基於使用者偏好之哼唱式音樂檢索系統

許瀚元
國立屏東科技大學資訊管理系
研究生
M9756018@mail.npust.edu.tw

王俊穎
國立屏東科技大學資訊管理系
研究生
M10156016@mail.npust.edu.tw

劉寧漢*
國立屏東科技大學資訊管理系
副教授
gregliu@mail.npust.edu.tw

摘要

隨著網路快速的發展及硬體儲存能力的增加，帶動了數位音樂的蓬勃發展，而如何在數量龐大的音樂資料中搜尋所需之歌曲，便為重要的課題。而最為簡單、直覺的查詢方式便是哼唱式音樂檢索，使用者只需哼唱一個音樂片段，將此片段當作查詢條件，進而與音樂資料庫中的音樂進行比對，傳回近似查詢的結果清單供使用者確認點選。但音樂間常有相似片段以及使用者往往無法精確地哼唱歌曲旋律，造成系統比對完音訊後會回傳許多並非使用者所需之查詢結果，而過多的查詢結果顯示於螢幕大小受限的行動通訊裝置上，使用者便需要頻繁使用翻頁之動作找尋音樂，使用上相當不便。所以本研究在查詢結果上加以考慮使用者以往的音樂查詢紀錄以及音樂偏好資料之推薦機制，將符合使用者喜好之音樂排序順序優先於其它音樂之前，讓使用者想查詢之音樂較可能出現在查詢結果的前列，以適合實作於螢幕大小受限之行動通訊裝置上。

關鍵詞：哼唱式音樂檢索、音樂查詢紀錄、音樂偏好資料

Abstract

The digital music is booming through rapid expansion of Internet and the increase of storage capacity of hardware. And how to search the intended music information is an important issue. The simplest way is query by humming. Users simply need to sing a piece of music, and this fragment will be searched as a query. However there is similarity between music clips. The users are so often unable to accurately sing the song melody that causing the system returns a lot of query results. Excessive query results displayed on mobile devices with limited screen size that will be flipped frequently to find music by users. It's quite inconvenient to use. So this research will add user's query records and music preference data to re-rank-music search results. It will be more likely appeared in the forefront of the query list which making the query music that the users wants. And it will be implemented to fit on screen size-constrained mobile devices.

Keywords: Query by humming, user's query records, music preference data

1. 前言

靠著音樂檢索技術的幫助，人們可以在現今網路音樂資料量非常龐大的情況下順利地的搜尋歌曲，但若沒有充分掌握該歌曲資訊，例如歌曲名稱、歌手資訊或專輯名稱等歌曲相關資訊，便無法利用相關的關鍵字去檢索音樂；而且許多歌名也不一定會符合其歌曲所含意境，使用者就算知道歌曲之中是如何哼唱也很難與歌名作聯想的連結，此時使用者想尋找該歌曲就顯得相當困難，而解決這種狀況最好的查詢方式就是直接哼唱一段音樂片段，將此音樂片段當作是查詢的條件，與音樂資料庫的音樂相互比對找出最相似的音樂，此種方法稱為—哼唱式檢索(Query by Humming, QBH)。

而智慧型行動通訊裝置近年來已非常普及，利用語音識別系統識別語音內容後進行查詢的技術已達一定水準，如:Apple 的 Siri 與 google 的語音識別系統皆已相當成熟，使用者只需唸出音樂名稱或歌手名字等訊息，語音識別系統便會辨識其內容進行查詢。而當使用者忘記音樂相關資訊，僅記得該音樂的某段旋律，便可藉由 QBH 來檢索音樂。但由於音樂間常有相似的旋律片段以及使用者哼唱時可能無法精確的哼出音樂旋律，以至於 QBH 系統會回傳許多的近似查詢結果，而過多的查詢結果在行動通訊裝置上顯示，由於行動裝置的螢幕大小受限，每個頁面所能顯示的音樂數量不多，造成使用者需頻繁使用翻頁之動作找尋所需之音樂，使用上相當不便。要如何將使用者所檢索之音樂排序在查詢清單前列，讓使用者可以在第一個頁面便可找到所檢索之音樂是本研

究之目標。

以往 QBH 只單純依照旋律的比對相似度排序查詢結果，但查詢清單中相鄰的音樂之間的音樂類型可能差異很大，造成較符合使用者偏好的音樂類型可能被排序在查詢清單的後列，所以在利用哼唱式系統檢索音樂時，在哼唱音訊與音樂資料庫中的音樂比對其旋律相似度後，應加以考慮個人的音樂查詢習慣與對不同音樂類型的偏好程度。例如某一位年紀較輕且熱愛搖滾音樂的使用者會查詢的音樂就比較不可能是 70 年代的藍調音樂，如果查詢的結果包含現今流行的搖滾樂及 70 年代的藍調音樂，則搖滾樂的音樂清單排序應較藍調音樂優先。讓較可能為使用者所查詢、喜好的音樂排序在音樂清單前列，減少在行動通訊設備上因為螢幕頁面所能顯示的音樂數量不多，而要翻頁去尋找所檢索之音樂的不便。

2.相關研究

2.1 哼唱式音樂檢索

隨著數位音樂科技的蓬勃發展，數位音樂資料量已非常龐大，音樂資訊檢索(Music Information Retrieval, MIR)技術逐漸受到研究與探討。音樂關鍵字檢索是將音樂資料加以分析、組織，依照歌名、歌手、作曲者等項目分門別類，只要關鍵字詞彙訂定的好，音樂關鍵字檢索的效果也會很好，但此種檢索因無參照音樂本身資訊，有時無法滿足使用者對音樂資訊檢索的需求；內容式音樂檢索部分，是基於音樂音訊資料之內容，即運用音訊本身內容的資訊，如音色、音高、節奏，等數個音訊特徵值，以此作為系統檢索音樂時的依據。

