APPLYING CONTENT-BASED RECOMMENDATION TO PERSONAL ITUNES MUSIC LIBRARIES

A Project
Presented to
The Academic Faculty

by

Oliver Jan

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Music Technology in the
School of Music, College of Architecture

Georgia Institute of Technology
May 2010

APPLYING CONTENT-BASED RECOMMENDATION TO PERSONAL ITUNES MUSIC LIBRARIES

Approved by:

Dr. Parag Chordia, Advisor
School of Music, College of Architecture
Georgia Institute of Technology

Dr. Gil Weinberg
School of Music, College of Architecture
Georgia Institute of Technology

Dr. Jason Freeman
School of Music, College of Architecture
Georgia Institute of Technology

Date Approved: 5/10/10

ACKNOWLEDGMENTS

I would like to thank all the professors of music technology for giving me the opportunity to be a part of such an exciting and innovative program at Georgia Tech. Their enthusiasm and imagination have been inspiring. I am also very grateful to have had the chance to collaborate with extraordinary colleagues from MSMT and the CBR group. I will cherish the camaraderie and experiences we had. Finally, I would like to thank my parents for their unwavering love and support.

TABLE OF CONTENTS

ACKNOWLEDGMENTS iii

LIST OF TABLES v

LIST OF FIGURES vi

SUMMARY vii

1. INTRODUCTION 1

1.1 Related Work 1
1.2 Application Overview 3

2. FEATURE EXTRACTION 5

2.1 Timbre 5
2.2 Rhythm 6
2.3 Chroma 7

3. PLAYLIST GENERATION 8

3.1 Ordering Distances 9
3.1.1 Similarity / Dissimilarity 9
3.1.2 Transitions 9
3.2 Metadata Filters 10
3.3 Ranking Features 10
3.3.1 Individual Feature 10
3.3.2 Weighted Combination 11
3.3.3 Cascade 11

4. EVALUATION 12

4.1 User Instructions 12
4.1.1 Computation Time 13
4.2 Results 13
4.3.1 Music Similarity 13
4.3.2 Tempo Accuracy 15
4.3 Discussion 16

5. CONCLUSION 18

5.1 Future Work 18

REFERENCES 19

LIST OF TABLES

Table 4.1: Genre groupings and respective song totals 14
Table 4.2: Neighborhood Clustering..... 14
Table 4.3: BPM Manual Annotation vs. oTunes Calculation..... 15
Table 4.4: Artists with largest number of similar songs (same genre) in playlists..... 16

LIST OF FIGURES

Figure 1.1: Collage of iTunes plug-ins which generate playlists 2
Figure 1.2: Snapshot of oTunes application under the Library tab 3
Figure 1.3: Block Diagram of oTunes System 4
Figure 2.1: Timbre Feature Extraction and Texture Window [7] 6
Figure 2.2: System Architecture of BeatRoot [8] 7
Figure 3.1: Snapshot of oTunes application under the Playlist tab 8
Figure 3.2: Individual Feature Selection Box under Rankings tab in oTunes 10
Figure 3.3: Weighted Combination Box under Rankings tab in oTunes 11
Figure 3.4: Element Reordering Box under Rankings tab in oTunes 11

SUMMARY

As personal digital music collections grow larger and larger, it becomes more difficult to choose what to listen to; it's unlikely that you remember every song you have and its metadata can only take you so far. By analyzing the audio waveform itself, musical similarities between songs can be made in order to provide personalized recommendations from a chosen seed song. This project presents a Windows application called "oTunes" which makes such recommendations from timbre, rhythm, and pitch-related features extracted from each song as well as user input regarding how the features should be utilized, and generates content-based playlists in iTunes.

1. INTRODUCTION

With the advent of the mp3 format for audio storage and compression, personal computing and portable mp3 players, personal music collections have migrated to the digital domain. Yet the methods of selecting what music from one's own library to listen to (and continue to listen to) have remained analogous to physically choosing a CD or vinyl record from a record store to play; music is still primarily chosen by genre, artist or album and people tend to continue listening to the same genre, artist, or album. But music can be characterized in a myriad of ways and music information retrieval (MIR) aims to recognize and extract some of these fundamental characteristics, such as timbre and tempo for example. Using MIR techniques can establish musical relationships between songs beyond its textual metadata and given a song as an example, similar songs can be suggested from these relationships and put into a playlist. As digital music libraries grow in size and diversity, these content-based recommendations can help enhance a user's experience with their own music library.

