基于内容的音乐检索研究[1]

王玮*　　蔡莲红**

(*香港中文大学系统工程与工程管理系)

（ **清华大学计算机科学与技术系 ）

摘要：基于内容的音乐信息检索是当前热门的研究课题，传统的音乐检索的研究对象多为 MIDI 格式的音乐数据，但是更多的歌曲则是常见的 WAV 格式的，因此基于 MIDI 格式的检索方法缺乏实际应用能力。本文采用较复杂的 WAV 格式的歌曲数据作为研究对象，通过对 WAV 格式的歌曲分析进行特征提取，并基于改进的识别检索算法进行哼唱检索系统的设计，实验结果说明该方法的有效性。

关键词：音乐；特征提取，哼唱检索；动态时间归整

1．引言

随着多媒体技术的迅猛发展，人们用计算机存储和管理多媒体信息成为可能，然而现有的信息检索技术还不能有效地满足人们对海量信息的需求，过去的信息是大多数是以离散形式存储在关系数据库中，并通过结构化查询语言(SQL)来进行查询检索，而多媒体数据则是连续的、形式多样的、海量的信息，目前多媒体数据库通常的管理方法是人工的进行基于文本描述的分类和检索，文本描述虽然适用于某些多媒体数据，但是人工操作费力费时，对描述音乐来说是高度主观、不准确和存在误导的。基于内容的技术目的就是为了解决这个问题，它可以分为分类和查询，即利用音乐本身的特征对其进行自动分类，取代手工的文本描述，用哼唱的方法进行查询。先前的音乐检索假设处理对象是 MIDI 格式的音乐数据，但是实际应用这种假设条件很难得到满足，更加常见的则是 WAV 格式的音乐数据，如何检索 WAV 格式的歌曲是本文所要解决的问题，通过对比哼唱歌曲和原歌曲的特征，基于改进的动态归整算法进行相似度计算，从而获得相应的检索结果。

2．相关研究

音乐分类研究的通常方法是从声音信号中提取统计数据，然后利用这些数据进行分类，在基于内容的音乐查询方面，哼唱检索是研究的主要方向之以，但是目前的研

[1] 联系邮件：wwang@se.cuhk.edu.hk，clh-dcs@tsinghua.edu.cn

究绝大部分都在音乐数据库中使用基于乐谱的存储格式（MIDI 格式），很少有研究关注于原始波形数据（WAVE），目前较有规模的项目有以下几种：

英国南安普顿大学的 QBH 系统，近些年来，几乎所有的有关音乐内容搜寻方面的研究报告都引用了英国南安普顿大学在 1995 年 ACM 多媒体研讨会种发表的论文[1]，他们同时开发了一套名为 QBH（Query By Humming）的系统，该系统让使用者可以透过麦克风哼唱来对音乐数据库搜寻。他们通过自相关计算求出输入声波的基频分布图，并用 U（当前音高比前音高高）、D（当前音高比前音高低）、R（当前音高和前音高相等）来表示当前音高和前后音节之间的相对关系，进而形成字符串，但是该系统的缺陷是并未发展出一套完整的音符切割程序，在使用 QBH 时，使用这需要自行切割，但是他们的研究在直觉式歌唱输入音乐搜寻上迈出了重要一步。

新西兰 Waikato 大学的 Rodger J.McNab 和新西兰数字音乐数据库合作开发了一套名为 MT（Melody Transcript）[2][3]的系统，采用 Gold-Rabiner Algorithm 找到输入声波的基频分布，并接着转成标准音符表示。

接着，他们将 MT 结合数字音乐数据库，开发成为一套 MELDEX 的系统[4]，让使用者可以透过麦克风哼唱就直接达到搜寻音乐数据库的目的，正确辨识率约在 77%-89%之间，但是 MELDEX 无法正确地将音符切割开来，因此使用者在哼唱时，在音符与音符之间，必须自行留下小小的简断或多加入“滴答”声音，因此对于非专业人士仍然相当的不便，也相当的不自然。

3. 音乐检索系统结构

音乐查询系统的结构可以用下图表示：

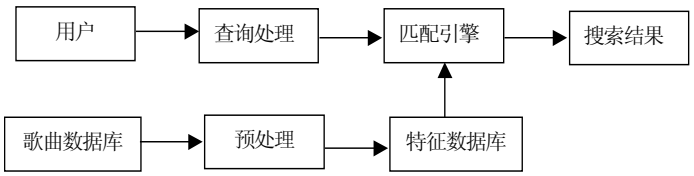

图 1 音乐查询系统的组成

预处理阶段从音乐数据库（可能是各种媒体格式）中提取旋律和节奏两个特征信息（一般是类似于音乐记谱的符号序列表示），保存为结构化的特征数据库，供查询时使用。

查询处理部分主要是将查询者哼唱的输入声音经过类似的处理变化成于数据库相同的特征表示以进行匹配。将原始音乐数据和输入都转换为特征序列表示以后，就可以使用字符串匹配或文本搜索方法进行查询，匹配搜索引擎的作用就是将转换后的输入序列与经过预处理的特征数据库中的特征进行比对，找到可能匹配的结果，并且按照匹配的相似度排序输出结果。

