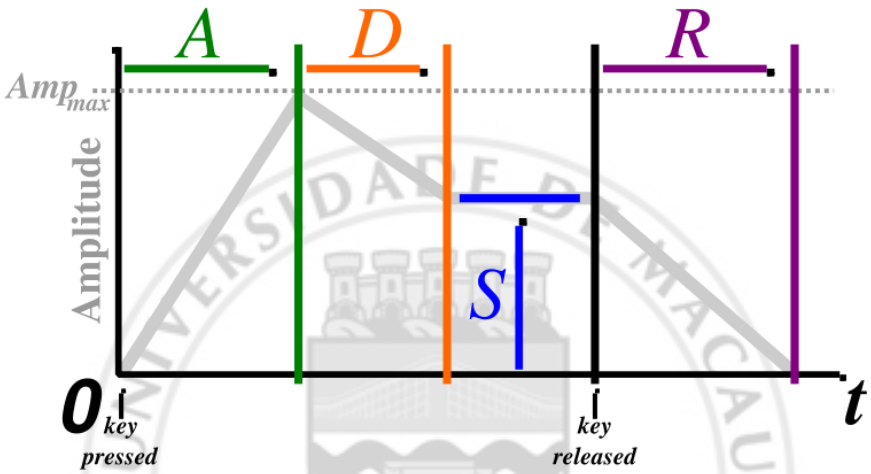


Figure 2: ADSR Table

Low energy is the RMS energy curve used for the low energy rate estimation; The ene rgy curve can be used to get an assessment of the temporal distribution of energy, in o rder to see if its remains constant throughout the signal, or if some frames are more co ntrastive than others. There is a way to estimate these consists by computing the low e nergy rate, for an example show the frames showing less-than-average energy percent age (Tzanetakis and Cook, 2002) [4].

3.2.2 Rhythm Field

In rhythm field, the extracted features are tempo and fluctuation peak.

Tempo is the speed or pace of the music. The feature is a frame-based tempo estimation.

Fluctuation peak is a summary with its highest peak, estimating the rhythmic is based on spectrogram computation transformed by auditory modelling and then spectrum estimation in each band [4].

3.2.3 Spectral Field

In spectral field, the features are extracted that:

The spectral centroid is a measure used in digital signal processing to characterise a spectrum. It indicates where the "centre of mass" of the spectrum is. Perceptually, it

has a robust connection with the impression of "brightness" of a sound [7]. It can be calculated by the equation,

$$C = \frac{\sum_{n=1}^{N} f(n)x_n}{\sum_{n=1}^{N} x_n} \quad (2)$$

Where x is the frame-based frequency, N is the number of total frame.

Here will be some high-frequency energy of musical feature.

- Brightness is the first central moment. A dual method consists in fixing this time the cut-off frequency, and measuring the amount of energy above that frequency (Juslin, 2000). The result is expressed as a number between 0 and 1. Figure 3 shows the range of cut-off energy [4].

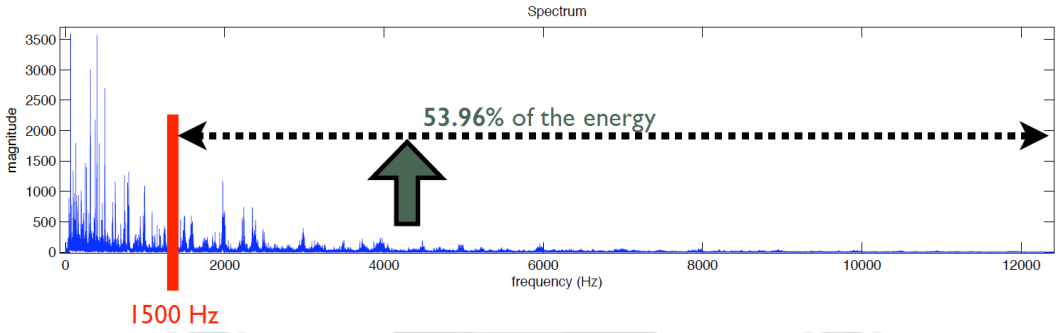


Figure 3: Brightness

- The second moment feature is spread. It is the frame-decomposed standard deviation (also can output variance) of the music clips.

- The third moment feature is called skewness. It is measuring the symmetry of the distribution. If the value is positive, it means more than half of values of the distribution are at the right. Negative if most of the value of distribution are at the left. When skewness is zero, it means there are equal value of the distribution at the left and right. The equation is for calculating skewness:

$$\mu_k = \int (x-\mu)^k f(x)dx \quad (3)$$

$$\gamma_1 = \frac{\mu_3}{\sigma^3} \quad (4)$$

- The forth moment feature kurtosis any measure of the "peakedness" of the probability distribution of a real-valued random variable [9]. The kurtosis is normally defined as

$$\gamma_2 = \frac{\mu_4}{\sigma^4} - 3 \quad (5)$$

- The rolloff 85 and rolloff 95 are the value to estimate the amount of high frequency in the signal consists in finding the frequency such that a certain fraction of the total energy is contained below that frequency. This ratio is fixed by default to 85% (following Tzanetakis and Cook, 2002), other have proposed 95% (Pohle, Pampalk and Widmer, 2005) (see figure 4) [4].

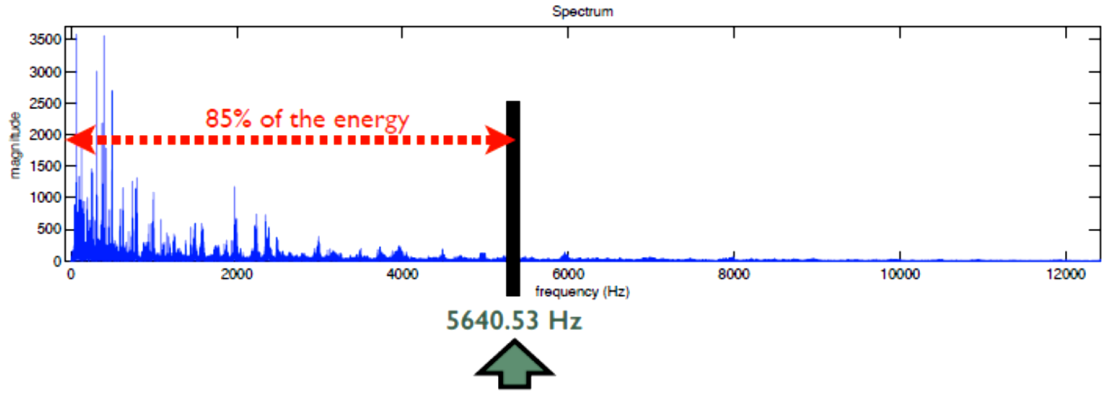


Figure 4: Roll off with threshold 85%

- The spectral entropy feature is using the relative Shannon (1948) entropy of the input to calculate. The Shannon entropy is calculated by the following equation,

$$\mathrm{H(X)} = -\sum_{i=1}^{n} p(x_i)\log_b p(x_i) \qquad (6)$$

- The flatness indicates whether the distribution is smooth or spiky, and results from the simple ratio between the geometric mean and the arithmetic mean [4], to calculate the spectral flatness:

$$\text{Flatness} = \frac{\sqrt[n]{\prod_{n=0}^{N-1} x(n)}}{\left(\frac{\sum_{n=0}^{\mathrm{N}-1} x(n)}{N}\right)} \qquad (7)$$

