ASHESI UNIVERSITY COLLEGE

DEVELOPING A CONTEXT-AWARE AUTOMATIC PLAYLIST GENERATOR (CAAPG)

By

HABEEB BIDEMI AREMU

Dissertation submitted to the Department of Computer Science

Ashesi University College

In partial fulfilment of Science degree in Computer Science Administration

April 2014.

I hereby declare that this dissertation is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature: ……

Candidate's Name : Habeeb Bidemi Aremu

Date : 04/18/2014

I hereby declare that the preparation an presentation of the dissertation were supervised in accordance with the guidelines on supervision of dissertation laid down by Ashesi University College.

Supervisor's Signature: ……

Supervisor's Name : Dr. G Ayorkor Korsah

Date : 04/18/2014

Acknowledgements

The road delivering this paper was a long and arduous one. I would first thank my parents who have supported my education up to this level. I will also thank my friends who kept on giving me pressure to put in my best and helped peer review my work.

Most of all though, I would like to thank my supervisor who had to endure nights of understanding my poor writing, but still did not mind making 133 corrections in one night.

Abstract

The current digitization of music and the sheer volume of the musical content available to listeners on local devices, such as mobile phones and IPod has been revolutionary. This trend has changed the way humans interact and experience their music. Music listeners can listen to their songs on the move. The most recent trend in the music industry is that users can organize and search for their songs based on emotions. However, most users have to manually create their playlists for particular situations. The work that this entails is cumbersome and sometimes negates the experience of the listener. The intuitive response to this problem is developing an automatic playlist generating (APG) system. Research on APG mostly focuses on using traditional metadata and audio similarity methods to create a playlist. In addition APG is seen as a static problem [1]. This means that APG is seen as a problem that does not change, however music listeners are always changing their listening habits.

This thesis supports and follows from the argument made in Chi chung-yi's work - that the APG problem is a continuous optimization problem. Additionally, in this paper I also argue that the best way to give users a good listening experience is to understand the user's preference(s) depending on the context. Context here simply mean the features that make up the environmental space in which the system is being used. The main idea in this paper is to show the importance of emotional categorization in the generation of playlist content, while simultaneously mapping

those categories to the user's context based on the users past activities on the system.

Reinforcement learning is the method used in this thesis to generate a personalized playlist, based on the context of use and the user's emotional preference. After implementing the system we use two hypothetical users to simulate the use of our system. Various metrics are defined to measure the performance of this approach.

Table of Contents

Chapter 1: Introduction

Recent rapid advancements in technology have affected many industries; it can be argued that none more than the music industry. After Steve Jobs' declared, at the unveiling of the IPod, that a thousand songs could fit in your pocket, the music industry was never the same. Music playing devices became smaller, more portable and could hold more music. The rise of such huge music libraries led to a greater need for playlist generation. This paper presents a system that generates a personalized playlist on a mobile smart phone. The system presented here is called the Context-Aware Automatic Playlist Generator (CAAPG). This system combines the principles of mood classification models, contextual information and user feedback to achieve the goal of creating a personalized playlist. In this work, contextual information includes the day of the week, the time of day, and the movement of the user. CAAPG allows users to experience their music in an exciting new way, discover lost songs in a purpose-driven manner and reuse playlists without the feel of routine.

CAAPG combines two key components to achieve this goal. The first is the classification of mood using already established models and the second is the use of contextual information to determine a user's mood. In recent works, classification of songs into proper mood categories has taken center stage in the building of playlist generation tools. Online music giants like Echonest and last.fm have invested heavily in playlist generation tools that use mood classification systems.

Additionally, there have been substantial amounts of literature written about the use of contextual information in generating playlists. Wang and Rosenblum present a system that uses low-level sensors to determine user activity for song recommendation. There exists a lot of literature proposing the use of contextual information for playlist generation , however not many systems have been implemented using this method, furthermore none have used reinforcement learning as tool to solve the problem of automatic playlist generation (APG). Therefore, I believe using contextual information together with a traditional mood classification system such as Russell's Valence-Arousal categorization, to generate playlists is a novel idea; one which will improve the personal nature of a generated playlist and the music experience of users.

Before going into further details, some recurring terms that will be featured in this research are explained in the following glossary.

- ***Purpose-aware*** means the proposed system should be able to recognize "user defined goals (purpose)" [2]. This means goals are set by users who expect the system to help them achieve the set goals. For example, If a user feels stressed and needs to calm his nerves, he would want to play a song that suits this purpose.
- The term '***context of use***' means the real life situation in which the system is being used [2]. For example, if a user was jogging while using the system, the act of jogging would be the context of use.
- A word often used by many but which still needs clarification for the purpose of this research paper is ***mood***. Moods, as defined by L.sizer in [3] are

generalized, nonspecific affective states like melancholy, ennui or ebullience. In much simpler language, mood is the emotional state of a user. L.sizer [3] defines affective states as a spectrum of different types of emotional experiences, where the experiences range from "paradigm emotion states" at one end to moods at the other. In [3] there is a distinct difference between what the paper calls a 'paradigm emotion state' and mood. A more detailed definition of mood and its difference from "paradigm emotion states" will be further discussed in the literature review.

As already stated, the advent of portable music players and the growing size of digital music libraries have revolutionized the music industry. This revolution is not only limited to the music industry, but also extends to the users. Music is now a major feature in our everyday activities. This has led to the many efforts made towards the advancement of personalized playlists. This is why the proposed system from this research includes a user feedback component, to enable the user give his/her playlist a personal touch.

There are a variety of ways in which an APG system, such as the one proposed by this thesis, can be applied. Playlist generation systems can be applied to scenarios such as exercising, where music is used to motivate and push users to achieve a particular goal in a period of time. They can also be applied by students who wish to create a study playlist. Scenarios such as those explained above rely heavily on the mood of the user and the affective mood the song puts the user in. The mood of the user is the emotional state a user is currently experiencing, not necessarily from external factors, while 'affective mood' is the emotional state caused by

external factors such as playing a song. CAAPG will support users who need to put themselves in a particular mood or who want to maintain a particular mood over a period of time.

The exponential increase in digital music over the years, has led to users storing more songs on local libraries. This results in a phenomenon that has come to be known as "The Long Tail" [4], "which describes the statistical distribution of a high-frequency population that immediately trails off to a low-frequency population" [4]. Simply put, due to the large number of songs, there exists a subset of songs that are played more often, while the rest are hardly ever discovered. Many music recommendation systems have focused on retrieving these long tail songs; this process is described as music recovery. The system developed by this thesis also looks to address the problem of music discovery.

This paper is organized as follows: The second chapter explores and analyzes some of the research related to CAAPG. In addition, it takes a brief look at some of the seminal papers in the mood and music literature, which are mostly in the field of psychology. Hence, through the review of existing literature, an attempt will be made towards evaluating the feasibility of the proposed system. In the third chapter, I present some of the theories that led to the research design decisions. An in-depth explanation of how the different components of this system are implemented, together with how the system evaluated is given below.

Chapter 2: Literature review

In the field of personalized context-aware and purpose-aware playlist generation, several systems have been proposed to easily differentiate the listening context and the purpose of a playlist. The connection between music and mood cuts across several fields such as mathematics, psychology and philosophy. In this chapter, a discussion and analysis of the literature on mood classification models, context aware systems and reinforcement learning is carried out. This section will review some of the proposed and implemented playlist generation systems with respect to their relevance to this project. This section will also look at proposals from different fields such as machine learning that directly affect this project.

Context-Aware systems

This section reviews and analyzes systems that have been implemented using contextual information in building a playlist generation tool.

Gordon Reynolds discusses a "design proposal to further the research area of context-aware and emotion-aware music devices. Of particular interest is how environmental data may be used to infer a listener's mood and how such information may integrate into the process of automatically generating a music playlist" [5]. The system proposed here should work without user interaction. When a listener wants to listen to a song, the system, which has already been trained

with the chosen environmental features, will analyze the listening environment using sensors. Environmental data will be matched and assigned to appropriate audio features [5]. The song library of the user is then filtered using the required features and a selection algorithm will generate an appropriate playlist.

Gordon argues that there are strong links between an individual's environment and the individual's attitude. Gordon justifies using environmental features to determine mood, by using the foundations of attitude theories formulated in the 1930s. Although the system was not implemented, The paper considers 7 environmental features, which include time and date, weather, lighting conditions, humidity conditions, temperature conditions, noise level and listener's activities. One of the core aspects of CAAPG is that it uses similar environmental features such as those presented above to determine mood. The methodology section will justify why each of these particular environmental features was selected.