長時間以來內容式音樂檢索儼然成為研究者所著重的研究議題，並且在網路速度、資料庫技術與電腦運算能力取得巨大的進展之際，研究者開始進行此領域中具備互動與人性化的哼唱音樂檢索之研究，1995 年學者 Ghias[2]等人開發一套哼唱式查詢(Query By Humming, QBH)系統，使用者只需哼唱一段音樂旋律，使用自相關函數(Autocorrelation Function, ACF)將此哼唱旋律轉換為音高，並搭配旋律輪廓的概念將旋律輪廓當作查詢條件比對音樂資料庫的音樂，進行音樂檢索。Ying G.S.[8]則是利用平均振幅差函數(Average Magnitude Difference Function, AMDF)將音樂旋律轉換成音高以利之後的查詢比對。但在基頻求取音高的 ACF、AMDF 所求出音高較為不精準，且在音訊資料眾多時將產生龐大的運算量，為了更精確、更迅速的抓取旋律音高，便有其他學者提出一種基於頻域的音高追蹤方法，利用倒頻譜(Cepstrum)[3]先將音訊資料切割成音框，以快速傅利葉轉換(Fast Fourier Transform, FFT)做轉換運算，找出時域上最高能量位置以求得音高。

哼唱式音樂檢索系統目前研究主要是運用音訊資料裡的音高、旋律、節奏等資料[9]進行音樂歌曲的匹配，由於使用哼唱旋律作為查詢的依據，是基於音樂的基本特質，即便是使用者忘記歌曲名稱、歌手或專輯等相關資訊，而只要記得某段音樂旋律，藉由哼唱式音樂檢索系統仍可在大型音樂資料庫中進行音樂查詢。

2.2 端點偵測

端點偵測法 (Endpoint Detection)或稱為語音活動偵測法(Voice Activity Detection, VAD)是指可以將一個語音中每段有聲音片段的開始與結束位置偵測出來的演算法，以判別語音中那些區段是屬於有聲片段，哪些是無聲片段。常見的端點偵測法可以分成時域與頻域兩大類，時域是透過設定音量大小與過零率(Zero Crossing Rate)的門檻值進行端點偵測[1]；頻域則是使用傅立葉轉換(Fourier Transform)將音訊轉換成頻譜，藉由頻譜的變異數或熵(Entropy)的高低作為決定端點位置的判斷。

2.3 音高追蹤

音樂旋律由一連串的音高高低變化所組成，故哼唱式檢索系統在查詢時，即依據哼唱音訊裡的音高的高低變化當作查詢依據，在資料庫裡找出相應的旋律。一般音高是經由人們聆聽一段音樂之後，人們對於音樂片段主觀的判斷其旋律的音高高低，而要使電腦程式對一段音樂裡的旋律作客觀的音高判斷過程，通常稱為音高追蹤(Pitch Tracking)。音高追蹤首先要先把有聲音訊切割成音框且相鄰音框可以重疊(Overlap)，然後計算每個音框的基頻，計算音框基頻的方式分為時域、頻率兩大類，以 ACF、AMDF 為代表的時域計算方式是利用音框週期位置上的相似性取得基頻[8]；以 Cepstrum 為代表的頻域計算方式是利用 FFT 將音框轉換到頻譜上，對應新的頻譜與音框找出凸顯的高點以求得基頻[3]。時域與頻域的方法各有其

優劣，較新的研究偏向結合兩種方法進行音頻提取，如 YAAPT(Yet Another Algorithm for Pitch Tracking)[6]就是混合兩方法，同時進行時域與頻域的基頻候選提取，然後透過動態規劃(Dynamic Programming)選擇最佳的基頻候選。

2.4 編輯距離

編輯距離(Edit Distance)[5]由俄羅斯科學家 Vladimir Levenshtein 在 1965 年提出，所以也稱 Levenshtein distance，編輯距離是一種計算兩個字串之間距離的方法，計算由第一個字串轉變成另一個字串最少要經由多少次的刪除(Delete)、插入(Insert)、代換(Substitute)的基本字串運算，才會變成第二個字串。如果距離越大代表兩字串越不相同，反之，距離如果是零則代表兩字串是完全相同的，藉由計算編輯距離的大小可以判別兩字串的相似程度。

2.5 推薦系統

推薦系統屬於一種可以有效過濾資訊的應用，在可能的選擇項目裡將可能受使用者喜好的資訊推薦給使用者。目前常見的音樂推薦系統大致分為內容導向式推薦系統(Content-Based Recommend System)、合作式過濾推薦系統(Collaborative-Filtering Recommend System)、混合式推薦系統(Hybrid Recommend System)。

2.5.1 內容導向式

內容導向式的音樂推薦方式，有些是以音樂的元資料(Metadata)描述音樂的外部特徵以進行音樂分類，達到音樂訊息的管理；而另一部分則對音樂內容特徵進行分析，根據音樂項目間的關聯性比對使用者的過去的偏好資料加以分析並提供推薦結果。此方法首先必須建立使用者輪廓，此輪廓主要記錄一些關鍵詞或是使用者的相關資訊，經由過濾系統將每一份新加入的音樂與其他音樂做比對，以確保過濾後的音樂與使用者的偏好相吻合的並推薦給使用者。Phelan O.[4]利用公共 Twitter 的訊息資料與使用者好友的 Twitter 的訊息資料重新排序使用者 Twitter 訊息的排序。音樂方面學者 Wang X.[7]結合智慧型手機裡的活動排程，依照使用者的環境產生適合的播放清單。