相关研究中多使用 MIDI 格式的歌曲作为处理对象，由于 MIDI 是一种基于乐谱的存储格式，它更加结构化，比未经处理的 WAV 波形数据容易处理，但是实际场合广泛应用的是 WAV 格式的文件，它包含的内容和表现力远远超过 MIDI 格式的文件所能描述的内容。描述 WAV 格式文件的特征成为迫切需要解决的问题，本文中采用底层 MFCC 参数作为其特征描述信息，MFCC 特征在语音识别、音频分类和检索研究领域应用十分广泛，本文在考虑 MFCC 特征的同时也加入了短时能量函数和短时过零率作为特征描述。

4. 检索算法

这里的检索算法采用的是改进的动态时间归整匹配算法，模板匹配法是多维模式识别系统中最常用的一种相似度计算方法，在训练过程中，经过特征提取和特征维数的压缩，并采用聚类方法或其他方法，针对每个模式类别产生一个或记个模板，识别阶段将待识别模式的特征矢量与各模板进行相似度计算，然后判别它属于哪个类。音乐检索也可以用模板匹配法进行相似度计算，但与在语音识别中的典型应用相同，它在特征维数方面存在时间对准问题，是通常模式识别匹配计算时不具备的一些特殊情况，对哼唱检索来说，有人哼唱快，有人哼唱慢，而且在以次哼唱内的每一个音符的长度也不可能完全与原来音乐保持一致，因此在匹配时如果只对特征矢量序列进行线性时间归整，其中的音素有可能对不准，而应该采用某种非线性时间归整模板匹配算法，这在语音识别中已经得到广泛应用。

动态时间归整是采用动态规划方法将一个复杂的全局最优化问题化解为许多局部最优化问题一步步进行决策的。令参考模板特征矢量序列为 $A=\{a_1,a_2,\cdots,a_i\}$，输入特征矢量序列为 $B=\{b_1,b_2,\cdots,b_j\}$，$i\neq j$。动态时间归整算法就是要寻找一个最佳的时间归整函数，使被测模板的时间轴 i 非线性的映射到参考模板的时间轴 j，使总的累计失真量最小。定义最小累计失真函数 $g(i,j)$，表示到匹配点对 (i,j) 为止前面所有可能的路径中最佳路径的累计匹配距离。

下面介绍动态时间归整算法的具体步骤：

（1）初始化：令 $i(1)=j(1)=1, g(1,1)=2d(a_1,b_1)$

$g(i,j)=\begin{cases}0 & (i,j)\in \mathrm{Re}\,g\\ huge & (i,j)\in \mathrm{Re}\,g\end{cases}$，其中 $\mathrm{Re}\,g$ 为约束平行四边形。

（2）递推计算累计距离：

$$g(i,j)=\min\{g(i-1,j)+d(a_{i-1},b_j)\cdot W_n(1),g(i-1,j-1)+d(a_i,b_j)\cdot W_n(2),$$
$$g(i,j-1)+d(a_i,b_j)\cdot W_n(3)\}\ i=2,3,\cdots,I,j=2,3,\cdots,J;(i,j)\in \mathrm{Re}\,g$$

（3） 回溯求出所有的匹配点对：根据上一步的最佳局部路径，由匹配点对 (I,J) 向前一直回溯到（1，1）。这个回溯过程对于求平均模板或聚类中心来说是必不可少的，但在识别过程中往往不用进行。

5. 哼唱检索系统

本实验系统供收集了 84 首流行歌曲作为检索的素材，其中包括英文歌曲 13 首、中文歌曲 71 首，女歌手 14 人，男歌手 32 人的作品，还包括一部分影视歌曲等，原始文件都是 MP3 格式的文件，采样率是 11025Hz、单声道、8bit 的 wav 文件。

用户哼唱时间规定为 10 秒，这已经能够满足一般要求，录制采样率为 11025Hz、单声道、8bit 的 wav 文件。哼唱输入的预处理和分帧的方法与风格分类实验相同，然后进行端点检测，从哼唱输入中提取有效部分：

计算各帧能量 $E_n=\frac{1}{N}\sum_m[S(m)w(n-m)]^2$，其中 $S(m)$ 为输入信号，$w(m)$ 为宽 N 的矩形窗，得到平均能量 $E=\sum_i E_i/m$，以 $g=c*E$（c 为常数）作为静音段或噪声，在提取特征时只计算非静音段。

对哼唱歌曲和原歌曲的对应片段分别提取 MFCC 特征，从对应的包络可以看出相似性，通过实验发现对于 12 维的 MFCC 来说，只有低维系数以及过零率存在较好的相似性。

下面介绍检索系统的处理过程：首先提取计算原始歌曲的 MFCC1~3 和过零率特征。如对于播放时间为 4:30 秒的歌曲“我只在乎你”（许景淳），可以分割成 $\frac{(4*60+30)*11025}{128}=23255$ 帧，每帧计算 MFCC1~3 和过零率共 4 维特征，得到 23255*4 的特征矢量。对 10 秒的哼唱输入则得到 861*4 维的特征矢量。