PAPA system

Nuria and Lucas present physiology and purpose-aware (PAPA) framework [6]. This framework uses the physiological responses with traditional song metadata to automatically generate a playlist that will aid the user in achieving user-defined goals. This paper also introduces the "MPTrain" application that applies the principle of the PAPA framework to generate playlists for a user. The PAPA framework draws its inspiration from two initially implemented systems. These are the PATS system and the HPDJ system. The PATS system uses the concept of "context of use" to predict what song to play next. The context-of-use is defined in [4] as the real world environment in which the music is heard, be it a party, a romantic evening or

traveling in the car or train. The context of use is specified by the user using the system. The HPDJ system "uses sensors to determine physical and physiological responses of a crowd and gives real-time biofeedback to inform the song selection algorithm" [4].

The typical workings of the PAPA framework is as follows: A user listens to music from his personal music library using a portable device. The system monitors the user's physiological responses via sensors attached to the user. The sensors are external of the mobile device. The sensors send the information wirelessly to the mobile device. The system then selects the next optimal song to be played. This is assuming that the system has already been trained with the user's profile which contains personal information and historical data from the user's previous interactions with the system. Nuria and lucas present an example of a user whose goal is to relax. The goal will be mapped to the values in the user's physiology, e.g low heart-rate, galvanic skin response, respiration and rate of movement. The user then listens to music from the system. The system will monitor the values of the user's response to the music. The values are then used to select the next "optimal" song. [4].

Nuria and lucas present 2 mappings that are estimated for creating a user model: the mapping between the musical features and the user's physiological states and the mapping between the user's goal and the user's state. These two mappings along with the historical data make up the user's model [4]. MP Train is a mobile application that applies the principle of the PAPA framework. It is designed to enhance exercise performance of users using music. MPTrain's hardware includes a heart rate and acceleration monitor that are attached to the user and are wirelessly

connected to the mobile device. MPTrain works as follows: it allows a user to enter desired workout conditions such as the desired heart rate before the exercise session starts. The system then monitors the current heart rate and compares it to the desired result. The system will then select appropriate songs that have the features to guide the user to the desired goal. The MPTrain allows users to explicitly change the songs if the user feels it does not match what he wants. Although the learning algorithm learns to map the physiological responses to the music features, it however does not incorporate user feedback in its algorithm.

Lifetrak

Lewis proposes a music player called Lifetrak that stays in tune with a user's life by using a music engine that is sensitive to the context of the user to drive what music is played [7]. The system is influenced by (i) the location of the user, (ii) the time of use of the system, (iii) the current speed of the user, and (iv) urban environment information such as weather, traffic and sound modalities. The above-mentioned contextual information is obtained via the use of a GPS unit, RSS feeds and a microphone.

Lifetrak is built on a learning model that is heavily reliant on user feedback. This helps determine if a chosen song is appropriate enough for a particular context. Lifetrak is made up of four main components. The first is the user space, which simply represents the database of songs for the user and context tags for each song. Context tags are tags that simply represent the context in which a particular song should be played. The second component is the context engine which gets the contextual information from different sensors and categorizes them in specific tags. The third component is the rating generator, which combines the context of the

individual and the song database which contains individual context tags for each song, and generates a ranked playlist. The final component of Lifetrak is the music player, which simply provides an interface for the end user. One of the key features of this system is that it is highly user-driven. Inasmuch as user feedback information is key to personalizing any playlist, the Lifetrak system's approach of relying on users to provide context tags for each song in the database is too cumbersome. This in effect defeats the intended purpose of the system, which is to reduce user interaction.

Personalized affective music player

The Affective Music player (AMP) [2] is built on the foundations of content validity, construct validity and ecological validity. Content validity simply means the system should show strong relations between the physiological measures and the affective states of the music player. Construct validity in this case is shown by establishing a good theoretical framework for the relation between physiological changes and the affective states. The ecological validity in this context simply means that the system should work properly in real life situations as opposed to only in controlled laboratory settings. The player presented in this thesis, CAAPG, also builds on these three validity foundations. These foundations led the designers of the AMP to formulate five design considerations.

The first is the fact that music taste is highly personal and as such, personally selected music is a much stronger "affect inducer" than "experimenter-selected music"; therefore the systems must be personalized. The second design consideration was the distinction between mood and emotions. According to this paper, "mood lasted longer and changed gradually, moods were not related to any

particular object and were often experienced without concurrent awareness of their origin" [2]. The AMP focused on mood since its goal was to direct affective state for a longer period. The third design consideration is the "Affective loop", which takes into account three steps that need to be repeated in the system; "(1)infer user's current affective state from physiology, (2) set an affective goal state and select music based on the goal and current state, and (3) measure the affective physiological changes". These three steps would need to be looped frequently. The fourth consideration was the external contamination of other physiological effects to the ones being measured (these effects are called noise). Probabilities are used to model physiological effects, which helps deal with noisy data. The fifth design consideration is that "physiological activities tend to move to a stable neutral state", therefore the principle of initial values is used in the system, as initial values will determine the affect goal.

Mood classification

Owen The paper presents a music classification system that helps users browse their music database by mood [4]. The mood classification system presented in this thesis borrows heavily from research done by Owen [4]. CAAPG goals are focused on (i) the generation of context-aware playlists, (ii) finding lost music and (iii) the experience of the user while using the system.

The hypothesis that combining the elements of "emotional modeling, audio feature extraction and lyrical analysis, will result in a novel and intuitive tool that users can use for their daily activity of generating playlists" [4], is one that this thesis builds on. Owen looks at two major approaches of emotional modeling, which are the

categorical and the dimensional approaches. "The categorical approach is one that consists of several distinct classes that form the basis for all possible emotional variations" [4].The categorical approach is often used for goal-oriented situations or in this case, purpose-driven situations. The dimensional approach however, "classifies emotions along several axes such as valence (pleasure), arousal (activity) and potency (dominance)" [4]. This paper used five audio features for its music classification. The five audio features presented were mode, harmony, tempo, rhythm and loudness. The paper used these features because they were part of the musical parameters used in Hevner's original mapping of musical features [4]. Kate Hevner's circular model of emotion is one of the known and accepted models in the field of psychology and music psychology. .

Hevner's model

Most of the literature that has to do with music classification point to Kate Hevner's circular model of emotion. Hevner groups emotions into eight categories, each having distinct musical characteristics; that is, low level audio features [4]. These categories are: dignified, sad, dreamy, serene, graceful, happy, excited and vigorous. The audio features studied by Hevner include mode, tempo, pitch, rhythm, harmony, and melody [4]. Music is classified into one of the emotional categories based on a weighted some of the above feature. Figure 1 below shows the different categories and the adjectives. There have been several revisions to the Hevner model over the years. This thesis will, however, use the original Hevner model in collaboration with Russell's model, as in used in Owen's paper, to map musical features to an emotional space.