2.5.2 合作式過濾法

此方法也稱為社會性過濾法(Social Filtering)，合作式過濾法可記錄使用者的偏好，透過分析使用者過去的偏好，計算出偏好相似的使用者分成同一群聚，再利用相同群聚對項目的評價預測使用者的對項目的偏好。Zhang Y.[10]從微型部落格收集使用者對電視節目的喜好，將這些資料作為使用者輪廓，將使用者分成群聚並推薦其電視節目。合作式過濾推薦系統可大致分為兩大類：以模型為基礎的合作式過濾(Model-Based Collaborative Filtering)：記錄使用者習慣、偏好等特徵建立使用者模型，透過使用者模型選擇適合的音樂為推薦；以使用者偏好為基礎的合作式過濾(User-Based Collaborative Filtering)：利用統計計算使用者的偏好與音樂特徵，將具有相似音樂偏好的使用者分群進行推薦。

2.5.3 混合式推薦

由於各種推薦方法在實際應用上個別遭遇到一些問題，因此有學者提出結合兩種或兩種以上不同的資訊篩選技術作組合，產生混合式推薦法(Hybrid Recommendation Method)，一般常見的混合式推薦是將內容式與合作式過濾法作結合來實作，以適當的解決上述兩推薦的缺點，產生最佳的推薦結果。混合式推薦系統主要分為兩大類：第一類是將兩種推薦方法分別獨立產生一組推薦清單；合作式推薦是比對使用者與相同群聚的網站瀏覽路徑，以推薦相關的網站。第二類是將兩種推薦方法結合，一起產生一組推薦結果。

3. 系統架構與流程設計

在問題的假設上，現實生活中當使用者利用哼唱式查詢查詢歌曲，傳統的 QBH 只依照比對後的旋律相似度大小進行音樂清單的排序，使得排序相鄰的音樂之間音樂曲風可能是差異很大的，造成使用者所偏好的音樂可能不在音樂清單的前列。此種情況發生在現今相當普及的行動通訊裝置上則更為不便，導致使用者得頻繁的使用翻頁的動作找尋音樂，若能進一步的分析 QBH 的查詢結果，考慮到各個使用者的歷史查詢紀錄與音樂偏好，重新調整查詢結果的排序，使得符合使用者偏好的音樂排序能優先於其他音樂，將能改善此缺失提升查詢清單的準確性。

3.1 系統架構與方法

本研究系統架構在流程初期的哼唱查詢步驟，依照一般 QBH 系統的音高擷取與查詢比對流程進行哼唱音訊與資料庫音樂的比對，得到查詢音訊與資料庫音樂的相似度排序音樂清單。接著系統會找尋使用者是否有過查詢的紀錄，並比較該次查詢與歷史查詢紀錄之間的差異度，若該次哼唱查詢的訊號與歷史記錄之間的差異度在設定的 threshold 之內，第一階段便將在 threshold 之內且差異度最小的歌曲作為查詢答案排序在清單第一的位置。之後，再利用登載系統所收集的使用者音樂類型之偏好程度作計算，找出使用者較傾向的音樂，重新進行清單排序。若哼唱查詢的訊號與歷史記錄之間的差異度超越設定的 threshold，則會忽略歷史記錄的資料，直接利用使用者對音樂類型之偏好程度來重新排序音樂清單。

清單推薦系統的運作流程，如圖 3-1 所示。

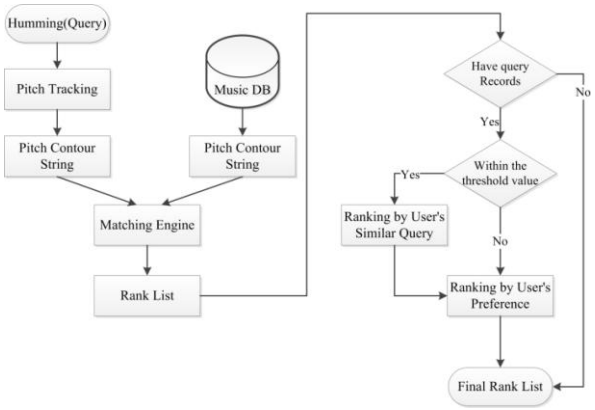

圖 3-1 系統流程

因為在系統重新排序查詢結果時，將使用到使用者的個人資料與歌曲的文字性描述資料。所以在使用者首次使用本系統前，需登載個人資料。使用的類別有性別、年齡、母語、居住地、職業、教育程度，表 3-1 為某使用者之個人資料範例。

表 3-1. 使用者資料的範例

Attribute ID	Attribute	Value
A_1	性別	男
A_2	年齡	29
A_3	母語	中文
A_4	居住地	基隆
A_5	職業	學生
A_6	教育程度	碩士

而歌曲的文字性描述資料則參考網路上對於該歌曲的評論，經由人工的方式來建立資料，表 3-2 為歌曲的描述資料範例。

表 3-2. 音樂資訊的範例

Attribute ID	Attribute	Value
MA_1	編號	1
MA_2	歌手性別	男
MA_3	年代	1990
MA_4	音樂類別	搖滾樂
MA_5	節拍速度	快

3.2. 哼唱式音樂檢索系統

在此小節中，將探討哼唱式音樂檢索(Query by humming, QBH)之處理流程，其查詢結果的排列順序是直接依照查詢音訊與資料庫音樂的旋律相似度來排序。

QBH 系統包含三個主要組成部分，音高追蹤、音高輪廓、匹配引擎，如圖 3-2 所示。

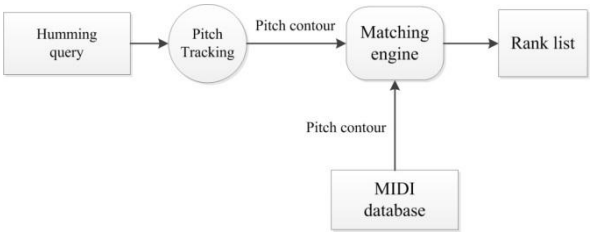

圖 3-2 QBH System

3.2.1 音高追蹤(Pitch tracking)