1.1 Related Work

In the past decade, music information retrieval has spawned into an interdisciplinary research field that is developing new ways of interacting with music and facilitating the organization and retrieval of music. Sony Computer Science Laboratory's MusicBrowser was among the first systems to search a collection of mp3s based on its content by incorporating automatic tempo analysis and timbre similarity; Aucouturier and Pachet suggested that timbre is a means to identify songs and represent musical taste [1]. Tzanetakis and Cook also introduced approaches to analyzing beat and tempo [2] that are used to detect songs that sound similar.

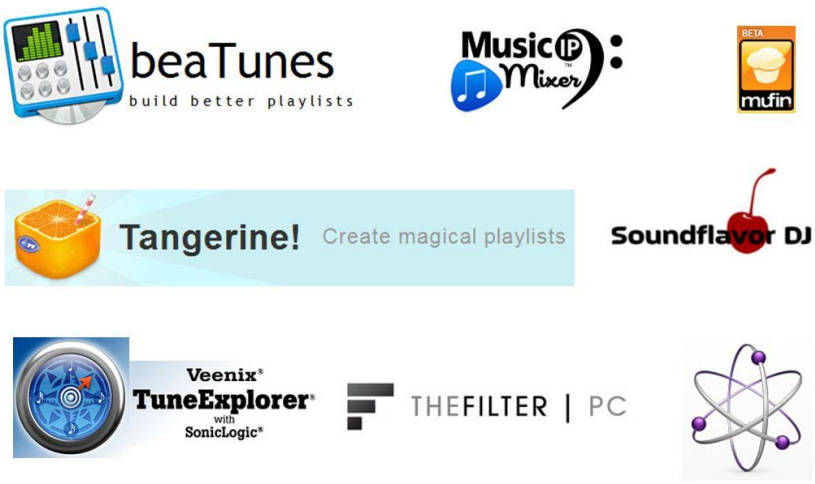

Figure 1.1: Collage of iTunes plug-ins which generate playlists

Tangerine!, an iTunes plug-in, generates playlists based on tempo (BPM and beat intensity) analysis; beaTunes also utilizes beat detection, as well as key detection and color determination to build playlists. Mufin creates playlists based on the musical similarities between songs in an iTunes library from a 3-5 second fingerprint of each song. Over forty musical characteristics are analyzed in order to create each fingerprint. MusicIP Mixer uses a fingerprinting system and suggests songs after matching their analyzed fingerprints to its database. SoundBite, from the Centre for Digital Music at Queen Mary, University of London, creates playlists from similar sets of forty features that describe different characteristics of the music.

There are two features already built into iTunes that allow the user to generate automated playlists. The “Smart Playlists” feature lets the user construct a set of criteria defined by metadata, automatically filters the library based on the criteria and puts the returned songs into a playlist [3]. The Genius feature utilizes the song rating system, as well as the listening and purchasing habits of other iTunes users, to select songs to be included in a “Genius” playlist based on a seed song. While this collaborative filtering approach performs similarity tasks satisfactorily [4], it relies on correct metadata and is unable to make recommendations if the music is not in their system. There is also always

a popularity bias inherent to any social system which can skew the representation of the top-played and top-selling songs and artists, and as well as those making up the long tail by not being played as often and not selling as well [5].

1.2 Application Overview

This application, oTunes, analyzes the contents of songs in one's personal iTunes music library and aims to provide a means for users to generate different types of playlists based on their own preferences. In this regard, oTunes gives the user control over how the extracted features from each song are utilized to populate a playlist, which differs from most other content-based plug-ins for iTunes that predefine how the features are used for recommendation or incorporate social metadata from external sources. Since the process of creating a playlist usually entails picking out a song as an example [6], a query by example recommendation model is implemented here given a seed song selected from a list of tracks loaded into oTunes by the user. This is unique among iTunes applications because the user does not need to switch to the iTunes window to select a seed song.

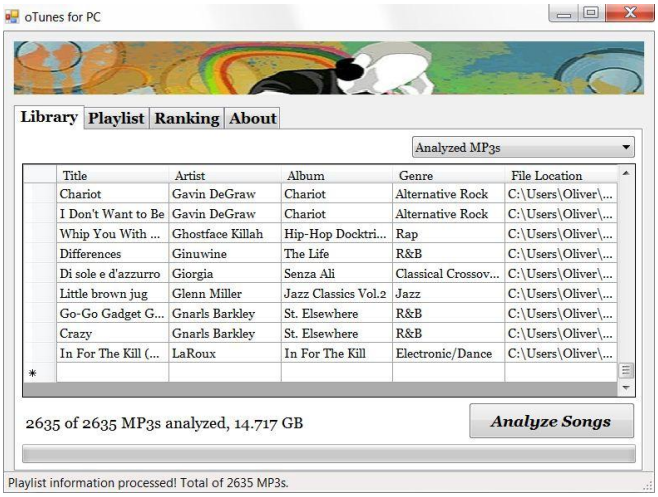

Figure 1.2: Snapshot of oTunes application under the Library tab

The C# Windows Form Application was used to design and implement the GUI of oTunes because it is an event-driven application that allows for rapid development and prototyping of the user interface in Windows. oTunes communicates with iTunes through the iTunes COM Interface for Windows SDK and queries a SQLite Database that stores each analyzed track's information, including representations of extracted features, and the feature distances between tracks. Feature extraction is done using the MARSYAS open source software framework built in C++, which was wrapped for use by C# and integrated into the oTunes application. MARSYAS was chosen to build the MIR system because of the many integrated building blocks for MIR tasks it supports and also because of its relatively fast run-time performance.