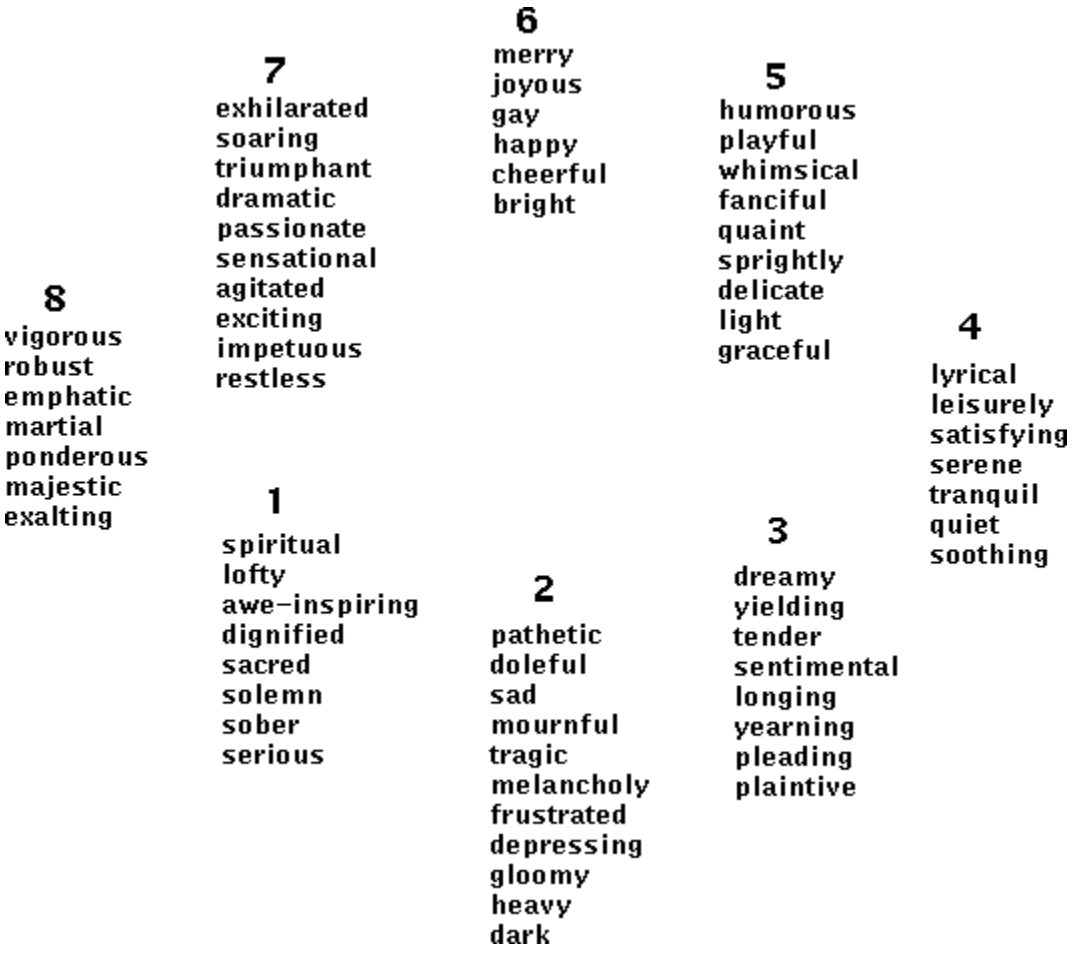

Figure 1: Hevner's adjective circle [7] p. 26

James Russell presents a dimensional approach to music classification. "A dimensional approach is one which classifies several emotions along several axes. These include valence, arousal, and potency [4]. James Russell proposed a circumplex structure (see figure 2). "This structure is made up of two main dimensions underlying the structure of emotion: valence (distinguishing emotions such as happy from sad) and arousal (distinguishing emotions such as excited from

sleepy)" [4]. This model can easily be represented on a two Cartesian plane with valence on the 'X' axis and arousal on the 'Y' axis.

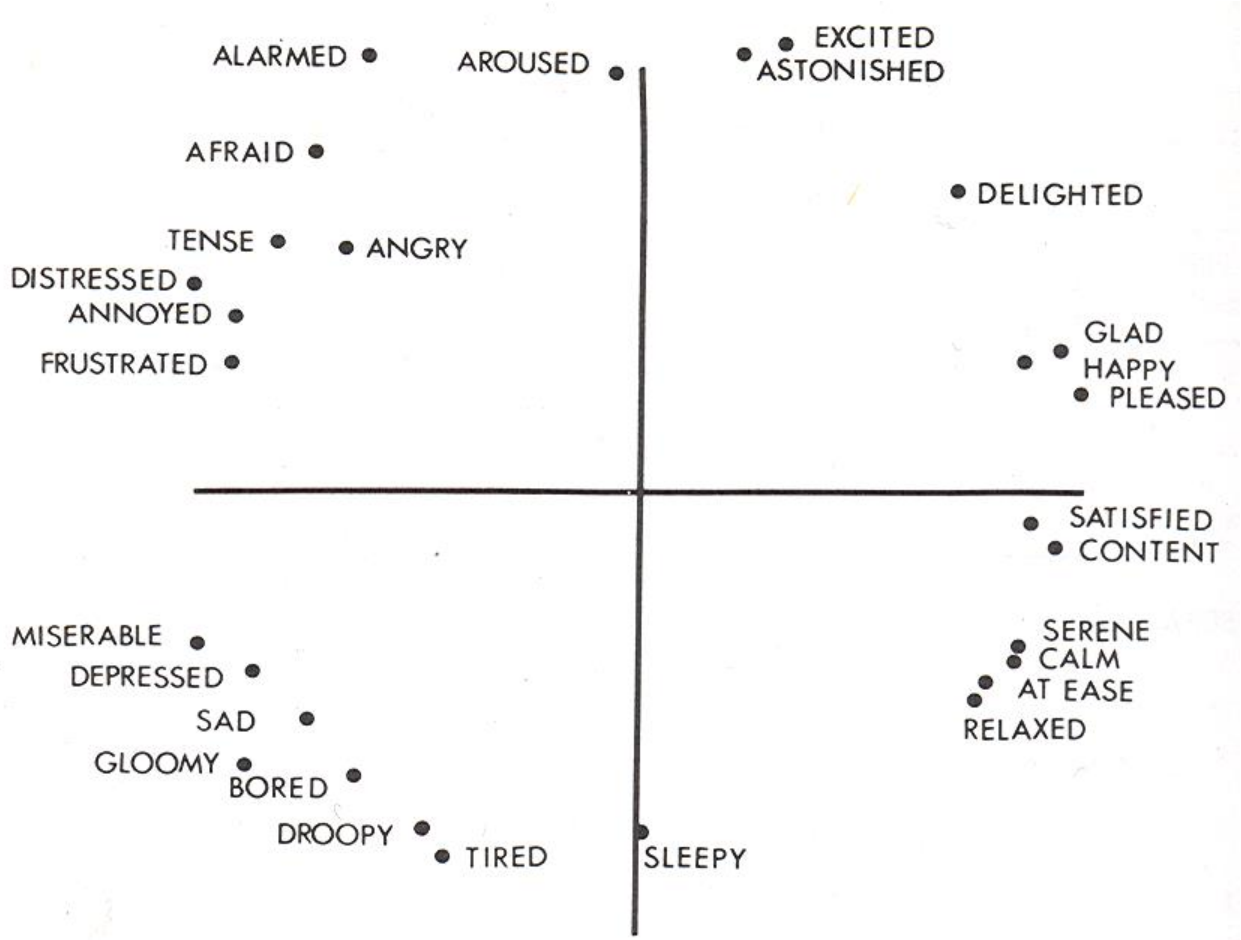

Figure 2: Multidimensional scaling of Russelled circumplex model of emotion [4], p. 25

Four out of the six features Hevner described are used for emotional modeling purposes in this research. These features are mode, harmony, tempo and rhythm.

Borrowing from Owen's research methodology in [4], melody and pitch will be ignored because the same extraction tool will be used hence it is expected that the same constraints that restricted his research will apply: "Melody apparently is a very difficult audio signal to extract and the extraction tool used here distorts the pitch of the audio file" [4]. Loudness will instead be used as the fifth feature because it results in high arousal and excitement. Table 1 shows the weighting of each feature and emotion. "Positive values translate to major mode, simple harmony, fast tempo, regular rhythm, and high loudness, while negative values translate to minor mode, complex harmony, slow tempo, irregular rhythm, and low loudness" [4].

Table 1: Mapping of musical features to Russell's circumplex model of emotion [4]**p.51**

Mood	**Mode**	**Harmony**	**Tempo**	**Rhythm**	**Loudness**
Pleasure	12	11	-7	-5	0
Excitement	24	16	20	-10	10
Arousal	0	-14	21	2	20
Distress	0	-14	21	2	10
Displeasure	-16	-2	-14	-3	0
Depression	-8	-4	-7	10	-10
Sleepiness	-12	4	-16	-9	-20
Relaxation	3	10	-20	-2	-10

The literature reviewed thus far indicates that there has been a lot of discussion around playlist generation and the use of environment features. However, little or no implementation has been done using contextual features simultaneously. This research seeks to implement a system that uses contextual features and combines them with established emotional models, such as the Hevner model, in implementing a playlist tool. A step by step description of how this is done is examined in the next chapter.

Reinforcement learning

Reinforcement learning is a sub-area of machine learning. As explained by chi chung-yi is how an agent ought to take actions in an environment so as to maximize the long–term reward of taking that action [1]. The learner in reinforcement learning can be defined as an agent that makes decisions about what actions to take in a particular environment. The learner receives rewards or penalties for the action it takes. After running a number of trials the learner should learn the best policy.. Reinforcement learning basically attempts to map states in a particular environment to a particular action.