首先使用端點偵測決定音訊開始和結束的位置，我們使用過零率(Zero Crossing Rate)來判斷音訊起始與結束位置，設定音框中音訊通過零點次數的門檻值，當音訊通過零點的次數低於設定之門檻值便決定音訊之端點，即可有效的切割每段音訊的端點，如圖 3-3 所示。

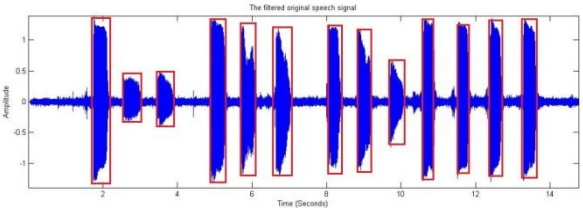

圖 3-3 端點偵測

將每段紅框內的音訊訊框切成音框(frame)，每個音框長度為 35 毫秒且相鄰的音框重疊(overlap)為 25 毫秒，接著便要計算出每個音框裡的基頻(fundamental frequency)，這邊我們利用 Stephen A.Zahorian 和 Hongbing Hu 所提出的 YAAPT(Yet Another Algorithm for Pitch Tracking)計算基頻，該方法結合時域與頻域的

估計法進行音頻的提取，同時進行時域和頻域的音調提取，之後透過動態規劃(Dynamic Programming)從多個音調候選中找出最佳的音調特徵值。

將哼唱之音訊藉由 YAAPT 計算其基頻，如下圖 3-4 所示。

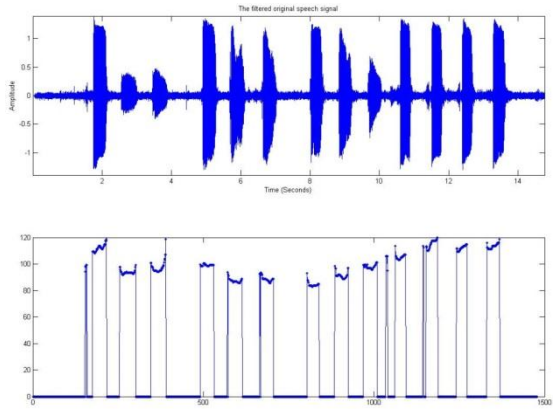

圖 3-4 哼唱小蜜蜂音訊基頻計算

3.2.2 音高輪廓(Pitch contour)

我們將查詢語句之頻率依照音高與頻率的對照表，將週期基頻值進行音高對照以求得每個訊框裡的音高，如表 3-3。介於兩半音之間的頻率，以兩半音所對應頻率的平均值為劃分範圍，以 Octave 1 的 C#與 D 音為例，兩半音所對應頻率的平均值為 35.65，則 34.6~35.65 範圍的頻率我們視為 C#音，35.65~36.7 範圍的頻率視為 D 音(不包含 35.65)。

表 3-3 音高與頻率對照表

Note	Octave 2	Octave 3	Octave 4	Octave 5	Octave 6
A	55.0	110.0	220.0	440.0	880.0
B^b	58.3	116.5	233.1	466.2	932.3
B	61.7	123.5	246.9	493.9	987.8
C	65.4	130.8	261.6	523.3	1046
C#	69.3	138.6	277.2	554.4	1108
D	73.4	146.8	293.7	587.3	1174
E^b	77.8	155.6	311.1	622.3	1244
E	82.4	164.8	329.6	659.3	1318
F	87.3	174.6	349.2	698.5	1396
F#	92.5	185.0	370.0	740.0	1480
G	98.0	196.0	392.0	784.0	1568
G#	103.8	207.7	415.3	830.6	1661

在端點偵測裡切割的每段音訊開始和結束的位置裡，可能因為使用者聲音抖動而會取得許多不同的音高，但使用者其實在這段音訊裡只想表達一個音高，且已知人聲的音高範圍大約在 50Hz 到 1000Hz 之間，所以我們排除人聲範圍外的音高，將每段音訊裡出現次數最多的音高代表該段音訊所要表示的音高，將音調平滑化。

由於大部分的使用者不具有絕對音準，人聲可能與音樂所使用的調性不同，無法直接利用音高進行比對，也就是起音之音高往往與欲查詢之音樂有差距，因此，我們將進一步把音高字串轉換為 Pitch contour string，也就是計算一個音與前一個音之間的半音值差異，所以單純的音高描述將轉換為前後音高變化量產生 Pitch contour string，以提高比對容錯性。

3.2.3 比對引擎(Matching engine)

哼唱的 Pitch contour string 與音樂資料庫的 Pitch contour string 之間的差異程度，我們使用 Edit distance 計算法計算兩 Pitch contour string 間之距離。

由哼唱的 Pitch contour string 轉變成音樂資料庫的 Pitch contour string 每經由一次刪除步驟則 cost 加 1、一次插入步驟則 cost 加 1、一次代換步驟則 cost 加 1，下列為採用 Edit distance 運算時採用之計算式：

$$\mathrm{D}[x,y] = min\begin{Bmatrix} D[x-1,y]+1 \\ \mathrm{D}[x-1,y-1]+1 \\ \mathrm{D}[x,y-1]+1 \end{Bmatrix} \qquad (3\text{-}1)$$

利用 Edit distance 計算出之兩字串總編輯成本做為歌曲與查詢之相異度；計算各備選歌曲之相異度後，找出相異度排名最小前 k 名，作為音樂清單顯示於移動設備螢幕上，由使用者決定是否為點播之歌曲。