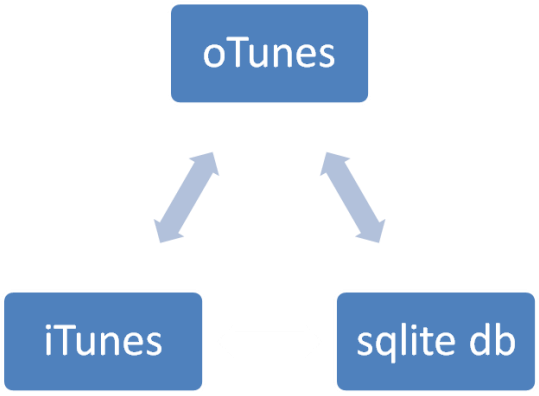

Figure 1.3: Block Diagram of oTunes System

2. FEATURE EXTRACTION

The preprocessing component of oTunes is the feature extraction, which attempts to characterize a particular segment of audio by representing it more abstractly in order to compute similarities. This application extracts three of the most fundamental musical features (timbre, rhythm, and pitch-related chroma) from each song that are commonly used in MIR.

2.1 Timbre

The overall tone quality or color of a sound is known as its timbre, and it is a common and proven descriptor of music information retrieval tasks. Mel Frequency Cepstral Coefficents (MFCCs) are used to capture the timbral quality of a song because they represent its spectral shape and mimic its input to the human auditory system by mapping its frequencies to the Mel Scale [1]. Thirteen MFCCs, along with the spectral centroid, rolloff, and flux, make up the feature vector that is extracted from each clip of audio. The spectral centroid and rolloff are also both measures of spectral shape, and the spectral flux is a measure of the amount of local spectral change [2].

The final 64-dimensional feature vector is constructed from the means and variances taken across "texture windows" whose length corresponds to the minimum amount of time required to identify a particular music "texture" that characterizes the overall timbre [7]. In this implementation of George Tzanetakis' algorithm, the length of the texture window is 44 frames, or approximately one second, and 30 seconds from each song is analyzed after a 60-second delay. This algorithm was chosen for oTunes because it performed well at the annual Music Information Retrieval Evaluation eXchange (MIREX) in 2007 and it was much simpler to implement in MARSYAS.

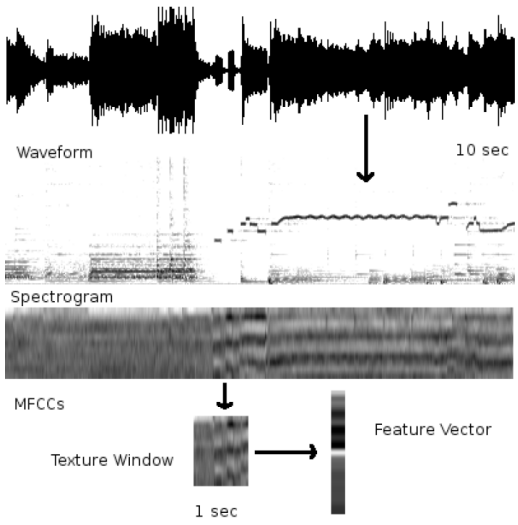

Figure 2.1: Timbre Feature Extraction and Texture Window [7]

2.2 Rhythm

Rhythm can be defined as the movement or variations of patterns through time, and people have always been able to innately tap their feet, snap their fingers, or clap their hands to the beat of the music. Identifying this basic rhythmic pulse of a song can be accomplished by modeling the perception of beat through tempo induction and beat tracking. oTunes utilizes an implementation of the INESC Beat Tracker (IBT) on the MARSYAS platform that is based on the BeatRoot system created by Simon Dixon. This beat tracking system was chosen because it performed well at MIREX in 2006 and it was included in the MARSYAS package. Tempo is induced from clusters of inter-onset intervals (IOI) that are calculated from onset times and agents are assigned tempo hypotheses to match predicted with actual events [8]. These beat agents are created and killed according to the beat adjustments made around each prediction. A beat "referee" determines which beat agents have highest scores, and the result is converted into beats per minute (bpm). The induction time is set in oTunes to be five seconds and ten seconds from each song is beat-tracked after a 55-second delay.