Reinforcement learning is also sometimes called online learning because the system learns based on the feedback it gets from the environment when it takes a particular action. This means that reinforcement learning, unlike other forms of machine learning, is well suited for problems where we have no previous data. Other forms of machine learning, such as supervised and unsupervised learning

require preliminary data from which to learn, draw patterns and make conclusions. This method of learning is well-suited for determining a trade-off between long-term and short-term reward [1]. Reinforcement learning has been used to solve various problems such as elevator scheduling, robot control and games such as backgammon [1].

Reinforcement learning has been used in several recommender systems. In chi chung-yi's paper a system called WebWacther is described as a web tour guide that uses QLearning to give user's their desired pages. In this system pages represent states and hyperlinks represent actions. The rewards are computed based on the content of the page and its similarity to the user's profile key words. WebWacther is used for online information filtering which maintains a profile for each user containing keywords of interests and updates each word's weight according to the implicit and explicit feedbacks received from the user. The proposed learning method showed superior performance in information quality. It also shows good adaptation speed to user preferences [1].

In Chi Chung-Yi's thesis, he argues that the "Automatic playlist generating (APG) problem is better modeled as a continuous optimization problem". He proposes a model, where a user's behavior while playing music is collected as immediate feedback in learning a user's preference for a music emotion within a playlist [1].

Table 2: The summary of Chung-yi chi's notations [1] **p 18**

Notation	**Definition**	**Description**
S		The system
U		The user
M	M={m1,m2,…,mn}	The music collection
m	m ∈M	A song in the music collection M
O	O={replay,skip,rate}	The user operation set
o	O∈o	A user operation in the user operation set O
e	e=(o,t)	An operation entry
l	l=(m,t,t,Q)	A listening log
Q	Q={e1,e2,…,en}	A finite set of operation logs
H	Ht=(l1,l2,…,lk)	The listening history at time t
util(mt)		The utility function

Chi models the APG problem as a reinforcement learning problem, However chi does not take into consideration the context of the user. Unlike chi's work context-awareness is central to solving the APG problem using reinforcement learning. This thesis borrows a lot from the terminology used by chi in his paper [1], to define the APG problem. The summary of the notations are in Table 2 above.

In Chi's work, states are represented by a log of historical listening sequence of the user. This meant that there could be an infinite number of states which make the learning process difficult to converge. The historical states are reduced to a more manageable number *m*. this means every state has *m* song emotion classes. Actions in this work are defined as choosing a particular emotional class. The emotional class, just as in my work, is based on James Russell's music emotion classification. A simple system called MEonPlay is introduced in chi's work. MEonPlay works as follows: an episode will start with an initial song, and this song is either user specified or picked randomly by the system. An episode means a time frame in which the system is started by playing a song and ended by closing the music player [1]. The system then dynamically plays the next song. The user then gives feedback as any one of the three operations, to indicate whether or not they like the song[1]. This feedback is then used to recommend future songs based on the sequence of songs listened to in the past. Figure 5 shows an overview of the MEonPlay system.

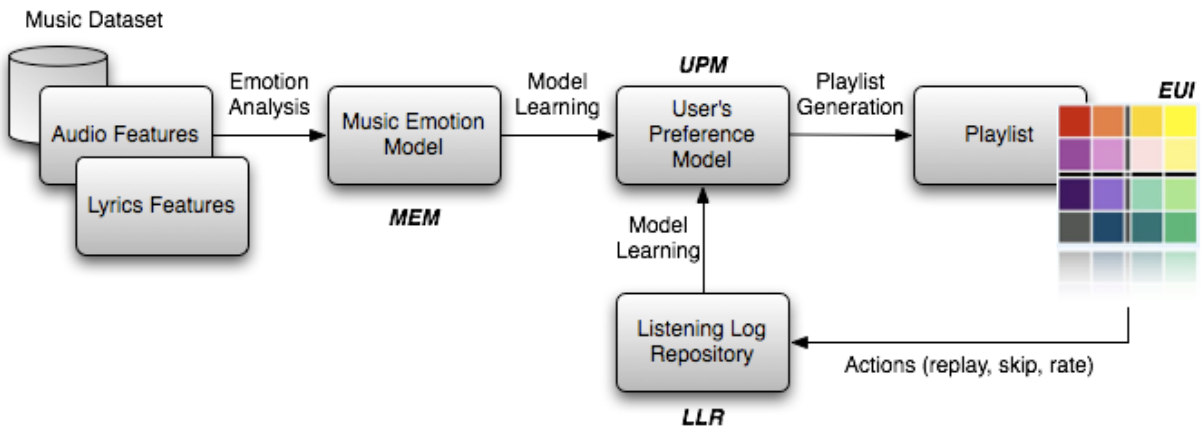

Figure 4: The System Architecture of MEonPlay Automatic playlist Recommender [1] P.20

Chapter 3:

Methodology

The goal of this thesis, as mentioned earlier, is to provide a context-aware personalized automatic playlist generation system, CAAPG. CAAPG will be built to learn a user's emotional preference based on his/her listening behavior given the user's listening context. The system will use its knowledge about the user to predict what song the user wishes to listen to next. Learning in this paper will be achieved using a reinforcement learning framework. This thesis will use a simulation method to evaluate the CAAPG system. This chapter consists an in-depth explanation of how the proposed system was implemented, as well as how the system was tested. The chapter will start with a presentation and description of the problem we are trying to solve. Next, we will explain key terminologies that will be used throughout the rest of the paper. Thirdly we will describe how we intend to solve the problem. Finally we describe what experiments we will do to evaluate the system.

Problem Definition

The APG problem as related to my system can be defined simply as the problem of choosing the right song with a particular emotion for a user in a particular context. Machine learning is known as the major tool in computer science for solving prediction problems such as the APG problem. Machine learning simply means learning from the past to predict the future. In this paper we have decided to solve the problem using reinforcement learning (RL). How and why RL is used in this system will be further explained later on in the paper.

In describing the problem, new terminology is used. For the purposes of this research I will borrow from the terminology used in [1]. Table 2 in the previous chapter, presents these notations and their description. My work however does redefine some of the notations to suit my purposes. The summary of the notation used for this thesis is presented in the table below. **S** denotes the system that will be solving the APG problem. **U** denotes the hypothetical user creates to test CAAPG. Music database M represents a finite set of songs that a user possesses. Find other key definition below.

Table 3: The summary of the notations

Notation	Definition	Description
S		The system
U		The user
M	M={m1,m2,…,mn}	The music collection
m	m Є M	A song in the music collection M
O	O={replay, skip, do nothing}	The user operation set
o	O Є O	A user operation in the user operation set O
e	e=(o,c)	An operation entry in a context c

Definition 1: Episode

An episode as defined by chi in his paper is a "one-time" listening period which starts when a user opens the music player and plays the first song. The period ends when the user closes or end the music player [1]. User operation set O= {replay, skip} represents a finite set of operations a user can perform during an episode [1].

Definition 2: Emotional Model

The mood classification portion of my methodology combines Hevner's circular model of emotion and Russell's two-dimensional model to map musical features to an emotional space. The emotional categories, used by CAAPG, are: Pleasure, Arousal, Distress, Displeasure, Relaxation, Sleepiness, Excitement and Depression. Every song that will be used in creating the CAAPG system will fall under one of the above categories.

Definition 3 Context/Contextual information

This APG system relies on the use of information about the environment of the user. In choosing appropriate contextual features, my thesis picks 3 key environmental features. These features are: the time of day, which could be morning, afternoon, or evening. The second feature is the day of the week, which is either weekday or weekend. The third environmental feature is the user's movement which is either moving or not moving. There are 12 possible contexts when we combine the 3 features mentioned above.

Definition 4 User Entry/Feedback

A user feedback is simply the responses of the user to actions taken by the system. There are only 2 possible types of user feedback allowed in this system; they are replay and skip. This makes up the user's operation set as indicated above. The interaction between the hypothetical user and CAAPG follows the below scenario:

1. An episode will start with either a user-selected song or a system selected song. The song selected by the system will be done arbitrarily.
2. Afterwards, CAAPG will continuously check what context the user currently finds himself. Then the next song is picked dynamically based on the already learned user preference model (UPM).
3. During an episode a user can perform any of the 2 user entries, to show the like or the dislike for a particular song's emotion in a particular context.
4. The system then updates the UPM based on the response from the user.

Proposed Solution

CAAPG

In this thesis, we propose a system named CAAPG. This system will help create personalized automatic playlists for users, by learning user habits when in a particular context. CAAPG's working process is as follows: Given a particular user's music database M and a given context, the system has to dynamically pick the next song emotion and then randomly pick a song from that emotional category, such that the reward of the action is maximized. The concept of reward will be explained in detail later on in this paper.