- Plomp and Levelt (1965) have proposed an estimation of the sensory dissonance or roughness, related to the beating phenomenon whenever pair of sinusoids is closed in frequency. They propose as a result an estimation of roughness depending on the frequency ratio of each pair of sinusoids represented as follows [4]:

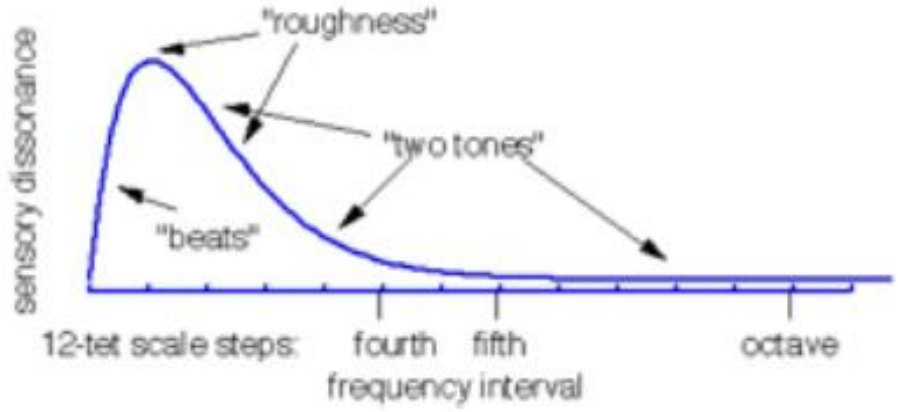


Figure 5: Roughness

- The irregularity of a spectrum is the degree of variation of the successive peaks of the spectrum.

- The inharmonicity feature is the amount of partials that are not multiples of the fundamental frequency, as a value between 0 and 1. Also, it considered here takes into account the amount of energy outside the ideal harmonic series [4].

- Zero-crossing rate (ZCR) is another basic acoustic feature that can be computed easily. It is equal to the number of zero-crossing of the waveform within a given frame. ZCR is often used in conjunction with energy (or volume) for end-point detection. In particular, ZCR is used for detecting the start and end positions of unvoiced sounds [8].

- Spectral flux is a measure of how quickly the power spectrum of a signal is changing, calculated by comparing the power spectrum for one frame against the power spectrum from the previous frame [10].

- MFCCs (Mel-frequency cepstral coefficients) offers a description of the cepstrum shape of the sound. Delta-MFCCs and delta-delta-MFCCs can also called differential and acceleration coefficients. We recall that the computation of the spectrum followed the following scheme [4]:

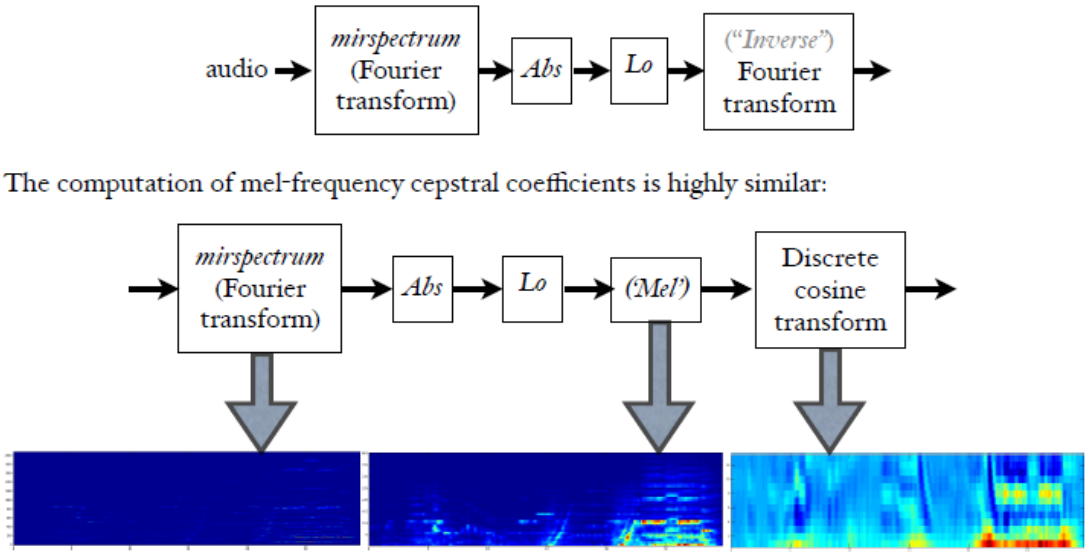


Figure 6: Step of computing MFCCs

Here the frequency bands are positioned logarithmically (on the Mel scale) which approximates the human auditory system's response more closely than the linearly-spaced frequency bands. And the Fourier Transform is replaced by a Discrete Cosine Transform. A discrete cosine transform (DCT) is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers. It has a strong "energy compaction" property: most of the signal information tends to be concentrated in a few low-frequency components of the DCT. That is why by default only the first 13 components are returned.

3.2.4 Harmony Field

There are five areas of features including in harmony field such as Chroma gram peak, Chroma gram centroid, key clarity and mode, and also HCDF.

- The Chroma gram, also called Harmonic Pitch Class Profile, shows the distribution of energy along the pitches or pitch classes.

  First the spectrum is computed in the logarithmic scale, with selection of, by default, the 20 highest dB, and restriction to a certain frequency range that covers an integer number of octaves, and normalization of the audio waveform before computation of the FFT. The Chroma gram is a redistribution of the spectrum energy along the different pitches (see figure 7).