由于数据量很大，为了提高检索速度和效率，采用如下方法：

```
f(n)为原始特征序列，s(m)为简化后的特征序列
m=0
for (i=1:10:n)
   j=i+10;
   if(mean(f(i:j))>mean(f(1:n)))
```

```
        s(m)=max(f(i:j))
    else
        s(m)=min(f(i:j))
    end
    m=m+1;
end
```

这样，歌曲“我只在乎你”共 23255*4 的特征矢量就可以转化为 2325*4，哼唱特征序列为 86*4，数据量压缩成原来的 1/10，经过这样处理以后的特征序列保留甚至强化了原来的相似性。

然后使用 DTW 算法将长度为 86 的哼唱特征序列与长为 2325 的原始歌曲进行匹配，在匹配时首先标注原始歌曲中每句歌词的开始时间，例如：

[00:27.98]如果没有遇见你

[00.31.26]我将会在哪里

[00.34.89]日子过得怎么样

这在实际的卡拉 OK 演唱带中存储了相应信息，然后在原始特征序列中对应的时间位置截取特征序列与哼唱序列进行 DTW 计算，得到多个距离值。截取特征序列时考虑时间的不等长问题，每个起始点都计算时间长度定为 0.9/1.0（等长）/1.1 等多次距离值，对应上述例子，可以得到每个起始点的距离值数据：

[00:27.98] （0.9）=5，（1.0）=9，（1.1）=13

[00:31.26] （0.9）=42，（1.0）=28，（1.1）=52

[00:34.89] （0.9）=64，（1.0）=55，（1.1）=59

从中取最小的距离值[00:27.98](0.9)=5，认为哼唱与原歌曲中这个位置的原唱最为相似，时间比例为 0.9。对数据库中 84 首歌曲逐一进行这样的计算，得到 84 个曲内最小距离，然后对这 84 个距离进行由小到大的排序，得到相应的结果：

Rank	歌手名字	歌曲曲目
1	许景淳	我只在乎你
2	张学友	偷心
3 …	王菲	半途而废

6. 性能测评

实验系统对 6 位哼唱者的 70 余次哼唱进行了检索测试，在测试时，首先向参试者提供歌曲数据库的曲目清单，由参试者自由选择哼唱曲目，测试的结果统计如图 2 所示。

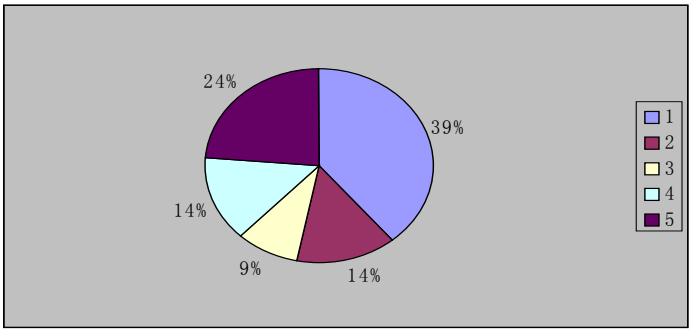

图 2 测试结果

图 2 中 1：rank=8~15；2：rank=2~5；3：rank=1；4：rank>40；5：rank=16~40。这个测试结果与一些基于 MIDI 格式和旋律特征的方法相比要差一些，文献[4]中的 rank=1 的识别率可以达到 82%，但是由于本文处理的是 wav 格式的文件，同时对哼唱者的哼唱没有要求限制（文献[2]要求只能从歌曲的开始部分进行哼唱），因此这个结果是可以接受的，其 rank<15 的概率已经超过了 60%。同时在实际系统中加入了检索结果约束的功能，用户对排序后的检索结果可以根据歌曲的作者、名称、专辑和歌词等文本属性中的关键字进行模糊查找，从而辅助筛选出符合要求的结果。

参考文献

[1]Foote, J. “An Overview of Audio Information Retrieval” Institute of Systems Science, National University of Singapore,1997

[2]许文豪，高名扬，张智星， “直觉式歌唱输入音乐搜寻引擎”.中国台湾清华大学

[3]A.Ghias, J.Logan, D.Chambetlain, B.C. Smith. “Query by Humming-Musical Information Retrieval in an Audio Database”,ACM Multimedia, San Francisco,1995 .

[4]Roger J. McNab, Lloyd A. Smith, Melody Transcription for Interactive Applications, Department of Computer Science, University of Waikato, New Zealand

[5]迟蕙生等 《数字语音信号处理》电子工业出版社，1995

Research on Content-Based Retrieval of Music

Wang Wei* Cai Lianhong**

(Department of Systems Engineering and Engineering Management, CUHK)

(Department of Computer Science and Technology, Tsinghua University)

Abstract: Content-based retrieval of music is one of the most activity research fields. Traditional retrieval assumption object data are formed by MIDI format, but the assumption is unsatisfied with practical request. Most of music is stored in wave format. In this paper, we present an approach to deal with WAV format music files and design a query-by-humming system. The experiment result demonstrates this approach is effectively.

Key words: music, feature extraction, query-by-humming, dynamic time warping