3.3 依據使用者的歷史查詢資料進行重新排序

在此小節中，將探討系統收集足夠的使用者查詢的歷史資料時該如何重新排序音樂清單，根據我們的觀察，同一位使用者對同一首歌曲的查詢，不論經過了幾次查詢，所哼唱出的音樂片段往往都是相近的訊號，而此現象在我們後續的實驗中也獲得證實。因此為了避免使用者要查詢以往查詢過的相同歌曲時，判定對應過的正確歌曲並非在清單前列且還需重複花費時間在回傳歌曲的判定上，我們將使用者每次的查詢及使用者判定對應過的正確歌曲記錄下來。紀錄方式如下表所示：

表 3-5 使用者歷史查詢記錄

No.	User	Query Contour	Matching Music

	ID	String	ID (MA1)
1	User1	+1, +2, +1, -2	22
2	User2	-2, +3, 0, +1	52
3	User2	-3, 0, 0,+2, +3	63
4	User3	+1, +1, +2, -3, +5	103
5	User1	+1, +3, +1, -2	40

將使用者哼唱之訊號轉為輪廓字串後(表示為 CSq)，首先會比對 useri 自己的歷史查詢記錄的 Query contour string(表示為 CSh)，找出 edit distance 在設定的門檻(表示為δ)之內的字串(表示為 Ruseri)，之後再從其中尋找 edit distance 最小的一筆記錄，下列為計算過程：

$$\text{Music}_{CS_h} = arg\ min\ Edit(CS_q, CS_h)$$
$$CS_h \in Ruser_i \quad (3\text{-}2)$$
$$\min \text{Edit}(CS_q, CS_h) < \delta$$

該筆記錄所記載的 matching music 即作為查詢答案排序在清單第一的位置，如果最小值的紀錄有兩筆以上，考慮越新的查詢紀錄越可能是使用者想要查詢的音樂，所以選擇 NO 數較大的紀錄為查詢答案。對於門檻值參數δ的設定，我們將在實驗中進行測試並找出適當的設定值。

3.4 依據使用者偏好資料進行重新排序

此查詢將利用 QBH 之比對法對於資料庫內的音樂進行相似度計算。經由相似度計算後可得到 k 條相似度較高的歌曲(k 值將為使用者螢幕在一個畫面中可呈現的歌曲選單數量的三倍)，此 k 條歌曲被稱為候選集合。在候選集合中的歌曲雖然和查詢之間有不同的相似度值，但我們將其均視為可能的答案，也就是在後續的排序時將忽略這些相似度值。

使用者查詢的音樂通常與自己的偏好有關，例如某個喜愛搖滾樂的使用者，查詢搖滾樂的機率比查詢古典樂的機率高。因此基於此假定，此查詢考慮使用者過去查詢音樂類型的偏好。將使用者查詢記錄與音樂資訊經由關聯運算後可得如下表格：

表 3-6 使用者音樂偏好記錄表

MA_1	MA_2	MA_3	MA_4	MA_5	MA_6
1	男	2010	搖滾樂	快	日文
3	女	2010	搖滾樂	極快	英文
52	男	1990	R&B	慢	英文
70	男	1990	R&B	快	中文
155	女	1970	Blues	慢	英文

針對 MA2 至 MA5 各屬性出現過的音樂特性標籤，均可從表 3-6 計算出對應的機率值。假設 MAi 屬性共有 x1, x2,…, xn 種音樂特性標籤，則使用者對於 xj, j=1, 2, …, n 的偏好程度則定義為

$$L(x_j) = \frac{S_j}{S} \quad (3\text{-}3)$$

S 為表 3-6 裡的記錄數量，Sj 為在 MAi 裡出現 xj 的次數。

針對候選集合中的每首歌曲，我們可以根據其對應的屬性值計算使用者對該歌曲的偏好程度。假定某一候選音樂 Mx 具有(xma2, xma3, xma4, xma5, xma6)之屬性值，則使用者對該歌曲的偏好程度為

$$Like(M_x) = \frac{\sum_{k=2}^{6} L(X_{ma_k})}{5} \quad (3\text{-}4)$$

舉例來說，假設在候選集合中有一首歌曲 M1 其對應的屬性值為 MA2=男, MA3=2010, MA4=搖滾樂, MA5=快, MA6=英文，則偏好程度 Like(M1)=(3/5+2/5+2/5+2/5+3/5)/5=0.48。假設在候選集合中另一首歌曲 M2 其對應的屬性值為 MA2=女, MA3=1990, MA4=Hip-hop, MA5=快, MA6=日文，則偏好程度 Like(M2)=(2/5+2/5+0/5+2/5+1/5)/5=0.28。比較兩首歌曲，使用者應該比較傾向查詢的是 M1 這首歌曲，若兩首以上的歌曲偏好程度值相同時，則比較原先利用 QBH 計算出的相似度，相似度較小之歌曲排序在前面。查詢結果之順序即依據候選集合中所有的使用者偏好值進行排序。

4. 實驗分析

本章將根據第三章所提出之架構與方法，利用使用者歷史資料與偏好資料重新排序 QBH 系統的查詢結果，並將重新排序之音樂清單與原始 QBH 所列音樂清單進行比較，比較使用者對於重新排序音樂清單前與排序後，在音樂清單前五首音樂便會點選的機率。

4.1 實驗環境

本研究實作之系統開發環境、音樂資料庫與錄音參數分別以下述 3 小節說明：

4.1.1 系統環境

本研究之系統開發環境，使用之軟、硬體資源如表 4-1 所示:

表 4-1 系統開發環境

硬體資源	
PC	Intel Core(TM)2 Duo CPU 2.60GHz, 2GB RAM
軟體資源	
作業系統	Microsoft Windows 7
程式語言	C#, JAVA
開發環境	Visual Studio 2010