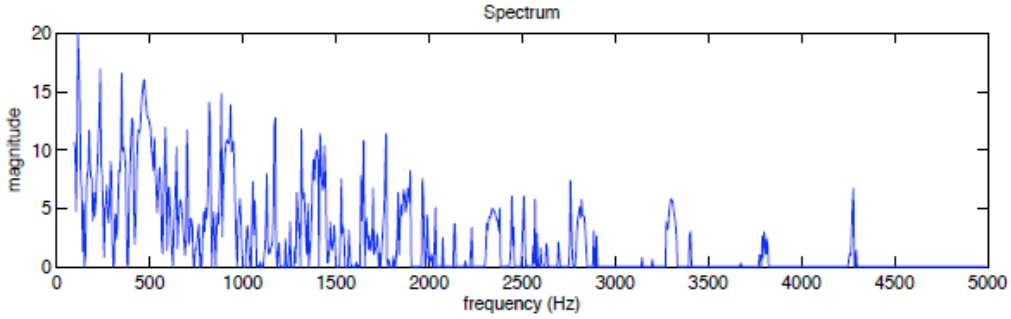


Figure 7: Chroma gram

- The key clarity and mode are the frame-decomposed clarity and mode of tonal. It estimates the modality, i.e. major vs. minor, returned as a numerical value between -1 and +1: the closer it is to +1, the more major the given excerpt is predicted to be, the closer the value is to -1, the more minor the excerpt might be [4].

- The Harmonic Change Detection Function (HCDF) is a function to detect changes in harmonic content. It maps 12-bin chroma vectors to the interior space of a 6-D polytope; pitch classes are mapped onto the vertices of this polytope. Close harmonic relations such as fifths and thirds appear as small Euclidian distances [11].

The extracted features and fields are shown in Table 3 (see table 3).

Field	Feature	Feature Types
Dynamic	RMS energy	Mean, stdiv, slope
	Slope	Mean, stdiv, slope
	Attack	Mean, stdiv
	Low energy	Mean
Rhythm	Tempo	Mean, stdiv, slope
	Fluctuation Peak	PosMean, MagMean
Spectral	Spectrum centroid	Mean, stdiv, slope

	Brightness	Mean, stdiv, slope
	Spread	Mean, stdiv, slope
	Skewness	Mean, stdiv, slope
	Kurtosis	Mean, stdiv, slope
	Rolloff 95	Mean, stdiv, slope
	Rolloff 85	Mean, stdiv, slope
	Spectral Entropy	Mean, stdiv, slope
	Flatness	Mean, stdiv, slope
	Roughness	Mean, stdiv, slope
	Irregularity	Mean, stdiv, slope
	Zero crossing rate	Mean, stdiv, slope
	Spectral flux	Mean, stdiv, slope
	MFCCs	Mean, stdiv, slope (12 entries for each)
	DMFCCs	Mean, stdiv, slope (12 entries for each)
	DDMFCCs	Mean, stdiv, slope (12 entries for each)
Harmony	Chroma gram peak	Mean, stdiv, slope
	Chroma gram centroid	Mean, stdiv, slope
	Key clarity	Mean, stdiv, slope
	Key mode	Mean, stdiv, slope
	HCDF	Mean, stdiv, slope

Table 3: Generated Features

3.3 Model Training and Classification

At this chapter, we will introduce the methods of model training that used in this project.

3.3.1 k-Nearest Neighbour Classifier

The k-nearest neighbour classifier (k-NN classifier) is a classical classifier for pattern recognition or data classification. It is a non-parametric method used for classification and regression [12].The method of the classifier is to make some data instances having closer feature vector distance for one group and then continuing to group the data using the previous grouping result. This classifier is often used to be a baseline method for comparing with other complicated method.

Steps of k-NN classifier: keep the number of points k_n = k constant and leave the volume to be varying,

$$\hat{p}(\underline{x}) = \frac{k}{NV(\underline{x})} \tag{8}$$

Choose k out of N training vectors, identify the k nearest ones to $\underline{x}$. Out of these k identify k_i that belong to class ω_i. And assign

$$\underline{x} \rightarrow \omega_i : k_i > k_j \; where \; \forall i \neq j \tag{9}$$

The data input order may affect the result of the result that the k-NN classifier generated because it is relying on the data already input to the model and calculate the instance between the existing data and new comers. Also, the chosen k needs to judge that it is suitable for the case of classifying. A concept of classification using k-NN classifier is shown in figure 8[21].

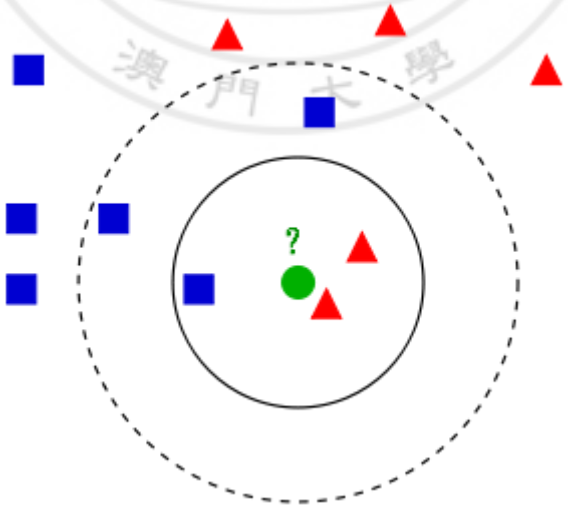


Figure 8: Example of k-nearest neighbour classifier

3.3.2 Support Vector Machine

Support vector machine (SVM) is a supervised learning model with associated learning algorithms [13]. The original SVM algorithm was invented by Vladimir N. Vapnik and Alexey Ya. C. in 1963. In 1992, Bernhard E. B., Isabelle M. G. and Vladimir N. V. suggested a way to create non-linear classifiers by applying the kernel trick to maximum-margin hyper planes [14].

The goal of the basic SVM is to make the data to be linearly separable classes, and the classifier

$$g(\underline{x}) = \underline{w}^T \underline{x} + w_0 = 0 \tag{10}$$

That leaves the maximum margin from both classes.