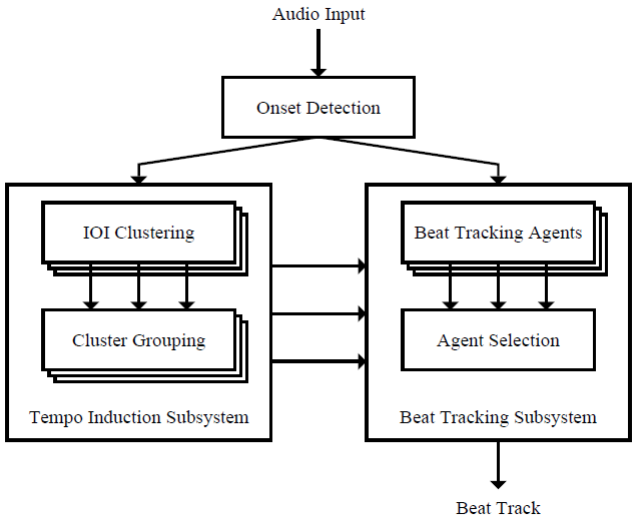

Figure 2.2: System Architecture of BeatRoot [8]

2.3 Chroma

Chroma is the quality of pitch or in simpler terms, what pitch sounds like. Pitch can be characterized as the perceived fundamental frequency of sound; pitches that belong to the same pitch class are said to have similar quality, or chroma. These pitch classes can be represented by chroma features extracted from the spectrum of an audio signal. The algorithm used by oTunes for chroma representation takes this spectrum and then finds the salient frequencies and pitches. These pitches are assigned to the closest pitch class on an equally-tempered western scale, and the pitch classes are normalized and folded into one octave. A histogram is then taken based on the chromatic bin and it is returned as a 12-dimensional chroma profile [9]. This algorithm by Matthias Varewyck implements the conversion of an amplitude spectrum to a chroma profile and was chosen for oTunes because it was an existing class on the MARSYAS platform.

3. PLAYLIST GENERATION

Playlists are generated by returning the closest songs based on ordering and ranking criteria of data extracted from timbral, rhythmic, and chroma features. Songs that share similar musical characteristics of timbre, rhythm, and pitch quality respectively can be construed as similar, and this serves as the basis from which recommendations are made; if someone likes one song, it is likely that they will like a similar song. While solely basing song recommendations on this assumption can lead to problems of too much similarity and not enough novelty for example, oTunes attempts to address some of these pitfalls by incorporating additional options for the user to choose and additional parameters for the user to define.