The CAAPG system consists of 2 major components, which are the mapping of the songs to the emotional categories (Emotional Model), and the mapping of a user's contexts to a particular emotional category (User Preference Model). As already stated above, the emotional model adopted for the CAAPG is Russell's two-dimensional (valence-arousal) model of emotion. The UPM in CAAPG aims to estimate a user's emotional preference in a particular context.

Using Reinforcement Learning

The goal of this thesis, as mentioned earlier, is to provide a context-aware personalized automatic playlist generation system.

The fact that CAAPG is personalized to individual users makes it possible to represent it as a reinforcement learning problem. Using any of the typical machine learning methods such as supervised and unsupervised learning would not work in solving this problem. For supervised learning method one needs a problem in which we the system has been trained before with previous examples. The PAPG problem is one in which we do not know the answer beforehand, and we have to keep optimizing our answers to the user's preference. The CAAPG system has to be able to predict what a user wants to listen to when given a particular context. The user feedback (skip and replay) will serve as reward systems that will help teach the system a user's preference.

Test and data

Once the implementation of this system is complete, I will test the system using hypothetical or simulated users. The characteristic of these users will be explained in the next chapter. The hypothetical users will provide the data that will be analyzed in this paper. The user's responses – skip or replay will serve as our major source of data. I will measure variables such as the skip rate and replay rate. I will also measure how long it takes for the learning algorithm to converge. In my test I will also determine what values the learning parameters should take to help my system learn fastest.

Chapter 4: Design and Implementation

This chapter gives a detailed explanation of how the CAAPG system is implemented. Here we discuss the music database used, the learning algorithms and the different tools used in creating the CAAPG system.

Database and Music dataset

Database Organization

CAAPG has one major database called “Music classification”. This database holds a song table. The song table contains three columns for names of songs, the emotional category of each song and the ID of the song.

Music dataset

CAAPG uses dummy data generated from generatedata.com. This data helped represent my music dataset in the CAAPG simulation. In my music database I have

a collection of 5000 dummy songs. All the songs are randomly categorized into the 8 different categories of James Russell's emotional model. I decided to use dummy data because of the time constraint and the general difficulty in using digital signal processing for classification.

Implementing Reinforcement learning Algorithm

Model

The problem of Automatic playlist generation is one we should consider as a continuous optimization problem. In this section, I will describe in detail the algorithm to predict a user's preferred song, and how I implement the personalized playlist generation as a reinforcement learning problem.

Music Emotion Model

My thesis uses the eight adjectives of Russell's Valence-Arousal plane model to categorize the music dataset. These eight adjectives are: pleasure, excitement, arousal, distress, displeasure, depression, sleepiness and relaxation. In my database I represent these categories as integer numbers from 1 to 8.

Modeling states and Actions

To predict the next song of any particular episode, the states should represent contexts. There are 3 different variables that make up each context. The first one is the time of day, which can either be morning, evening or afternoon, the day of the week. The second variable is the day of the week; for simplicity and the reduction in the number of states, the day of the week is divided into two categories:

weekday and weekend. The third variable is whether or not the user is moving. This means that we have 12 possible contexts/states at any one point in time. The action is then defined as the different emotional categories.

Reward and Penalties

The CAAPG system gives a reward for every positive action a user takes and sets a penalty for every negative action a user takes. All feedback in this system are assumed to be explicit, however some operation weigh more in reward or penalty based on importance. The positive actions are replaying a song, and listening to a song until it is over. For replaying a song, the reward is set to be twice that of just listening to the song to the end. This is due to the assumption that a user wanting to listen to a particular song again in a particular context is a sign of greater user preference than just listening to the song once. The negative action in this situation will be skipping a particular song. Skipping a song will be seen as explicit feedback of the dislike of that particular emotion in that particular state.

Policy

A policy in reinforcement learning is simply a mapping between states and actions or. It is sometimes known as a universal plan in artificial intelligence. The policy basically determines how the user will react to stimulus or feedback from its environment. The main aim of a policy is to help balance the trade-off between exploitation and exploration in reinforcement learning. Exploitation simply means that a system takes advantage of what it has previously learnt to predict what action would give it the best reward. Exploration however means the system tries out another possible action apart from what has been learnt previously; this action

may produce better rewards, sometimes however, the action taken might be worse than what has been learnt.

A policy simply looks to balance when to exploit and when to explore. There are three common policies in RL. These are Є-greedy, Є-soft, and soft max.

Є-greedy – this policy will pick the action with the highest reward most of the time. This policy favors exploitation, however it will still explore a few times. When exploring, the actions are selected uniformly thereby giving every action a chance to be tried.

Є-soft – this is the opposite of Є-greedy. This policy favors exploration more than exploitation. in simple terms it is more likely to explore than exploit.

softmax – This policy ranks each of the actions based on their values in a particular state. An action is selected randomly with regards to the weight associated with each action. This means that the worst action is unlikely to be chosen.

In this system, I implement a policy that uses a combination of both Є-soft and Є-greedy. When thinking about an automatic playlist generating system, it is intuitive to think that the longer a system is used the more it will learn about a particular user's preference. In the implementation of this system I start with Є-soft policy and explore more; however as time goes on I will then use a much greedier algorithm and exploit more.

Solving the APG problem with temporal-difference learning

Before I explain what Temporal Difference (TD) learning is, it is important I understand what a value function (VF) does in reinforcement learning. VF is an estimation of how good a particular action is in a given state [8]. The following notation is used for value functions.

$V^{\pi}(s)$ – this represents the value of a state, *s,* under the policy, *π* .

$Q^{\pi}(s,a)$ – this represents the value of taking action, *a,* in state, *s,* under the policy, *π*.

Temporal Difference Learning is used to estimate the VF. This helps to deal with the problem of delayed reward. If I did not estimate the value functions, The "state-action value pair" will only be updated at the when I receive the final reward [8]. TD learning methods calculate the estimate of the final reward at each state and the value of the value function is updated for every run the learner makes. its ability to update value functions based on estimation has earned TD learning the nickname *bootstrapping* in the reinforcement learning field.

It is easy to see why TD learning is the method best suited for the CAAPG problem. Since I did not know the reward for performing an action in a particular state, I needed to estimate the value at each point in time. Algorithm 1 below shows a simple generic TD learning procedure. The algorithm is made up of a nested loop, I initialize the state in the outer loop, afterwards I perform 4 major functions in the

inner (1) given the policy and the state (context)choose an action(emotional category) (2) take the action (3) take note of the reward and the next state (4) update the action-value function (Q-value) [1].

Algorithm 1: Generic Temporal Difference Learning

Initialize Q(s,a) using random values

repeat {for each new episode}

Intialize s

repeat{for each step in the episode }

choose action a given by π(the chosen policy) for s

take action a

observe reward r and the next state s^I

update Q(s,a)

$s \leftarrow s^I$

until s is terminal

until the end of learning

In this approach, two TD learning methods are considered: Q-learning and SARSA. The two methods both estimate value functions, however the difference between both methods is that Q-learning is an off-policy control method and SARSA is an on-policy control method. Off-policy TD method is one that can update estimated value functions based on hypothetical actions. On-policy algorithms however, update strictly on actions that have been taken rather than hypothetical actions. I use the following update rule:

$QLearning: \ Q(s,a) \leftarrow Q(s,a) + \alpha\,[r + \Upsilon \max_{\alpha}, Q(s^I, a^I) - Q(s,a)]$

$SARSA = Q(s,a) \leftarrow Q(s,a) + \alpha\,[r + \Upsilon Q(s^I, a^I) - Q(s,a)]$

The notations used in update functions are:

α – this is the learning rate, it is set between 0 and 1. 0 means that the system is not learning at all, meanwhile a closer number to 1 means that learning occurs faster.

Υ- Discount factor, it is also set to between 0 and 1. This model's the fact immediate rewards are more important than future rewards.

$\max_{\alpha}$ – the maximum reward that is attainable in the state following the current one, i.e. the reward for taking the optimal action thereafter.