The distance of a point $\underline{\hat{x}}$ from a hyper plane is given by

$$Z_{\hat{x}} = \frac{g(\underline{\hat{x}})}{\|\underline{w}\|} \tag{11}$$

Scale $\underline{w}$, $\underline{w}_0$, so that at the nearest points, from each class, the discriminant function is ± 1:

$$|g(\underline{x})| = 1\{g(\underline{x}) = 1 \; for \; \omega_1 \; and \; g(\underline{x}) = -1 \; for \; \omega_2\} \tag{12}$$

Thus the margin is given by,

$$\frac{1}{\|\underline{w}\|} + \frac{1}{\|\underline{w}\|} = \frac{2}{\|w\|} \tag{13}$$

Also, the following is valid,

$$\underline{w}^T \underline{x} + w_0 \geq 1 \; (\forall \underline{x} \in \omega_1) \tag{14}$$

$$\underline{w}^T \underline{x} + w_0 \leq -1 \; (\forall \underline{x} \in \omega_2) \tag{15}$$

For the final goal of the support vector machine, it is to estimate a line for two groups that leaves a maximum margin of them. It can be helpful that it can provide the flexible for the model that has trained and separate the data for the largest distance when the new data is coming. Figure 9 shows the concept of separating two groups using SVM.

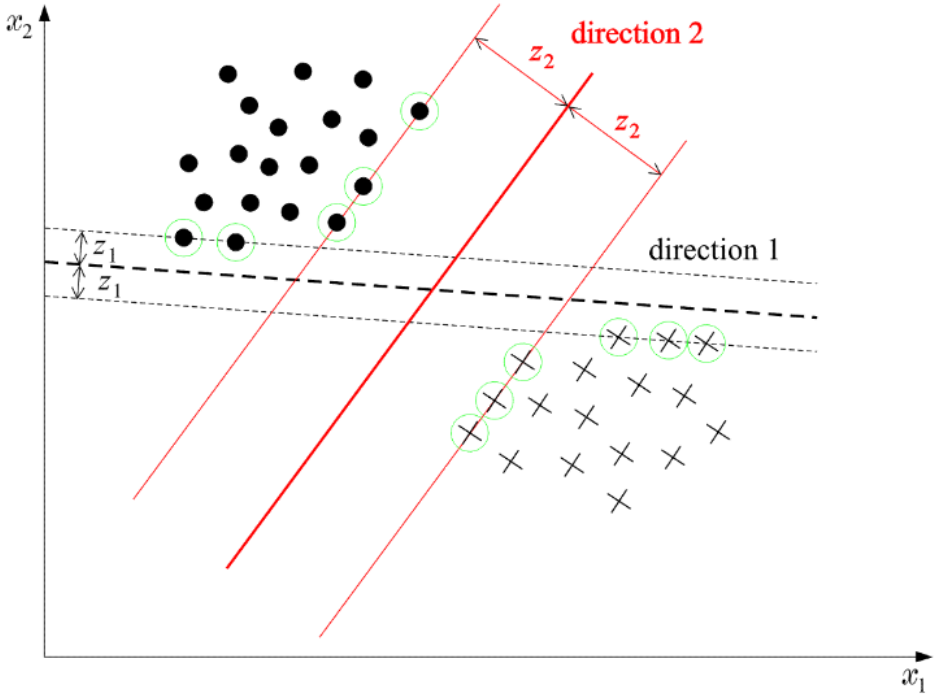


Figure 9: Example of support vector classification

For the non-linear classification of SVM, Bernhard E. B., Isabelle M. G. and Vladimir N. V. suggested a way that replacing every dot product by a non-linear kernel function to maximize the margin of hyper planes [16]. Some common kernels are listed in the table (see table 4) and non-linear training example uses SVM (see figure 10).

Radial basis Functions	$k(\underline{x},\underline{z}) = \exp\left(-\frac{\|\underline{x}-\underline{z}\|^2}{\sigma^2}\right)$
Polynomial (homogeneous)	$k(x_i, x_j) = (x_i \cdot x_j)^d, \mathrm{d} > 0$
Polynomial (inhomogeneous)	$k(x_i, x_j) = (x_i \cdot x_j + 1)^d, d > 0$
Gaussian radial basis	$k(x_i, x_j) = \exp\left(-\gamma\|x_i - x_j\|^2\right)$ for $\gamma > 0$ where $\gamma = \frac{1}{2\sigma^2}$
Hyperbolic tangent	$k(x_i, x_j) = \tanh(\kappa x_i \cdot x_j + c)$

Table 4: Kernel for non-linear SVM

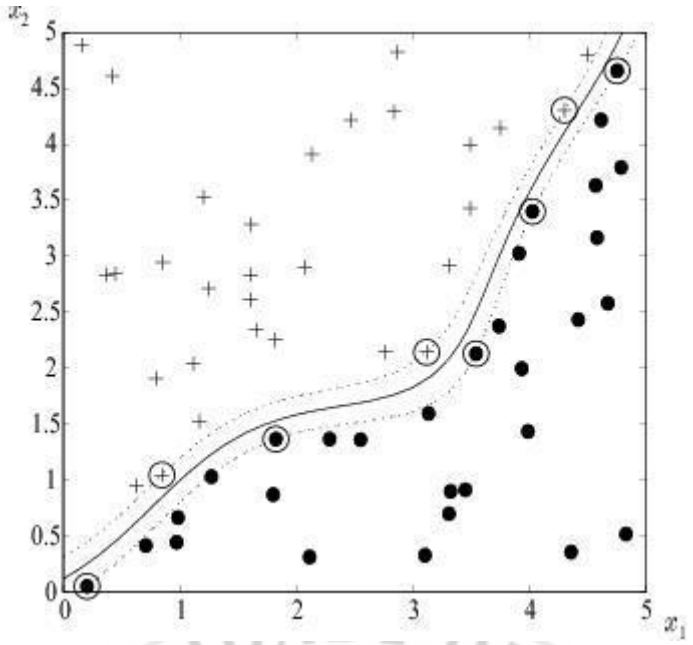


Figure 10: An example of non-linear classification using SVM

In this work, the classification by SVM is using an open source and implemented in different programming language called LIBSVM [15]. It is helpful that users do not need to do coding of the hard equations and functions for every experiments using SVM training.

3.3.3 Extreme Learning Machine

3.3.3.1 Basic ELM

Before knowing the ELM algorithm, first introduce the single hidden layer feed forward networks. Huang raise a method for solving the problem of a single hidden layer feed forward networks (SLFNs) with random hidden nodes [17].