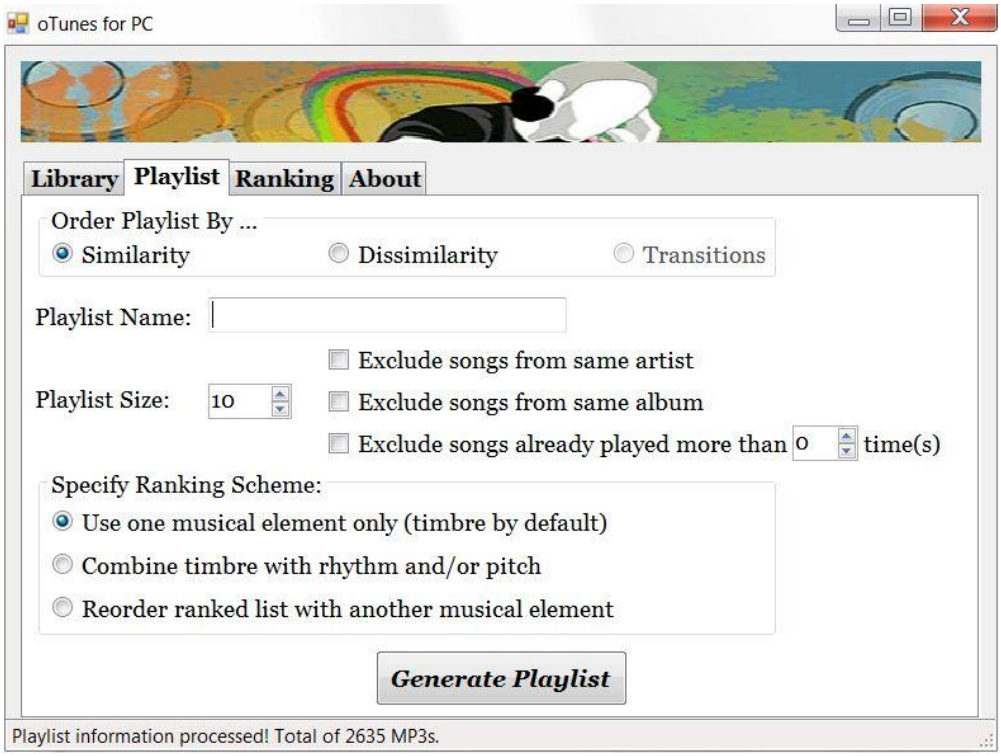

Figure 3.1: Snapshot of oTunes application under the Playlist tab

3.1 Ordering Distances

The feature extraction generates a table in the oTunes SQLite database which stores the unique id that iTunes assigns each track, along with the associated artist, album and playcount, the 64-dimensional timbral feature vector, the bpm, and the 12-dimensional chroma profile. Feature distances are calculated between each analyzed track by simply taking the Euclidean distance between the corresponding values in each feature representation. These distances between pairs of songs for each feature are stored in another table in the database and normalized to generate a z-score that expresses the similarity between two songs, in essence creating a similarity matrix. A playlist is then generated from querying this matrix for the feature distances from the seed song to all other songs in the database. How this database query is formulated depends on what the user specifies in oTunes.

3.1.1 Similarity / Dissimilarity

If the user chooses to order the playlist by "Similarity", oTunes simply returns the shortest distances between the seed track and rest of the analyzed tracks. "Dissimilarity" on the other hand returns the largest distances and gives the user a chance to experience a playlist made up of songs that are furthest away from the selected seed song in terms of its feature representations.

3.1.2 Transitions

Playlists can also be generated more like "mixes", where the order of the songs matter [6], by analyzing the beginning and ending of each song with content-based recommendation techniques. To create a playlist based on "Transitions" given an initial seed song, the first song of the mix will be the seed song. The ending of the seed song

will be compared with the beginning of the rest of songs, and the song's feature representation with the closest distance will be chosen for the next song in the mix. Then the ending of that song will be compared with the beginning of the rest of songs, and so on until the specified length of mix is reached.

3.2 Metadata Filters

To improve the diversity, novelty, and reach of potential playlists, there are three checkboxes that allow the user to exclude or filter out songs from the same artist, songs from the same album, and songs already played more than X number of times. If X = 0, essentially that means all songs that have been played before (playcount = 0) will be excluded from generated playlist.

3.3 Ranking Features

To give the user the ability to decide what features of each song to use and how to use them when generating playlists, three options are presented that enable the user to produce recommendations in a myriad of ways based on their own criteria.

3.3.1 Individual Feature

Figure 3.2: Individual Feature Selection Box under Rankings tab in oTunes

This ranking scheme simply gives the user the option to rank the distances calculated from one feature representation only. None of the other features will be utilized.

3.3.2 Weighted Combination

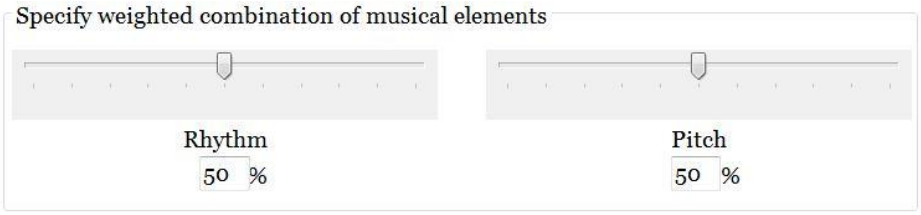

Figure 3.3: Weighted Combination Box under Rankings tab in oTunes

This ranking scheme allows the user to make weighted combinations of rhythm and pitch features with timbre, which is always set at 100%. In order to do the linear combination of the features, the distances of each feature are standardized by computing the z-score, or the mean of all the distances subtracted from raw distance divided by the standard deviation. In order to avoid recalculating the mean and standard deviation for all the distances every time a new track is analyzed, the means and variances are updated incrementally.

3.3.3 Cascade

Figure 3.4: Element Reordering Box under Rankings tab in oTunes

This ranking scheme enables the user to specify one feature and the number of songs returned (based on that feature) to reorder, and to sort by another feature and specify the number of songs to output (the playlist size), i.e. sort one musical element by another musical element.

4. EVALUATION

In this section, a primer on how to use oTunes will be provided, as well as the estimated computation times for song preprocessing. Music similarity and tempo accuracy will be evaluated and the corresponding results presented and discussed.

4.1 User Instructions

To download the oTunes, please go to the website below to save and extract the zip file to your computer. Open the readme file for more setup information.

http://www.oliverjan.com/otunes

Once the application is launched, your "Music" library information will load into the oTunes window. Before being able to generate playlists, you will obviously have to analyze some songs: to start analyzing your entire music library, simply click on the "Analyze Songs" button under the Library tab. You can also load in other libraries or playlists for song analysis by using the dropdown menu. You can cancel and resume analysis at any time.