4.1.2 音樂資料庫

本研究中，音樂資料庫的音樂來源是經由網路上有音樂背景者所歸類的 MIDI 格式音樂。相關音樂的類型與數量如表 4-2 所示。

表 4-2 音樂資料庫分類表

編號	類型	數量
1~50	搖滾	50
51~100	R&B	50
101~150	爵士	50
151~200	Blue	50
201~250	抒情	50
251~300	Hip-hop	50

4.1.3 錄音參數

在實驗環境中，其哼唱錄音參數如表 4-3 所示：

表 4-3 哼唱錄音參數設定

參數名稱	數值
取樣頻率	11KHz
音訊格式	wav
頻道	單聲道

4.2 實驗結果與分析

本實驗參加人數為三十人，男女比例為八比二，年齡分佈於 20 歲到 28 歲之間，具備基本音樂知識。

4.2.1 實驗 1.參數 δ 調整

實驗作法是先讓受測者可以自由的選聽資料庫中的音樂，選聽資料庫中音樂的期間為一個禮拜，讓受測者對資料庫中音樂的旋律有基本認知。之後，讓受測者哼唱查詢一首想聽的音樂，並從系統所列出之查詢清單中點選音樂且聆聽該首音樂確認是否為自己所查詢之音樂，若非為查詢的音樂則重新回到音樂清單中點選音樂，重複點選音樂且聆聽確認的步驟直到找到自己所查詢的音樂，當使用者聆聽且確認該首音樂為自己所查詢的音樂，便紀錄查詢時哼唱的 Pitch contour string。

最後，我們每兩天讓使用者哼唱首次查詢的同一首音樂，依然讓使用者聆聽該音樂並確認，並記錄每次查詢的哼唱 Pitch contour string，此實驗進行 10 天，計算每個使用者每次查詢的哼唱 Pitch contour string 與首次查詢的哼唱 Pitch contour string 之間的 Edit distance。若 Edit distance 值越大代表首次查詢哼唱的旋律與後幾次查詢所哼唱的旋律越不相同；若 Edit distance 值越小表示首次查詢哼唱的旋律與後幾次查詢旋律幾乎無異。

實驗結果：

表 4-4 使用者哼唱同一首音樂旋律差異度

查詢次數 / User-ID	二	三	四	五	六	Avg.	σ
1	1	1	0	0	0	0.4	0.49

表 4-5 平均旋律差異度

User-ID	Avg.	σ	User-ID	Avg.	σ
1	0.4	0.49	16	1	0.89
2	0.8	0.4	17	0.2	0.4
3	0	0	18	0.6	0.49
4	0.8	0.4	19	0.4	0.49
5	0.2	0.4	20	0	0
6	0.4	0.49	21	1.2	1.94
7	0.2	0.4	22	0.4	0.8
8	0.2	0.4	23	0.8	0.98
9	0.4	0.49	24	1	0.63
10	1	0.63	25	0.8	0.75
11	1.2	0.4	26	0.4	0.49
12	0.4	0.49	27	0	0
13	0.2	0.4	28	0.2	0.4
14	0.8	0.4	29	0.4	0.8
15	1	0.89	30	0.2	0.4
GA				0.52	0.54

結果說明：

表 4-4 是記錄一個使用者首次查詢哼唱的 Pitch contour string 與之後哼唱的 Pitch contour string 之間的 Edit distance 並計算其平均與標準

差，表 4-5 則是三十位使用者的總平均(General Average, GA)的 Edit distance 與標準差(σ)。從表 4-5 可以看到三十位使用者的在哼唱同一首音樂時的平均 Edit distance 為 0.52，標準差為 0.54，平均 Edit distance 超過 1 且平均標準差接近 2 的只有一筆，這表示大部分使用者對於哼唱同一首歌時，總是哼唱自己所認知的音樂旋律，就算在每次查詢聆聽完音樂後，也依然是哼唱首次查詢時自己所認知的音樂旋律。

實驗 1 結論：

考量實驗 1 所得的資料，本研究所設計的系統中 Similar Query 機制門檻值參數δ便設定為 2。在使用者進行哼唱查詢時 Similar Query 機制便會將哼唱 Pitch contour string 與歷史紀錄裡的 Query Contour String 進行 Edit distance 計算，若 edit distance 最小的一筆記錄的 edit distance 在設定的δ值 2 內，則 Similar Query 機制便會將該筆記錄所記載的 matching music 作為查詢答案排序在清單第一的位置；如果 edit distance 最小的一筆記錄的 edit distance 超過 2，Similar Query 機制便排除使用歷史紀錄資料重新排序音樂清單。

4.2.2 實驗 2.比較 Traditional QBH/ Ranking by User's Preference QBH 排序之音樂清單，在查詢準確率的差異。

我們的系統設計主要是考量到行動通訊裝置螢幕大小的受限，假設行動通訊裝置螢幕一個頁面只顯示的音樂清單數量為 N 首，使用者想查詢的音樂若出現在螢幕第一頁的 N 首音樂之中，我們便是為當次的查詢是成功的。所以這邊的查詢準確率指的是在多次查詢中，使用者所查詢音樂在清單 TOP N 裡的比例。我們對查詢準確率的定義如下：

$$\text{查詢準確率} = \frac{\text{查詢音樂在清單TOP N的次數}}{\text{哼唱查詢總次數}} \qquad (4\text{-}1)$$

我們實驗的做法，以三十位受測者利用哼唱查詢系統進行音樂查詢，在音樂清單排序的部分，分別使用 Traditional QBH 直接依照哼唱音訊與資料庫音樂的旋律相似度大小進行排序(以下皆以 Traditional QBH 簡稱)，與 Ranking by User's Preference QBH 機制利用使用者偏好資料進行音樂清單的重新排序(以下皆以 User's Preference QBH 簡稱)，使用者只需在 Traditional QBH 系統所列出清單中點選音樂，該音樂也將對照 User's Preference QBH 機制所列清單中之音樂與排序，分別計算兩種方法的查詢準確率。實驗中受測者首次查詢 10 首音樂，之後每隔 7 天再進行 10 首音樂的查詢，再查詢之間使用者仍然可以自由的選聽資料庫中的音樂，此實驗共進行 1 個月，總計有 50 首音樂的查詢次數。受測者查詢時依照自己的喜好進行查詢，且不進行重複的音樂查詢。