For N arbitrary distinct samples (x_i, t_i), where $x_j = \left(x_{j1}, x_{j2}, \dots, x_{jn}\right)^T \in R^n$ $t_j = \left(t_{j1}, t_{j2}, \dots, t_{jn}\right)^T \in R^m$. A standard SLFNs with $\widehat{N}$ hidden nodes and activation function g(x) are defined as

$$\sum_{i=1}^{\widetilde{N}} \beta_i g\left(w_i \cdot x_j + b_i\right) = o_j, j = 1,2, \dots, N \qquad (16)$$

Where $w_i = (w_{i1}, w_{i2}, \dots, w_{in})^T \; and \; \beta_i = (\beta_{i1}, \beta_{i2}, \dots, \beta_{im})^T$. w_i is weight vector connecting the i-th hidden layer and input layer. β_i is weight vector connecting the i-th hidden layer and output layer.

The above SLFNs can approximate these N samples with zero errors means that

$$\sum_{i=1}^{\tilde{N}} \left\| o_j - t_j \right\| = 0 \tag{17}$$

That means, there exist W, β and b_i, such that

$$\sum_{i=1}^{\tilde{N}} \beta_i g\left(w_i \cdot x_j + b_i\right) = o \backslash t_j \,, j = 1,2, \dots, N \tag{18}$$

The above N equation can be written as

$$\mathrm{H}\beta = \mathrm{T} \tag{19}$$

Where

$$\mathrm{H}(\mathrm{w}_1, \dots, \mathrm{w}_{\tilde{N}}, b_1, \dots, b_{\tilde{N}}, x_1, \dots, x_n) = \begin{bmatrix} g(w_1 \cdot x_1 + b_1) & \cdots & g(w_{\tilde{N}} \cdot x_1 + b_{\tilde{N}}) \\ \vdots & \ddots & \vdots \\ g(w_1 \cdot x_N + b_1) & \cdots & g(w_{\tilde{N}} \cdot x_N + b_{\tilde{N}}) \end{bmatrix} \tag{20}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times m} \quad and \; T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m} \tag{21}$$

Where H is a matrix of hidden layer output of the neural network and i is the i-th hidden layer vector [18, 19].

When W and b are fixed, the special solution of $\mathrm{H}\beta = \mathrm{T}$ will be

$$\left\| H\hat{\beta} - T \right\| = \min_{\beta} \|H\beta - T\| \tag{22}$$

Where $\hat{\beta} = \mathrm{H}^{\dagger}\mathrm{T}$ is the least square solution of a general linear system $\mathrm{H}\beta = \mathrm{T}$ and where $\mathrm{H}^{\dagger}$ is the Moore-Penrose [17] generalized inverse of matrix H. The extreme learning machine is to find $\hat{\beta}$ for the answer.

In the project, we also using the algorithm implement in MATLAB and providing by [17].

3.3.3.2 Kernel based ELM

The kernel based extreme learning machine is using some kernel functions (see table 5) to replace the activation function of ELM to enhance the performance of the classification or regression. It do not need user to input the hidden neurons number because function is already control by the kernel function that user has defined to use.

Radial basis Functions	$k(\underline{x},\underline{z}) = \exp\left(-\frac{\left\|\underline{x}-\underline{z}\right\|^2}{2\sigma^2}\right)$
Linear	$k(x,y) = x^T y + c$
Polynomial	$k(x,y) = (\alpha x^T y + c)^d$
Wave	$k(x,y) = \frac{\theta}{\|x-y\|}\sin\frac{\|x-y\|}{\theta}$

Table 5: kernel function using in ELM-kernel MATLAB function

3.3.3.3 Multi-Layer ELM

Multi-layer extreme learning machine (MLELM) is developed by two parts: first is the extreme learning machine auto encoder (ELM-AE) and second one is the learnt features by using extreme learning machine classification [24]. The ELM-AM provides the initial hidden neurons weight for MLELM and do not need parameter settings. For the situation that the i-th hidden layer neurons number is equal to the i+1-th hidden neurons number, the feature mapping function will be chosen to be a linear function. If the number is not equal, the feature mapping function will be chosen to be a non-linear function such as sigmoidal function

$$\mathrm{H}^{\mathrm{k}} = g((\beta^k)^T H^{k-1}) \qquad (23)$$

where H^{k} is the output matrix of k-th hidden layer. The t is the output connections of the last hidden layer and the output node. The output is calculated by least square solution.

3.3.4 Online Sequential Extreme Learning Machine

The Online Sequential Extreme Learning Machine (OS-ELM) divided two parts for the learning of the hidden layers of output weight vector of SLFNs. The first part is the initial state, using few samples to find the β that is a previous single hidden layer's output weight. The second part is the online learning, using trunk-by-trunk or one-by-one (a special case) of data to update the β that already trained at the first part and get a updated result with the model [20].

Steps of the OS-ELM algorithm:

1. Randomly initialize the input weight $\mathrm{a_i}$ and the input bias $\mathrm{b_i}, \mathrm{i} = 1,2,\ldots,\widetilde{\mathrm{N}}$
2. Calculate the hidden layer output weight $\mathrm{H_0}$, where $\mathrm{H_0}$ is

$$\mathrm{H}_0 = \begin{bmatrix} g(a_1, b_1, X_1) & \cdots & g(a_{\tilde{N}}, b_{\tilde{N}}, X_1) \\ \vdots & \ddots & \vdots \\ g(a_1, b_1, X_{N_0}) & \cdots & g(a_{\tilde{N}}, b_{\tilde{N}}, X_{N_0}) \end{bmatrix} \tag{24}$$

3. Calculate the output weight $\beta^{(0)} = P_0 H_0^T T_0$, where

$$\mathrm{P}_0 = (H_0^T H_0)^{-1} \; and \; T_0 = \left[t_1, t_2, \ldots, t_{N_0}\right]^T \tag{25}$$

4. Assume the number of a trunk of sample is N_i, then calculate the hidden layer output weight H_i, where H_i is shown as the equation (23) for every trunk of samples (may be trunk-by-trunk or one-by-one).

5. Calculate the output weight $\beta^{(0)} = P_0 H_0^T T_0$, if there are multiple sample,

$$\begin{aligned} \mathrm{P}_{k+1} &= P_k - P_k H_{k+1}^T (I + H_{k+1} P_k H_{k+1}^T)^{-1} H_{k+1} P_k \\ \beta^{(k+1)} &= \beta^{(k)} + P_{k+1} H_{k+1}^T \left(T_{k+1} - H_{k+1} \beta^{(k)}\right) \end{aligned} \tag{26}$$

If there is only one sample,

$$\begin{aligned} \mathrm{P}_{k+1} &= P_k - \frac{P_k h_{k+1} h_{k+1}^T P_k}{1 + h_{k+1}^T P_k h_{k+1}} \\ \beta^{(k+1)} &= \beta^{(k)} + P_{k+1} h_{k+1} \left(t_{k+1}^T - h_{k+1}^T \beta^{(k)}\right) \end{aligned} \tag{27}$$

6. Finally, return β which is the weight of the model.

The figure (see figure 11) is the continue process of the prediction. That is using the OS-ELM training method for the modified data.