Algorithm 2 below provides the pseudo code of the CAAPG solution using Q-learning. The notation for this algorithm is as follows: *s* and s^I represent the current state and next state respectively; *a* means the action to be taken, which also means the emotional category to be chosen; r simply means the reward received for the action taken. In simpler terms, the algorithm above simply starts by giving all initial Q-values (state action pairs) arbitrary initial values. Afterwards, it starts a loop that checks which state I are currently in and initializes the system with the state it has been given. Then an inner loop is started, in which I choose an action based on the policy I are currently using. In this system, taking an action means choosing an emotional category; once this is chosen, I randomly pick a song from the chosen category. After all this, I finally update the Qvalue using the

Qlearning function and update it to the next state. The inner loop is ended if it gets to a terminal state and the outer loop is terminated once I reach the number of episodes to be run.

Algorithm 2 CAAPG with Qlearning method

Initialize Q(s,a) arbitrarily

Repeat (for each episode):

Check if S has changed

Initialize S

Repeat (for each step of episode):

Choose a from s using policy derived from Q

(e.g, E-greedy)

Take action a, (Pick an emotion from the category a),

Pick random song from category a and play

Check if user skips, check if user replays, observe r and s^I

Update Qvalues

$s \leftarrow s^I$

Until s is terminal

Until training is stopped

Algorithm 3 CAAPG with SARSA method

Initialize Q(s,a) arbitrarily

Repeat (for each episode):

Check if S has changed

Initialize S

Choose a from s using policy derived from Q

(e.g, E-greedy)

Repeat (for each step of episode):

Take action a (Pick an emotion from the category a),

Pick random song from category a and play

Check if user skips, check if user replays observe r, s^I

Update Qvalues

$s \leftarrow s^I$

Until s is terminal

Until training is training is stopped

Algorithm 3 uses similar notation as those in Algorithm 2; the major difference between them is the update function they use. The policy used in this thesis is one that starts out as very exploratory and as the number of episodes increase, the policy become more exploitative. This gradual change is achieved using a decay constant (epsilon). The pseudo code for choosing whether to explore or exploit is as follow:

Epsilon is initialized as 0.2

Delta is initialized as 0

Pick a random number between 1 and 0

If the random number is greater than delta

Explore

Else

Exploit

Increase delta by epsilon after every episode

Chapter 5: Experimental Evaluation

Due to time constraints, and the difficulty of digital signal processing, I could not carry out real user evaluations. I instead used experiments with hypothetical users. In my experiment, I compared the two main temporal difference algorithms with various evaluation metrics. I also picked accurate parameters using the simulation. The details and results are in the following sections.

The Hypothetical Participants/Users

I used 2 hypothetical users for my experiment. The first type was a user who had a fixed set of characteristics, throughout the simulation period. This is mimicking a user who hardly ever changes his/her listening habits. The second type of user is one that is constantly changing his/her listening habits based on a certain pattern. The second user changes its listening habit every 20 episodes. The notation HU1 represents the first type of user, while the notation HU2 represents the second type of user.

Simulation Design

The aim of the simulation was to measure whether or not CAAPG would be able to learn the hypothetical users' habits and maximize reward over time. During the simulation, each participant ran for 200 episodes. In each episode, a total of 120 songs were played. I chose 120 songs for each episode this was to ensure that all 12 states could be represented. In my simulation, I only move to the next state after 10 songs have been played. Recall that states in this thesis are the same as context, and that there are 12 possible contexts. For analytical purposes, I simply printed out the values of each user's average reward, average number of skips, and average number of replays per episode. In the simulation I preset the preferences of each user.

Comparing Performance of SARSA with Qlearning

Comparing reward rate

In figure 5 and figure 6, we clearly see that both SARSA and QLearning try to always maximize reward for both users. For user 1 we see that SARSA eventually flattens out and only gives a constant average reward. However the average reward for Qlearning fluctuates meaning that Qlearning does not do well with convergence. For user 2 SARSA's average reward drops gradually over 200 episodes, while Q learning maximizes its average reward and it increases over time. This shows that for user Qlearning might be too volatile, however for user 2 Qlearning learn the user's changing patterns. For a real world experiment Q learning would be a better

option than SARSA. This same phenomenon is also noticed in the other graphs below.

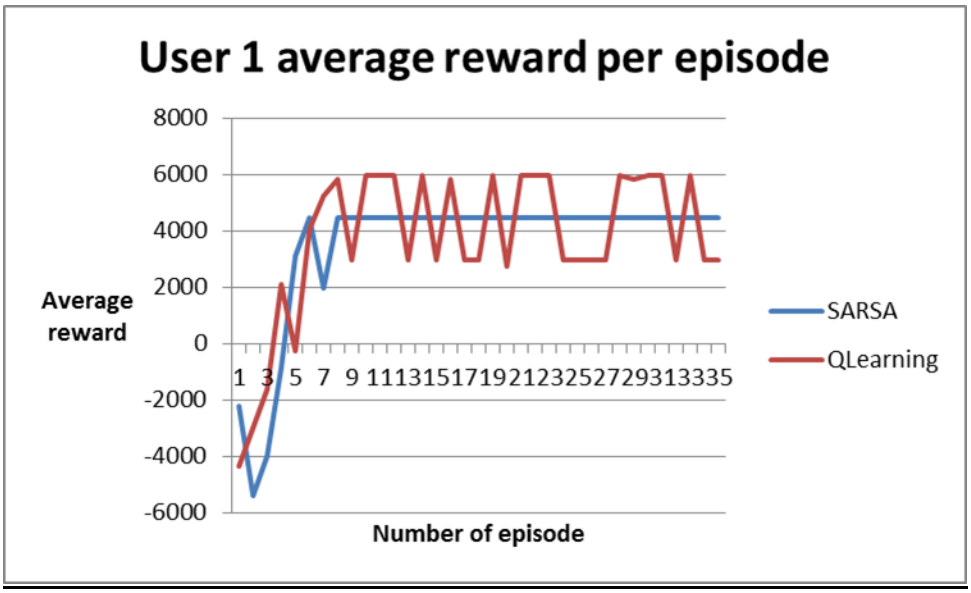

Figure 5

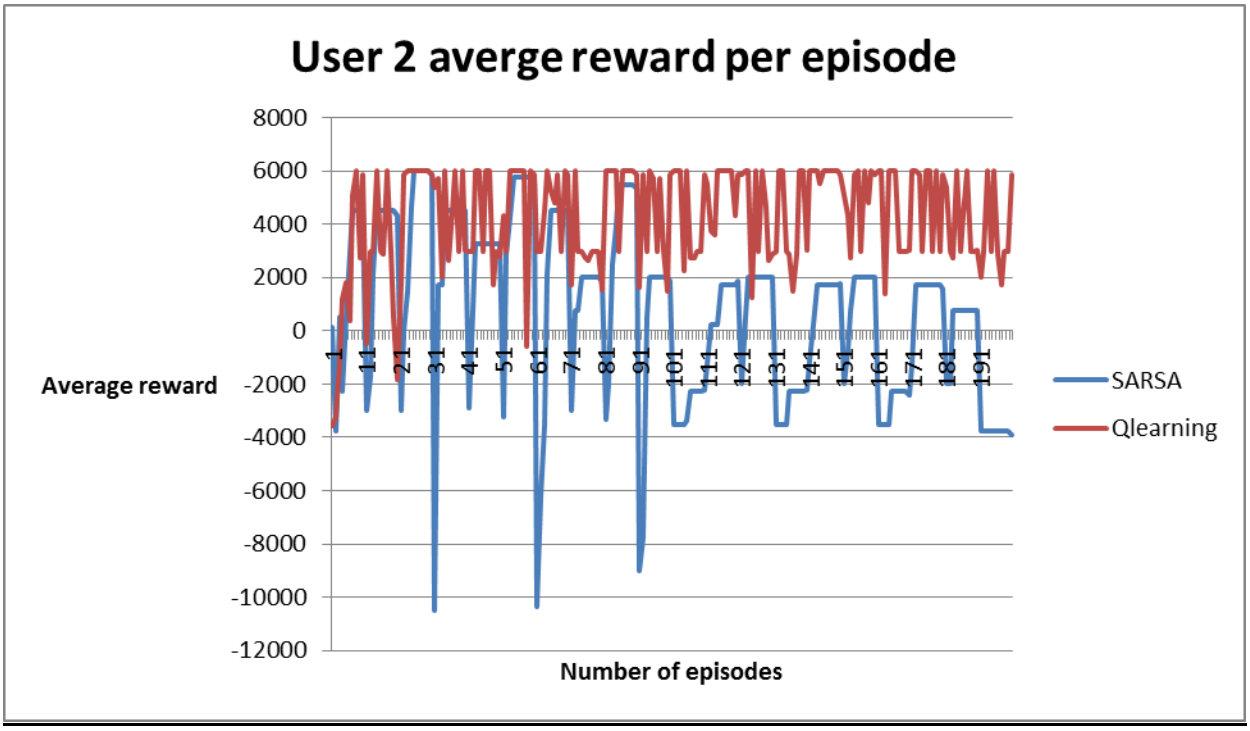

Figure 6

Comparing replay rate

In graph 7 and 8 below, we compare both method's replay per episode for user 1 and 2. We can see a similar pattern to that when we compared the average reward. SARSA flattens out beyond a certain point for user 1, however for user 2 it gradually drops over time.