實驗將分成三個部分進行：

一、User's Preference QBH 機制中音樂偏好記錄表裡五項音樂屬性(歌手性別、年代、類型、節奏、語言)數據的配對型態，對於不同 TOP N 的查詢準確率影響，每位受測者對於不同 TOP N 查詢的次數皆為 50 次，計算並列出平均值。

二、Traditional QBH 與 User's Preference QBH 在不同 TOP N 的情況下對查詢準確率的影響，每位受測者對於每個 TOP N 查詢的次數皆為 50 次，計算並列出平均值。

三、Traditional QBH 與 User's Preference QBH 查詢次數對於查詢準確率之影響。

實驗結果：

第一部分、

●音樂屬性的查詢準確率(MA2:歌手性別、MA3:年代、MA4:類型、MA5:節奏、MA6:語言)

表 4-6 不同音樂屬性的查詢準確率

屬性 \ TOP N	Avg.
MA_2+ MA_3+ MA_4+ MA_5	56.15
MA_2+ MA_3+ MA_4+ MA_6	55.81
MA_2+ MA_3+ MA_5+ MA_6	55.92
MA_2+ MA_4+ MA_5+ MA_6	59.73
MA_3+ MA_4+ MA_5+ MA_6	62.21
MA_2+ MA_3+ MA_4+ MA_5+MA_6	57.55

查詢準確率用%表示

結果解說：

可以看出在不考慮 MA2 的四項音樂屬性(MA3+ MA4+ MA5+ MA6)的平均查詢準確率最高，比起將全部五項音樂屬性列入計算的平均查詢準確率還高了 4.66%。說明了將 MA3、MA4、 MA5、 MA6 等屬性一起參考時有助於查詢準確率的提升，而 MA2 的屬性(歌手性別)可能是使用者較不在意的項目，把 MA2 加入使用使偏好程度的計算裡可能對查詢準確率的提升沒有很大的幫助。在第二、三部分的實驗，User's Preference 機制中使用者偏好程度

的計算上便只將 MA3、 MA4、 MA5、 MA6 屬性列入偏好程度中計算並重新排序音樂清單。

實驗結果：

第二部分、

本實驗將記錄 Traditional QBH 與 User's Preference QBH 在不同 TOP N 的情況下對查詢準確率的影響，因本研究之目的是希望能夠因應行動通訊裝置受限螢幕大小之情況，讓使用者能在不需翻頁的狀況下，便能在第一頁找到想查詢的音樂，TOP 數的擴張有違本研究之初衷，因此實驗的 TOP 數應考量在較小的範圍內以符合本研究之目的，故本實驗 TOP 數的範圍便設定在 5 以下。

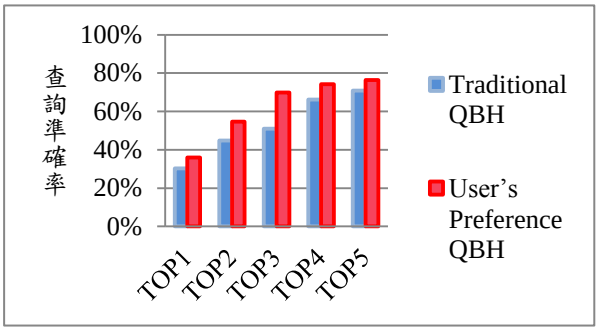

圖 4-1 不同 TOP N 情況下的查詢準確率

結果解說：

由圖 4-1 可以看出，在 TOP1 到 TOP5 的情況下 User's Preference QBH 機制的平均查詢準確率皆優於 Traditional QBH；在 TOP1、TOP2 的情況下 User's Preference QBH 機制的查詢準確率雖然是高於 Traditional QBH，但查詢準確率也只有不到 6 成；在 TOP4、TOP5 的情況下 User's Preference QBH 與 Traditional QBH 的查詢準確率皆達到 6 成以上，但兩者的查詢準確率幾乎沒有很大的差距，是因為當 TOP 數越高時，代表查詢準確率的容錯率越高。而在 TOP3 的情況下 User's Preference QBH 機制的正確率為 69.8%，是明顯優於傳統 QBH 的正確率 51.07%，且 TOP3 代表使用者查詢之音樂是出現在清單的前 3 名之中，行動通訊裝置的螢幕對於在同個頁面裡顯示 3 首音樂應無困難。以下第三部分的實驗，Traditional QBH 與 User's Preference QBH 查詢次數對於查詢準確率之影響的實驗，其中的查詢準確率便以 TOP3 的情況下作為計算。