⋮

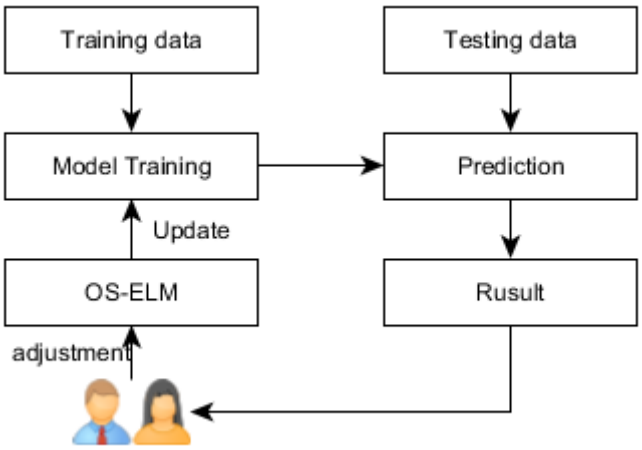


Figure 11: Process of OS-ELM to update the result

CHAPTER 4. Experiment Result

4.1 Experiment 1

In this experiment, we test the mean, standard deviation and slope of every features have generated and providing those fields. All of the training method such as k-NN classifier, SVM, basic ELM, kernel ELM and MLELM are used in the part. The result is shown by table 6.

Where the neighbour number k is set to 5 for k-NN classifier and SVM is using the polynomial kernel and ELM is using 50 for the parameter of hidden neurons number and kernel ELM is using the RBF kernel and MLELM is using the parameters (`3, [150, 300], [1e-1,1e4,1e8],0.05, [0.8,0.9], [0.8,0.9]`) for training these methods.

Method	Mean feature set	Std feature set	Slope feature set
k-NN classifier	28.57%	27.59%	23.15%
SVM	35%	41.87%	24.63%
ELM	34.52%	33.49%	29.06%
Kernel ELM	35.45%	32.51%	31.03%
MLEM	36.42%	36.94%	36.94%

Table 6: Result of experiment 1

4.2 Experiment 2

In this experiment, we test the effect of different musical fields of the separated dataset. The 4 feature sets, dynamics, rhythm, spectral, harmony, taking with different method (SVM, kernel ELM and MLELM) to predict the result. The result is shown by table 7.

Where SVM is using the polynomial kernel and kernel ELM is using the RBF kernel and MLELM is using the parameters (`3, [150, 300], [1e-1,1e4,1e8],0.05, [0.8,0.9], [0.8,0.9]`) for training these methods.

	SVM	Kernel ELM	MLELM
Dynamic set	25.61%	34.48%	38.42%
Rhythm set	31.18%	31.52%	39.9%

Spectral set	40.88%	41.87%	38.42%
Harmony set	36.45%	38.42%	41.37%

Table 7: Result of experiment 2

4.3 Experiment 3

In this experiment, we also use those 4 feature sets in experiment 2. But now is to take a combination of those feature set for all possible combination. SVM, kernel ELM and MELM will also use in this part to predict the result. The result is shown by table 8.

Where SVM is using the polynomial kernel and kernel ELM is using the RBF kernel and MLELM is using the parameters (`3, [150, 300], [1e-1,1e4,1e8],0.05, [0.8,0.9], [0.8,0.9]`) for training these methods.

Feature set	SVM	Kernel ELM	MLELM	KMLELM
Dynamic + Rhythm	30.04%	37.93%	30.54%	31.03%
Dynamic + Spectral	41.38%	43.34%	37.93%	42.86%
Dynamic + Harmony	38.42%	39.4%	34.48%	33.50%
Rhythm + Spectral	40.39%	43.84%	38.92%	42.36%
Rhythm + Harmony	33.99%	39.9%	37.44%	36.95%
Spectral + Harmony	42.85%	46.3%	41.87%	44.83%
Dynamic + Rhythm + Spectral	40.39%	43.84%	43.84%	44.83%
Dynamic + Rhythm + Harmony	35.96%	38.91%	39.90%	36.95%
Dynamic + Spectral + Harmony	44.33%	45.81%	43.35%	45.32%
Rhythm + Spectral + Harmony	43.84%	45.32%	43.84%	46.80%
All sets	45.32%	43.84%	45.32%	46.80%

Table 8: Result of experiment 3

4.4 Experiment 4

After we have found the best classifier, the model will be used for user customization. User may make some changes for the predicted result. In this experiment, we define a percentage of the data that the data have been modified by the user and see that how can it affects the result after the model re-trained. Also, the data label of data modification is randomly changed. The result is shown by table 9.

	OS-ELM	
Data Modified (%)	Training Time	Accuracy
10	0s	36.10%
20	0s	35.44%
30	0.0156s	36.77%
40	0.0156s	35.77%

Table 9: Result of experiment 4

CHAPTER 5. Evaluation

In experiment one, we test the feature sets between the mean, standard deviation and slope of the all type of the features we have generated. Refer to the table 6, we found that the k-NN classifier is not very efficient to deal with all the sets of features. It is hard to separate the high dimensional data. Also, we can see that all of the classifiers do not get a satisfied result. All of them are below 40%, some fields get less than 30%. The mean feature set may be the best performance set in the three sets. Other two sets are not stable (some classifiers get a high accuracy but also some classifiers get a low accuracy). The conclusion of this experiment is, it seems that no any relationship between those different statistical types of features.

In experiment two, we test the separated feature sets dividing by using the musical characteristics. We choose some better performance classifiers the continuing experiments such as SVM, kernel ELM and MLELM. There are four sets after the division. They are dynamic field, rhythm field, spectral field and harmony field. Refer to the table 7, we found that the results predicted by different classifier are similar. In different fields, the dynamic and rhythm field are the first and second lowest accuracy of the result. A reason for that may be there are so few features in the set (dynamic: around 10 and rhythm: 5) (Refer to the table 3).

In experiment three, we continue to test the difference between the four fields feature sets. We have tested the separated feature sets using combinations to get a new feature sets containing the four small sets. Refer the table 8, we examine again that the useful of the four sets. There may meet the worst result when only using the combination of dynamic and rhythm field. And the most useful set is the spectral field.