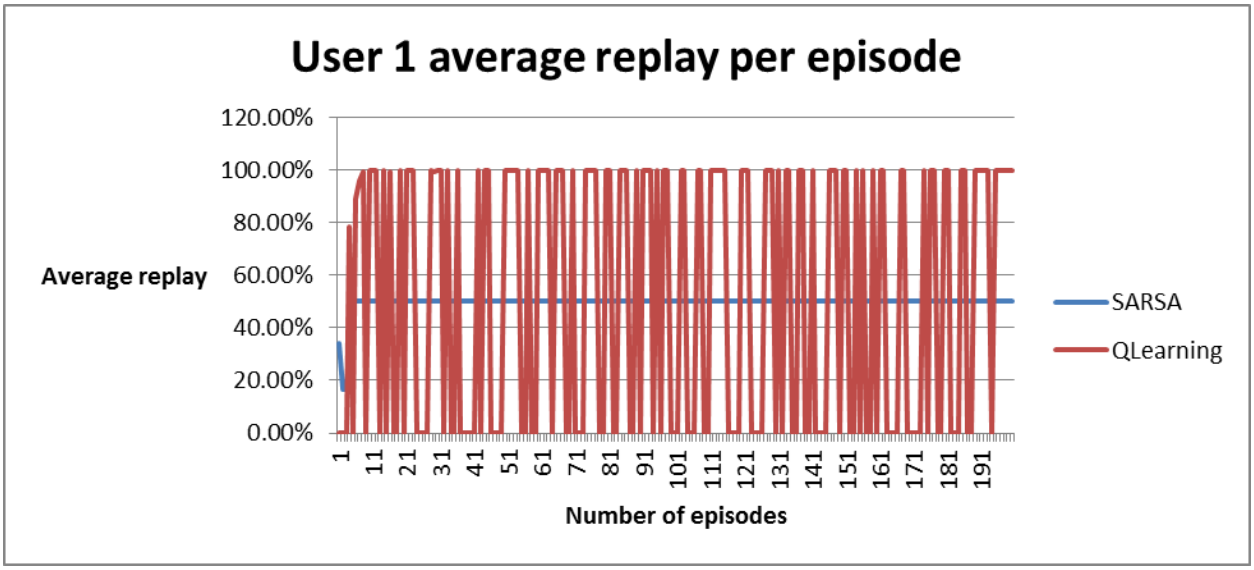

Figure 7

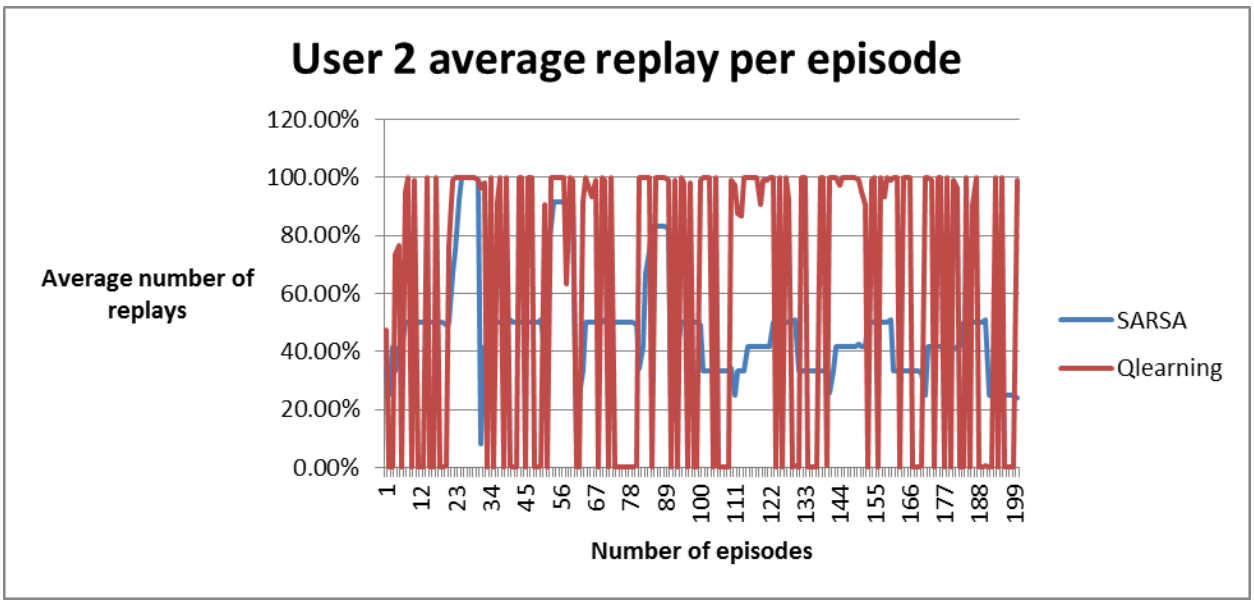

Figure 8

In figures 9 and 10 below, we see that both methods keep decreasing the number of skips. We can clearly see that SARSA does better for user 1, however the difference is quite negligible. For user 2, we clearly see that Q learning does much better than SARSA. As it quickly minimizes the average number of skips.