實驗結果：

第三部分、

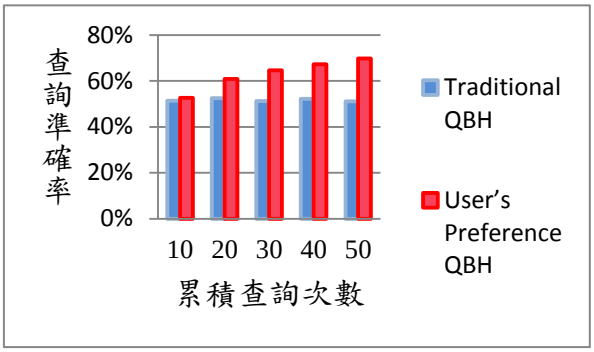

圖 4-2 查詢次數增加對查詢準確率之影響

結果解說：

由圖 4-2 可以看出，隨著查詢次數的增加，User's Preference QBH 利用使用者偏好資料調整排序清單方法的查詢準確率會逐步上升，顯示多次的查詢將可以幫助 User's Preference 機制更準確地查詢。尤其在查詢次數達到 20 次之後，查詢正確率會大幅增加，表示在系統收集到足夠的使用者偏好資料後 User's Preference 機制進行音樂清單的重新排序會表現得更好。由於查詢次數的增加，並不會改變 Traditional QBH 系統依照旋律相似度大小的排序方式，因此其查詢準確率大都在一定的範圍。實驗結果顯示，相較於 Traditional QBH 直接依照旋律相似度大小的排序方式，利用使用者偏好資料來重新排序的個人化音樂清單擁有較高的查詢準確率，且經由不斷查詢之後，查詢準確率也會逐步的上升，能讓使用者在查詢結果的 TOP3 便找到查詢的音樂，對於行動通訊裝置的使用者來說，可以在螢幕的第一個頁面就找到音樂，而不用頻繁的換頁去尋找，大大的提高了便利性。

實驗 2 結論：

由實驗 2 可以得知，User's Preference QBH 機制中參考 MA3、MA4、MA5、MA6 等 4 個屬性組合有最高的查詢準確率，並以只參考這 4 種屬性進行不同 TOP 數的查詢準確率實驗，在 TOP3 的情況下 User's Preference QBH 機制的查詢準確率遠高於 Traditional QBH 系統並達到了 7 成正確率的查詢水準，證明就算手機的一個頁面只顯示 3 首查詢結果供使用者點選，經由 User's Preference QBH 機制所排序的查詢清單，仍然有非常高的機率是符合使用者所要查詢的音樂。而在查詢次數對查詢準確率的影

響方面，也可以看出經由查詢次數的增加，經由 User's Preference QBH 所排序的查詢清單的正確率有持續增加，代表越多次的查尋越會突顯經由 User's Preference 機制所排序的清單的查詢正確率，這是在 Traditional QBH 系統上看不到的。

5. 結論

儘管就排序的方式而言，本研究所重新排序的音樂清單已較 Traditional QBH 系統依照旋律相似度大小所排列的音樂清單更符合使用者需求，但仍不能百分之百的達到完美的查詢準確率，可能是因為使用者偏好資料屬性的數量不夠充足，研究中將偏好資料中的屬性分為歌手性別、年代、曲風、節奏、語言等音樂的描述，但這不能代表音樂內容的全部，在未來的研究中可望在細分更多的音樂屬性，以求得更符合使用者喜好的偏好程度，將音樂清單做更佳的排列。

在未來的研究中，可望將本研究所提出的以使用者偏好資料調整清單的排序，搭配上其他不同的屬性資料，以求得更符合使用者需求的清單排序。另外由於本研究忽略了音長、休止符等音樂資訊，假設同樣的音高差異但不同音長的兩個旋律是代表不同的兩首音樂，但在本研究以 Pitch contour string 方式表示卻是一樣的音符字串，這會造成系統的音樂比對錯誤，導致查詢正確率的下降，在往後的研究中可以將音長、休止符等音樂資訊一起考量，以期望達到更好的查詢水準。

誌謝

本文由國科會"以 3G 網路傳輸為基礎之智慧型車載音響系統整合設計與實作"(NSC-102-2218-E-020-002-)經費支援，特此誌謝。

參考文獻

[1] Bachu, R.G., Kopparthi, S., Adapa, B., Barkana, B.D., "Voiced/Unvoiced Decision for Speech Signals Based on Zero-Crossing Rate and Energy", ***Advanced Techniques in Computing Sciences and Software Engineering,*** pp. 279-282, 2010.

[2] Ghias, A., Logan, J., Chamberlin, D., Smith, B., "Query by Humming: Musical Information Retrieval in an Audio Database", ***MULTIMEDIA '95 Proceedings of the third ACM international conference on Multimedia,*** pp. 231-236, 1995.

[3] Kim, H. K., Choi, S. H., "Cepstral Domain Interpretations of Line Spectral Frequencies", ***Signal Processing*** Volume 88, Issue 3, March 2008, pp. 756–760, 2008.

[4] Phelan, O., McCarthy, K., Bennett, M., Smyth, B., "Terms of a Feather: Content-Based News Recommendation and Discovery Using Twitter", ***Advances in Information Retrieval Lecture Notes in Computer Science*** Volume 6611, 2011, pp. 448-459, 2011.

[5] Sokol, D., Tojeira, J., "Speeding Up the Detection of Tandem Repeats Over the Edit Distance", ***Theoretical Computer Science Available*** online 7 May, 2013.

[6] Stephen, A. Z. and Hongbing, Hu, "A Spectral/Temporal Method for Robust Fundamental Frequency Tracking", ***Journal of the Acoustical So- ciety of America,*** vol. 123, pp. 4559–4571, 2008.

[7] Wang, X., Rosenblum, D., Wang, Y., "A daily, activity-aware, mobile music recommender system", ***Proceedings of the 20th ACM international conference on Multimedia,*** pp. 1313-1314, 2012.

[8] Ying, G.S., Jamieson, L.H., Michell, C.D., "A Probabilistic Approach to AMDF Pitch Detection", ***Proceedings of the 1996 International Conference on Spoken Language Processing, Philadelphia***, PA, Oct, pp. 1201-1204, 1996.

[9] Yu, H.-M., Tsai, W.H., Wang, H.-M., "A Query-by-Singing System for Retrieving Karaoke Music", ***Multimedia, IEEE Transactions on*** (Volume:10, Issue: 8), pp. 1626-1637, 2008.

[10] Zhang, Y., Chen, W., Yin, Z., "Collaborative filtering with social regularization for TV program recommendation", ***Knowledge-Based Systems*** Volume 54, December 2013, pp. 310-317, 2013.