To make a playlist, first select a seed song (make sure the track from a loaded iTunes playlist or library under the Library tab is highlighted in the oTunes grid) and then choose from the ordering, exclusion, and ranking options under the Playlist tab. Remember to name your playlist by filling in the appropriate field and specify the number of songs you want in your playlist. You can change the settings for the ranking schemes under the Ranking tab. Once you are ready to make a playlist, simply click on the "Generate Playlist" button and switch to your iTunes window to enjoy your new playlist!

4.1.1 Computation Time

As of the alpha release of oTunes, it takes a relatively long amount of time to extract three types of features, calculate the distances, and calculate or re-calculate the z-scores. Because of the substantial computation time, especially for preprocessing (about 15 seconds per song), this program would not be feasible for most people with large iTunes libraries to use unless they have a computer with sufficient computational power to populate the database in a reasonable amount of time. The best strategy to use this program in its current stage is to analyze songs when you can and slowly build up your oTunes database until you have enough songs to make meaningful playlists from.

4.2 Results

In order to begin to evaluate oTunes and the playlists that it generates, it is necessary to evaluate the effectiveness of the feature extraction. In the next two sections, the results from evaluating timbre and rhythm are presented.

4.3.1 Music Similarity

The idea of music similarity is at the core of most, if not all, music recommendation schemes. What makes songs "sound alike" can be attributed to its global timbre quality [1] and while this is subjective to each individual listener to some degree, an evaluation of timbral similarity can utilize available metadata to make plausible assumptions about the similarity between songs. This evaluation is based on the notion that two songs by the same artist are likely to be similar and under the assumption that songs within a certain type of genre generally have similar timbres overall.

From a dataset of 2635 songs, a hundred songs were selected to be seed songs and a hundred playlists were created from those seed songs using the following oTunes

settings: order playlist by similarity, playlist size of 21 (including seed song), use default timbre musical element only. All the genre types found in these 100 playlists were subdivided into fifteen groups of related genres in order to yield meaningful results from this relatively small sample set of songs.

Genre Groups	# of songs
Alternative Rock, Progressive Rock, Grunge	478
Classic Rock, Rock, Rock & Roll	385
New Wave, Ska, Punk Rock, Latin Rock, Mambo	98
Rap Metal, Heavy Metal	33
Indie, Avantgarde	88
Funk, Electronic/Dance, Disco	168
Hip-Hop, Rap	618
Trip-Hop, Electronic	88
R&B, Soul	309
Pop, Pop (M), Pop/Rock (F), Taiwanese (Pop)	120
Oldies, Doo Wop, Song	21
Classical Crossover, Soundtrack, Instrumental, New Age, World, World Fusion	58
Jazz, Blues	100
Country, Folk	100
Reggae	22

Table 4.1: Genre groupings and respective song totals

Given the seed song's labeled genre, reoccurring genres from the same genre group were tallied for the top 5, top 10, and top 20 (entire returned playlist) results. Then the average percentage of the top N results for each query in the dataset with the same genre was computed after normalizing the number of examples of each genre available in the dataset.

	Top 5	Top 10	Top 20
Genre %	54.40%	49%	48.20%

Table 4.2: Neighborhood Clustering

4.3.2 Tempo Accuracy

Using the same hundred songs from the previous evaluation, the perceptual tempo of each song was manually annotated by tapping the beats per minute (bpm) and compared to the calculated tempo rate from oTunes beat tracking. The tempo was deemed correct if it was within 5% of the manual annotation, including doubling (within 10%) of the tempo. The results yielded an error rate of 25%, which is comparable to the MIREX 2006 results for tempo extraction.