In experiment four, we test the performance of the OS-ELM that compared different percentage of data modification. Refer to table 9, the result does not have big changes for small or big data changing. A reason of that may be the randomly changing label.

CHAPTER 6. Discussion

Although the MIREX like hood dataset containing 903 songs in the dataset, the size of the sample may not be enough large to generate an accrue result of classification. Also, it is very hard to build an own music emotion recognition dataset with large samples. The construction is very expensive and time-consuming. For the accuracy, another factor may be the features of music. The features are not very powerful to recognise emotions of the music. It is because the emotion of music is a complex definition. The emotion may have changes between different people. So, there is another method to define the emotion of music that is using regression model [25]. That can reduce the misunderstanding between the definitions of music emotions. In the music recommendation, user may likely request the update of emotion for the song once on each time. So, the chunk size of OS-ELM for updating model is sat to 1 in this project. The desired result is by running OSELM to hundreds times. In the result between MLELM and KMLELM, the performance of the latter is better than the former. The MLELM has several parameters need to manually tune for each layer and KMLELM does not. Because it is using the kernel function to replace the usage of them. The drawback of KMLEM is, it need to have a larger memory (MLELM also need large memory) than the MLELM for processing the kernel functions.

CHAPTER 7. Ethics and Professional Conduct

7.1 Give proper credit for intellectual property

It is one of the ethical considerations. The main purpose of it is to protect the integrity of intellectual property. In the project and report, the source codes in MATLAB are open source and free of charge. The only work we need to do is to cite the work done by other people or research. There may have a problem that if those materials do not have a full citation in their works or some codes may have different source or need to cite multi titles.

7.2 Acquire and maintain professional competence

It is one of the professional conducts. The imperatives require programmers to participate in setting standards for appropriate levels of competence, and strive to achieve those standards. In this project, the result is not good as we expected. But we are continuing to find some solution for the improvement of the project. For example, search for some new algorithm or improved algorithms; find another definition of feature model for the classification; larger the dataset to enhance the learning rate.

CHAPTER 8. Conclusions

The main objective of this project is to giving recommendation according to the music emotions. It can mainly divide by two parts. First, we make the classification of music emotions using several methods. They are k-NN classifier, SVM, ELM (basic and kernel based), MLELM (basic and kernel based). The dataset we have used is MIREX like hood dataset that containing 903 songs and separated to 5 clusters of emotions. Using the dataset with several separated feature sets to do the prediction and get the best result. The performance of k-NN classifier is poor. The result of SVM and ELM is very similar and in a small range of difference but ELM can run faster comparing with them. After the prediction of the emotion, we will give the recommendation for the user which emotion he/she wants to listening to. There also is an improvement section for the user to adjust the prediction result. We use the OSELM to get a trunk-by-trunk update for the model. Although we have given the recommendation for the user, the result is not able to say that it is a reliable suggestion. Due to the special of the music emotion, it is unstable to classify a large of set of emotion. So, the future work seems to use a regression type of emotion recognition – the arousal-valance emotion plane [24]. But this approach needs some psychological knowledge and several high level features intuitively related to emotion representation. So, it is harder for generating the dataset with that information. It is more challenging on the data collection and feature generation.

CHAPTER 9. Appendix

9.1 Hardware Requirements

If you need to run the multi-layer ELM algorithm, depending on the size of feature sets (both feature dimensions and number of the data), at least 16GB RAM is recommended.

9.2 Software Requirements

For running all the experiments in this project, MATLAB is required and the version R2010a or later of MATLAB is recommended.

MIR Toolbox is available on the website [3]. For running the toolbox, Signal Processing Toolbox, one of the optional sub-packages of MATLAB, need to be properly installed. For installing extra plugins or toolbox, MATLAB needs user to manually set the toolbox path in the setting (see figure

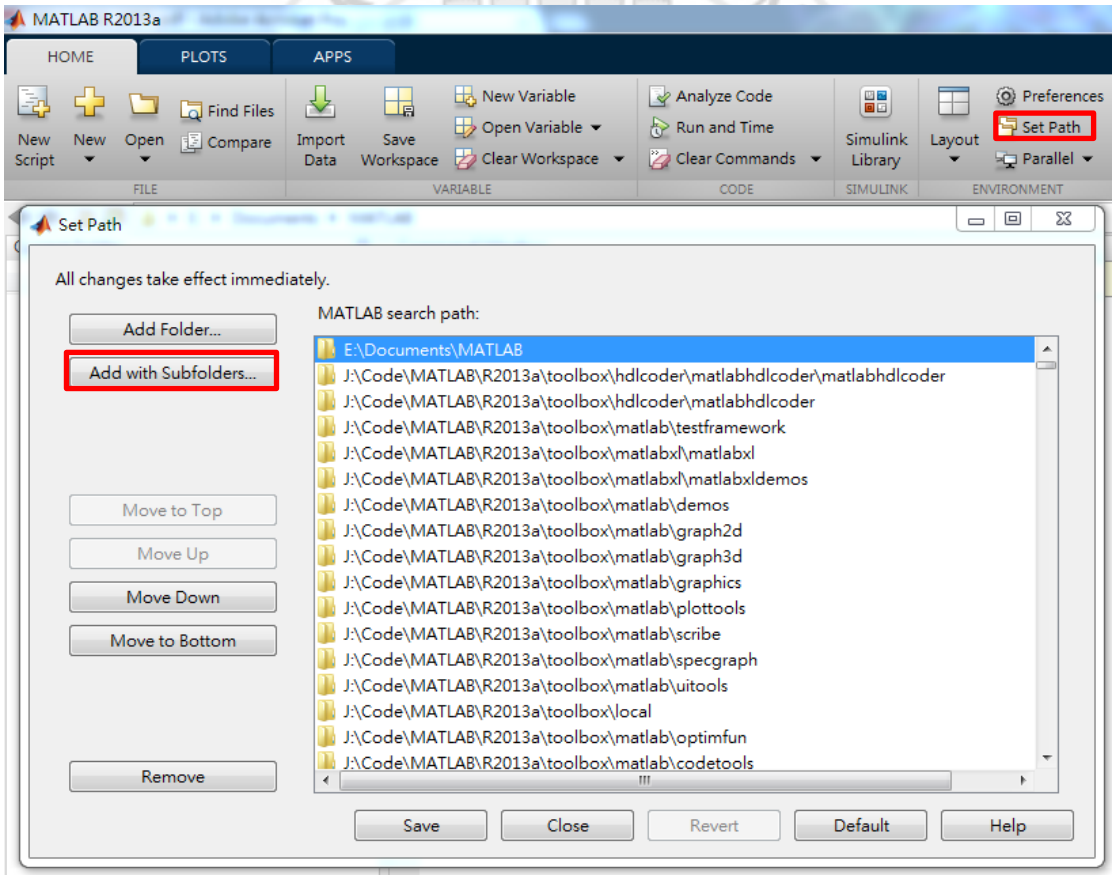


Figure 12: Steps of install extra toolbox into Matlab

The source code we have used in the project such as LIBELM and ELM (both basic ELM, kernel ELM and OS-ELM) are available on the website [15]. ELM http://www.ntu.edu.sg/home/egbhuang/elm_codes.html. All of them are written in MATLAB. The source code of multi-layer ELM algorithm, please reference the paper [23].