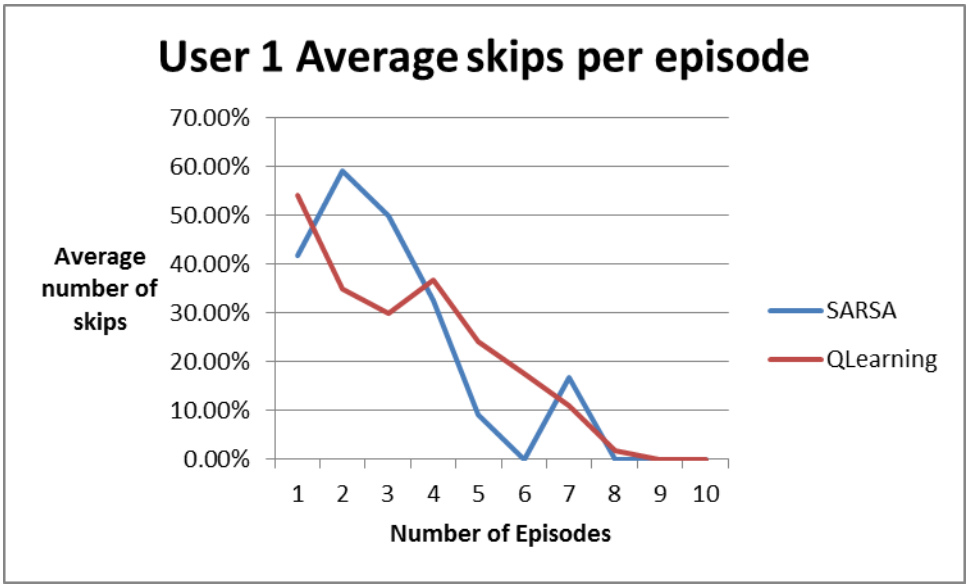

Figure 9

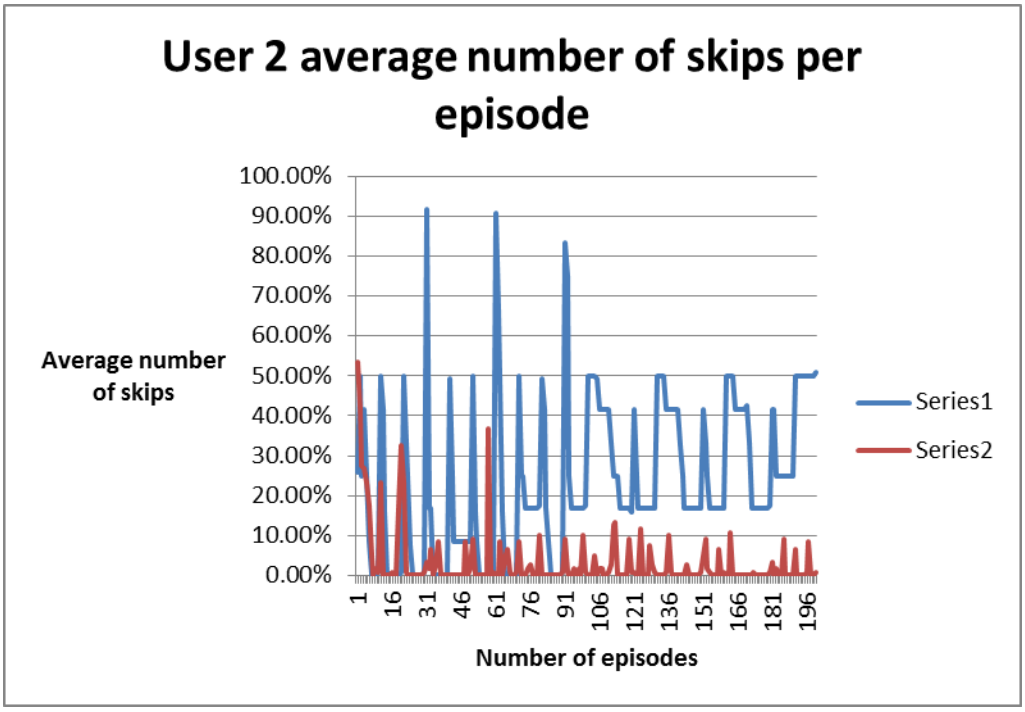

Figure 10

Parameter Selection with Simulation

To recap my experiment, I used hypothetical users to evaluate the approach in this paper. I use 2 types of hypothetical users. The first type of user is one that has a set habit; he never deviates from the original pattern. The other type of user is one who changes his listening pattern. In this hypothetical case, every user has two possible feedbacks: either replay or skip the song.

During the testing, I allowed each user to run the system for 200 episodes. Each state is allowed to run 10 sequences, before it moves on to another state. An episode only ends after 120 episodes. This guarantees that all 12 states will be used in one episode, each running 10 times.

Different decay constants

The decay constant as explained above controls how I exploit or explore. If I set the decay value too high, it will exploit too fast rather than explore. However, setting epsilon to a low value will allow more exploration but will not follow the optimum value quickly enough. To determine the appropriate value for epsilon, I gradually increase the value and observe the average reward, the average number of skips, and the average reward per episode.

			epsilon				
			0.01	0.05	0.1	0.15	0.2
Hu1	SARSA	Average Reward	4800	5728.625	5622	**5887.75**	5439.75
		Average Skip	6.805	1.435	0.745	**0.65**	0.345
		Average replay	106.02	116.315	108.6	**118.755**	99.31
	Q-Learning	Average Reward	4680.625	5712.625	5865.5	5898	**5918.94**
		Average Skip	7.535	1.645	0.745	0.605	**0.47**
		Average replay	104.895	116.725	118.34	118.94	**119.055**
Hu2	SARSA	Average Reward	3306.125	4312.5	4091.875	4415.125	**4441.25**
		Average Skip	8.075	1.15	1.055	0.55	**0.4**
		Average replay	52.615	58.245	48.945	59.35	**59.645**
	Q-Learning	Average Reward	3326.125	4252.625	4387.375	4412.875	**4428.875**
		Average Skip	7.855	1.665	0.745	0.58	**0.495**
		Average replay	52.315	58.425	59.215	59.41	**59.625**

Table 4: The average reward, number of skips, and replay per episode.

The table above shows two hypothetical users using the system. I can see that as epsilon increases, the average reward increases, the average skip reduces, and the average number of replays increase. For Hu1, in SARSA however, epsilon reaches an upper bound of 0.15 before the average reward and replay start to drop. For the purposes of this thesis, I use an epsilon value of 0.2

Chapter 6: **Conclusion**

The goal of this thesis was to put forward the hypothesis that an emotional-based and context aware system is a valuable way to solve the APG problem. First, I described the APG problem and its notation. CAAPG, an emotional-based and context-aware system, is proposed as a solution to the problem. I decided to model the problem as a reinforcement learning problem, because of the continuous optimization characteristic of the problem. I introduced the concept of reinforcement learning and key terms. I then described, in detail, how I formulate the problem as an RL problem. I explained how a user's context related to states and how the actions related to the emotional categories. Temporal Difference learning, which uses optimization algorithms such as SARSA and Q-Learning, is applied to CAAPG. I then described how I went about balancing exploration and exploitation using a decay constant I got after experimenting. The experiment was done with 2 hypothetical users who had different habits. I use metrics such as the average number of skips per episode, and the average number of replays per episode to measure the performance of the system. Both Qlearning and SARSA learn the user's preference model. However, Qlearning outperforms SARSA for this particular problem.

Limitations

In developing the CAAPG system, I faced major challenges. The previous plan of this project was to build an android application, however due to the difficulty of digital signal processing, I decided to simulate my system and show that it can learn from a hypothetical user. Lack of test in a real world environment is the second major limitation. This comes as a consequence of the first limitation. The final limitation I faced was that little or meager amount of literature was available on the use of reinforcement learning on playlist generation system.

Summary of contributions

Contrasting my work with what already exists in the field of Automatic playlist generation, my contributions are as follows: Firstly, I use an approach of playlist generation based on song emotions rather than the more traditional metadata and audio similarity methods. The use of emotional models is not novel however, but it is an emerging trend. Secondly in my work, I do not consider the APG problem as a static problem I rather see it as a continuous optimization problem. However, the major contribution my work makes in solving the APG problem is the use of creating a playlist based on emotion while taking the contextual information of the user into consideration.

Future work

This thesis gives good insight into the results of using emotion-based APG systems with contextual information. One of the major drawbacks of the thesis, was that it could not be tested on real users, doing a proper user evaluation should be further examined in the future. In this system, there are only three different features – time of day, day of week and movement – that make up what is called context of use. In the future, other key features in the users' environment that can influence users' listening habits should be examined. The speed of learning is essential for this system to work in a real world environment.

Bibliography

[1] C.-y. chi, Learning Emotion Transitions for a Personalized Playlist Recommender, TAIWAN: National Taiwan University Department of Computer Science & Information Engineering , 2009.

[2] J. .. Joris, v. d. b. L. Egon and W. H. Joyce, "Personalized affective music player," in *Affective computing and intelligent interaction and Workshops 3rd international conference*, Amsterdam, 2009.

[3] L. Sizer, "Towards A Computational Theory of Mood," *British Journal for the Philosophy of Mood,* vol. 51, no. 4, pp. 743-769, 2000.

[4] M. C. Owen, *A Mood-Based Music Classification and Exploration System,* Massachusetts: Massachesetts Institute Of Technology, 2004.

[5] R. Gordon, B. Dan, B. Ted and C. Eugene, "Towards a personal Automatic Music Playlist Generation Algorithm: The Need for Contextual Information," in *Audi Research Group*, Dublin, 2007.

[6] O. Nuria and K.-S. Lucas, "PAPA: Physiology and purpose-Aware Automatic Playlist Generation," in *Proceedings of 7th International Conference on Music Information retreival*, victoria, 2006.

[7] H.-J. M. J. B. F. L. Lewis Michael, A Handbook of Emotions, London: The Guilford Press, 2008.

[8] K. A. U. v. R. Eden Tim, "Reinforcement learning," CSE, [Online]. Available: http://www.cse.unsw.edu.au/~cs9417ml/RL1/index.html. [Accessed 12 04 2014].

[9] C. J. Sally and F. A. Bainbridge David, "More of an Art than a Science supporting the Creation of playlist mixes," in *proceedings of 7th International Conference on Music Information Retrieval*, victoria, 2006.

[10] V. F. van Gullic Rob, "Visual Playlist Generation on the Artist Map," in *Proceedings of 6th International Conference on Music Information Retrieval*, london , 2005.

[11] R. Sasank and M. Jeff, "Lifetrak: Music in Tune With Your Life," in *Proceedings of the ist ACM International workshop on Human Centered multimedia*, New York, 2006.

[12] W. Xinxi, R. David and W. Ye, "Context-Aware Mobile Music Recommendation for Daily Activities," in *Proceedings of the 20th ACM International conference on Multimedia*, New york, 2012.

[13] P. Key, *Research Design in Occupational Educational,* Oklahoma: Oklahoma state University, 1997.

[14] W. Trochim, "Research Methods," in *The Concise Knowledge Base*, Thomson Corporation, 2005.