title	bpm	tempo	match	double
takeonme	85	83.1	V	FALSE
fernando	100	195.7	1.957	TRUE
backinblack	93	181.8	1.9548	TRUE
daylight	92	189	2.0543	TRUE
fallin	193	192.6	V	FALSE
biginjapan	98	190	1.9388	TRUE
foreveryoung	68	149	2.1912	FALSE
1thing	100	205.5	2.055	TRUE
houserisingson	241	229	V	FALSE
heartbeat	128	163	1.2734	FALSE
operator	178.5	174.4	V	FALSE
rain	137	135.4	V	FALSE
bodymovin	102	90.9	0.8912	FALSE
yesterday	95	217.8	2.2926	FALSE
rain	112	211.1	1.8848	FALSE
michelle	119	117.5	V	FALSE
stayinalive	104	203.7	1.9587	TRUE
brick	97	193.2	1.9918	TRUE
beautiful	93	89.7	V	FALSE
rocky	97	94.9	V	FALSE
aintnosun	80	149.2	1.865	FALSE
ironman	77	144.4	1.8753	FALSE
definition	180	180.7	V	FALSE
redefinition	180	175	V	FALSE
dothismyway	174	166	V	FALSE
whatsmyage	156	170.4	1.0923	FALSE
shomechicago	126	168.1	1.3341	FALSE
onelove	152	153.9	V	FALSE
prayer	123	118	V	FALSE
feeling	110	216.6	1.9691	TRUE
motown	114	217.1	1.9044	TRUE
toxic	143	154.4	1.0797	FALSE
star	87	86.1	V	FALSE
swallowed	89	88.6	V	FALSE
nude	119	201.7	1.695	FALSE
forever	120	117	V	FALSE
johnny	167	163.1	V	FALSE
alovee	70	214.9	3.07	FALSE
alovet	69	209.9	3.042	FALSE
heaven	90	183.2	2.0356	TRUE
iceblink	102	212.7	2.0853	TRUE
pandora	159	186.6	1.1736	FALSE
yellow	164	171.5	V	FALSE
inmyplace	72.5	139.4	1.9228	TRUE
go	105	200.6	1.9105	TRUE
walkingaway	86	170.6	1.9837	TRUE
jamrock	76	74.7	V	FALSE
something	100	86.9	0.869	FALSE
sandstorm	136	132.9	V	FALSE
takefive	157	152.9	V	FALSE
oncedecember	130	127.5	V	FALSE
soulmeets	128	125.8	V	FALSE
sweetlull	87	173.5	1.9943	TRUE
whilearth	120	238.7	1.9892	TRUE
travelman	91	175	1.9231	TRUE
stilldre	94	91.7	V	FALSE
forgotdre	134	132.9	V	FALSE
beauty	60	118	1.9667	TRUE
ordinary	70	136.3	1.9471	TRUE
desperado	121	226.8	1.8744	FALSE
returntoi	176	172.9	V	FALSE
rain	122	80	0.6557	FALSE
ashes	138	133.6	V	FALSE
atlast	175	210.1	1.2006	FALSE
rockafella	153	149.6	V	FALSE
everlong	159	157.9	V	FALSE
differences	63	61.7	V	FALSE
deadpres	176	171.7	V	FALSE
allmylife	129	124.4	V	FALSE
letsgo	96	153.8	1.6021	FALSE
crazy	112	108.7	V	FALSE
mambo	174	171.5	V	FALSE
letsgetit	84	163	1.9405	TRUE
everything	163	178.8	1.0969	FALSE
entersandman	122	233.5	1.9139	TRUE
thewayyou	115	112	V	FALSE
tolovesome	91	178.7	1.9637	TRUE
heartshaped	100	204.4	2.044	TRUE
et	96	187.8	1.9563	TRUE
bdisaster	86	163.8	1.9047	TRUE
allmine	120	212.6	1.7717	FALSE
brass	99	190.6	1.9253	TRUE
purplerain	117	225.6	1.9282	TRUE
giveit	122	116.3	V	FALSE
wishing	95	185.5	1.9526	TRUE
smooth	119	221.8	1.8639	FALSE
isitcrime	82	95.8	1.1683	FALSE
cupid	122	242.1	1.9844	TRUE
aida	76	73.2	V	FALSE
notears	82	135.6	1.6537	FALSE
yourestill	67	65.4	V	FALSE
rain	70	67.8	V	FALSE
madworld	120	234	1.95	TRUE
headover	95	92.6	V	FALSE
bittersweet	86	83.4	V	FALSE
careless	78	148.5	1.9038	TRUE
promise	119	233.9	1.9655	TRUE
gonetilnov	86	178.9	2.0802	TRUE
gunpowder	147	142.7	V	FALSE
wetryin	99	184.9	1.8677	FALSE
TOTALS	100		42	33

Table 4.3: BPM Manual Annotation vs. oTunes Calculation

4.3 Discussion

The neighborhood clustering results for genre was expected given the MIREX 2007 results of the same algorithm; over half of the top five songs returned in timbral similarity playlists on average were of the same genre as the seed song and the percentage for top ten and twenty songs were all about the same. In many cases, the same artist was also present in playlists and some even returned songs from the same album.