CHAPTER 10. References

[1] R. Panda, B. Rocha, and R. P. Paiva, "Multi-Modal Music Emotion Recognition: A New Dataset, Methodology and Comparative Analysis" *Proceedings of the 10th International Symposium on Computer Music Multidisciplinary Research (CMMR), Marseille, France, October 2013*.

[2] X. Hu, J.S. Downie, C. Laurier, and M. Bay. The 2007 MIREX Audio Mood Classification Task: Lesson Learned. *In International Society for Music Information Retrieval Conference*, p 462–467, 2008.

[3] Lartillot, O. & Toiviainen, P. (2007). MIR in Matlab (II): A Toolbox for Musical Feature Extraction From Audio. *International Conference on Music Information Retrieval*, Vienna, 2007.

[4] O. Lartillot, P. Toiviainen and T. Eerola, "Manual1.6.1 , *MIRtoolbox — Humanistinen tiedekunta*. [No publication date or modified date available] [PDF]. Available: https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox/MIRtoolbox1.6.1guide. [Accessed: 9 May 2015].

[5] Tang, Yongchuan, Huynh, Van-Nam, Lawry, Jonathan, *Integrated Uncertainty in Knowledge Modelling and Decision Making: International Symposium, IUKM 2011, Hangzhou, China, October 28-30, 2011, Proceedings*, 1st Edition. Springer-Verlag Berlin Heidelberg, 2011, page 257.

[6] Abdull, " File:ADSR parameter.svg" , "*File:ADSR parameter.svg"*. [13:37, 1 March 2007] [SVG]. Available: http://en.wikipedia.org/wiki/File:ADSR_parameter.svg. [Accessed: 9 May 2015].

[7] Grey, J. M., Gordon, J. W., (1978). Perceptual effects of spectral modifications on musical timbres. *Journal of the Acoustical Society of America 63*, 1493–1500.

[8] R. Jang, "*Audio Signal Processing and Recognition*", "*Audio Signal Processing and Recognition*" [No publication date or modified date available] [ASP], Available: http://mirlab.org/jang/books/audioSignalProcessing/index.asp. [Accessed: 9 May 2015].

[9] Dodge, Y. (2003) *The Oxford Dictionary of Statistical Terms*, OUP. ISBN 0-19-920613-9

[10] Dimitrios Giannoulis; Michael Massberg; Joshua D. Reiss (October 2013). "Automating Dynamic Range Compression". *Journal of the Audio Engineering Society (Audio Engineering Society) 61 (10)*. Section 2.1.3.

[11] C. Harte, M. Sandler, and M. Gasser. Detecting Harmonic Change in Musical Audio. In *Proceedings of the Audio and Music Computing for Multimedia Workshop (in conjunction with ACM Multimedia 2006)*, October 27, 2006, Santa Barbara, Canada

[12] Altman, N. S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression". *The American Statistician 46 (3)*: p 175–185.

[13] Vapnik, V. (1995). "Support-vector networks". *Machine Learning 20*: p 273.

[14] Boser, B. E.; Guyon, I. M.; Vapnik, V. N. (1992). "A training algorithm for optimal margin classifiers". *Proceedings of the fifth annual workshop on Computational learning theory - COLT '92*. p. 144.

[15] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[16] Boser, B. E.; Guyon, I. M.; Vapnik, V. N. (1992). "A training algorithm for optimal margin classifiers". *Proceedings of the fifth annual workshop on Computational learning theory - COLT '92*. p. 144.

[17] Guang-Bin Huang, Qin-Yu Zhu, Chee-Kheong Siew, Extreme learning machine: Theory and applications, *Neurocomputing, Volume 70, Issues 1–3*, December 2006, Pages 489-501, ISSN 0925-2312, http://dx.doi.org/10.1016/j.neucom.2005.12.126.

[18] G.-B. Huang, Learning capability and storage capacity of two hidden-layer feedforward networks, IEEE Trans. Neural Networks14 (2) (2003) 274–281.

[19] G.-B. Huang, H.A. Babri, Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions, IEEE Trans. Neural Networks 9 (1) (1998) 224–229.

[20] G.-B. Huang, N-Y. Liang, H-J. Rong, P. Saratchandran, N. Sundararajan, On-Line Sequential Extreme Learning Machine, *Computational Intelligence, 2005, p 232—237.*

[21] Antti A., "*Example of k-nearest neighbour classificationnb*", "*Example of k-nearest neighbour classificationnb*" [28 May 2007] [PNG], Available:

http://commons.wikimedia.org/wiki/File:KnnClassification.svg. [Accessed: 14 May 2015]. (By Antti Ajanki AnAj (Own work) [GFDL (http://www.gnu.org/copyleft/fdl.html), CC-BY-SA-3.0 (http://creativecommons.org/licenses/by-sa/3.0/) or CC BY-SA 2.5-2.0-1.0 (http://creativecommons.org/licenses/by-sa/2.5-2.0-1.0)], via Wikimedia Commons)

[22] G.-B. Huang, “An Insight into Extreme Learning Machines: Random Neurons, Random Features and Kernels,”*Cognitive Computation*, vol. 6, pp. 376-390, 2014.

[23] L. L. C. Kasun, H. Zhou, G.-B. Huang, and C. M. Vong, “Representational Learning with Extreme Learning Machine for Big Data,” *IEEE Intelligent Systems*, vol. 28, no. 6, pp. 31-34, December 2013.

[24] Yi-Hsuan Yang; Yu-Ching Lin; Ya-Fan Su; Chen, H.H., "A Regression Approach to Music Emotion Recognition," Audio, Speech, and Language Processing, IEEE Transactions on , vol.16, no.2, pp.448,457, Feb. 2008