Artist	Top 5	Top 10	Top 20	Genre of Artist
wyclef	5	10	20	Hip-Hop
mjblige	5	10	17	R&B
drdre	5	10	16	Rap
scarface	5	9	18	Hip-Hop
blackstar	4	9	15	Hip-Hop
coldplay	4	8	17	Alt. Rock
blink182	4	8	14	Alt. Rock
drdre	4	7	12	Rap
sade	4	6	8	R&B
verve	4	5	10	Alt. Rock

Table 4.4: Artists with largest number of similar songs (same genre) in playlists

The labeling of genres is obviously very subjective and nowadays with the fusion of multiple genres and new emerging genres, it is difficult to categorize songs. But even listening to top returned songs of different genres, most do sound like the seed song. For example, "Desperado" by the Eagles is a softer classic rock song with piano and string instrumentation and although it returned no classic rock genre-labeled songs, it did return similar sounding songs with piano and strings (Randy Newman, Coldplay, Yanni, R Kelly). And also some test cases using cover songs or two nearly identical songs (i.e. Coco Lee singing the same exact song from the Crouching Tiger Hidden Dragon soundtrack, but just in two different languages) yielded the expected song as its nearest match.

A personal subjective and qualitative assessment of the feature combination was also made. Combining the timbre feature with percentages of rhythm and pitch-related chroma yielded some eclectic results that usually made sense on some level. For example, choosing just timbre when generating a playlist based on Dave Brubeck's classic jazz standard "Take Five" returned another song of his, "Pick Up Sticks," as the top hit not surprisingly. Both of these songs have a distinctive percussive texture (cymbal and hi-hat are prominent), along with background piano and solo trumpet. However, it was discernable that both songs had different tempos and were in a different key. By combining timbre with 50% rhythm and 50% pitch, oTunes returned two completely different songs in terms of artist and genre. The top hit was "The Beautiful Ones" by Prince and the Revolution; while this song shared some of the same timbral qualities as "Take Five", such as piano and hi-hat, their perceptual tempo was the same. The second returned song was "Strange Fruit" by Common, featuring John Legend, and this song had a chorus that shared some of the same notes as the melody from "Take Five." In this specific case, oTunes was able to recommend similar songs based on all three features extracted.

5. CONCLUSION

If a song is already in your library, it can be assumed that you already "like" it, so music recommendation in the context of your own library should also be focused on the ways you would like to listen to your music and what you would like to listen to at a particular moment in time. The goal of oTunes is to give the user opportunities to interact with his or her music library in ways that go beyond sorting or selecting songs by metadata. By providing transparency and flexibility in determining how to generate playlists, oTunes also gives the user the freedom and the chance to experiment with different combinations and sorting permutations.

5.1 Future Work

There is much that can be done and needs to be done in order for oTunes to continue developing and perhaps have a beta release. Optimization of computation time, fixing memory leaks, and threading improvement are critical software issues that need to be addressed sooner or later. Additional functionalities in the future could be the ability to handle multiple databases and to utilize the sharing function of iTunes in order to run oTunes recommendations on other libraries in the network. Also giving the user control of feature extraction parameters can allow for DJ's to analyze more specific portions of songs. And implementing better feature extraction algorithms or improving the existing ones will be necessary as the MIR field continues to advance. Perhaps one day in the near future, there will be no need for oTunes when iTunes incorporates an integrated content-based recommendation!

REFERENCES

[1] Aucouturier, J. and Pachet, F. (2002). Music similarity measures: What's the use? In Proceedings of 3rd International Society on Music Information Retrieval Conference.

[2] Tzanetakis, G., and Cook, P. Musical genre classification of audio signals. In IEEE Transactions on Speech and Audio Processing (2002), pp. 293--302.

[3] Cunningham, S., Jones, M., and Jones, S. (2004). Organizing digital music for use: An examination of personal music collections. In Proceedings of the 5th International Society on Music Information Retrieval Conference.

[4] Barrington, L., Oda, R., Lanckriet, G. (2009). Smarter than genius? Human evaluation of music recommender systems. In Proceedings of the 10th International Society for Music Information Retrieval Conference.

[5] Celma, O. (2008). Doctoral Thesis: Music Recommendation and Discovery in the Long Tail. Pompeu Fabra University, Spain.

[6] Cunningham, S. J., Bainbridge, D., and Falconer, A. (2006). More of an art than a science: Supporting the creation of playlists and mixes. In Proceedings of the 7th International Society for Music Information Retrieval Conference.

[7] Tzanetakis, G., "Marsyas submissions to MIREX 2007," in MIREX at 8th ISMIR Conference.

[8] Dixon, S., "Mirex 2006 audio beat tracking evaluation: Beatroot," in MIREX at 7th ISMIR Conference.

[9] Varewyck, M., Pauwels, J., Martens, J.-P., "A novel chroma representation of polyphonic music based on multiple pitch tracking techniques", Proceedings of the ACM International Conference on Multimedia, Vancouver, BC, Canada, 2008.

[10] Pampalk, E. (2006). Doctoral Thesis: Computational Models of Music Similarity and their Application to Music Information Retrieval. Vienna University of Technology, Austria.