User Based Hybrid Algorithms for Music Recommendation Systems

By

Murtadha Sami Luaibi Al-Maliki

The thesis is submitted in partial fulfilment of the requirements for the award of the degree of Doctor of Philosophy of the University of Portsmouth

September 2018

Declaration

Whilst registered as a candidate for the above degree, I have not been registered for any other research award. The results and conclusions embodied in this dissertation are the work of the named candidate and have not been submitted for any other academic award.

Murtadha Sami Luaibi Al-Maliki

Acknowledgment

I would like to express my gratitude to my country Iraq, represented by the Prime Minister Office – Higher Committee for Education Development (HCED) in Iraq for giving me this full scholarship to study PhD in the University of Portsmouth; This work would not have been possible without their support and encouragement.

Foremost, I would like to express my sincere gratitude to my supervisor Dr. Linda Yang for the continuous support of my PhD study and research, for her patience, motivation, enthusiasm, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better supervisor and mentor for my PhD study. Besides my supervisor, I would like to thank the University of Portsmouth, represented by the School of Energy and Electronic Engineering, the University library and the graduate school – represented by Dr. Heather MacKenzie for being very supportive by providing me an excellent environment and wise advice during my PhD study.

Last but not the least, I would like to show my deepest thankful and gratitude to my parents, my brothers, my sisters and my friends (Mahmoud Al-Faris, Mohammed Ibrahim, Mohammed Al-Wattar, Haider Murad, Ahmed Ahmed, Muntadher Fadhil, Mohanad Al-Behadili, Wassem and his wife Maha, Mr. & Mrs. Kirby, etc.) for their love, prayers, courage and moral support they gave me throughout the study.

Dedication

Every challenging work needs self-efforts as well as guidance of elders especially those who were very close to our heart.

My humble effort I dedicate to my sweet and loving

Father & Mother

Whose affection, love, encouragement and prays of day and night make me able to get such success and honour,

Along with all hard working and respected

Teachers

I will be forever grateful to whomever teaches me anything worth learning

Abstract

The amount of music available digitally is overwhelmingly increasing. The main purpose of music recommendation systems is to suggest quality relevant songs that fit with the user's preferences. Currently, most of the streaming music systems recommend songs based on Collaborative Filtering and Content-Based filtering techniques. However these systems fail in dealing with the Cold-Start problem. This thesis presents user-based hybrid algorithms for music recommendation systems to address the Cold-Start problem and to recommend music for both new and existing users based on their context by integrating the social information to provide context aware personalized music recommendation.

This thesis makes two major contributions: First, hybrid recommendation algorithms are developed using multi-strategy approach to give more accurate recommendations by combining collaborative filtering, content based and the user's context obtained from social network in order to provide both new and existing users with an easy way to discover new songs. In this way, the system is able to estimate what artist/song would match user preferences. Second, a generic Context-Aware Personalised Music (CAPM) framework is proposed for supporting the rapid development of context-aware music recommendation systems and for clarifying the whole process of recommendation. As there are myriad approaches of recommendation, there is a need for a generic framework not only to gather these approaches, but also to interpret them under the proposed framework. Recommendation algorithm types differ by the input structure. For example, social recommendation algorithm uses social information, collaborative filtering uses users rating data, whereas content based recommendation uses item's characteristics. This difference affects enormously the representation of data and consequently the process of recommendation. CAPM is able to present different input data and uniforms the recommendation process.

The proposed algorithms and the framework have been successfully evaluated via practical experiments by real users. The practical experiments are carried out by presenting a Context-Aware Personalised Music (CAPMusic) application in Google Play which helps users to discover new artists, albums or songs. Satisfactory results have been obtained which indicate that using the proposed hybrid recommendation algorithms leads to better results compared with using the pure content based and collaborative filtering techniques.

Table of Content

List of Figures

List of Tables

Glossary

MRSs………Music Recommender Systems

CARS………Context-Aware Recommendation System

CB..………...Content-based

CF…………..Collective Filtering

CAPM………Context-Aware Personalised Music Recommender Framework

MAE……..…Mean Absolute Error

KNN……..…K-Nearest Neighbour

LOD………..Linked Open Data

RDF………..Resource Description Framework

LDSD…….. Linked Data Semantic Distance

LDSDd…….Linked Data Semantic Distance Direct

LDSDid……Linked Data Semantic Distance Indirect

TF-IDF…….Term Frequency and Inverse Document Frequency

AP…………Average Precision

MRR……….Mean Reciprocal Rank

S(N)………..Self-information-based Novelty

R……………Recovery

Chapter 1

Introduction

Over the past ten years, people's consumption of music has altered dramatically. Music's digitisation has had a massive impact on the music industry. The evolution of the internet has also contributed to this transformation as it was originally both the source of digital music and its distribution channel. As a result, a huge amount of music became accessible. Narrowing the scope to music, we encounter this problem in the present world of digital music distribution. We are already faced with the new paradigm of music consumption: listeners now have instantaneous access to digital music collections of an unprecedented size. The majority of music recordings are available online, and the amount of digital music counts tens of millions of tracks and grows extensively. Major Internet stores such as the iTunes Store contain up to 28 million tracks adding thousands of new tracks every month. Such amount of music is not surprising, as music plays an important part in people's everyday life, and more and more people express and share their music creativity by owning modern technology. A recent study found that British adults, when randomly probed via their mobile phones (North, 2004), are interacting with music in 39% of cases. Listening to music has become the top leisure-time activity for most people (Rentfrow & Gosling, 2003, 2006). Without doubt, music enthusiasts now have wider access to music than ever. However, it remains difficult to discover new artists that one might like. Even though there are many music recommendation systems, they often get stuck in classifications of music that are too simplistic to yield anything interesting. Spotify, for example, has many playlists, but too often those playlists are based on simple genre tags, which results in a rather dull listening experience after a while.

The most used methods in MRSs are Collaborative Filtering (CF) and Content-Based (CB) filtering techniques. Collaborative Filtering is the process where the system analyses a user's preferences from his historical usage data. The system then recommends tracks based on other users that share the same music preferences with him. On the other hand, Content-Based technique uses a song´s content and metadata such as artists, album and genre to recommend music (Tkalcic & Chen, 2015). CF usually performs better recommendations than CB (Tkalcic & Chen, 2015). This is true only when we assume that there are already usage data such as previous tracks' ratings. Otherwise, CF cannot work accurately and as a result, it will suffer from the well-known Cold-Start problem which is divided into two categories: new items and new users. The first category refers to the issue caused by new items that are supposed to be recommended, but there is not enough information (such as ratings) associated with them. On the other hand, the new user problem happens when a new user joins a system, but little is known about him. As a result, the recommendation system cannot make personalised recommendations for him until he starts rating different items (Negre, 2015). On the other hand, CB is generally less sensitive to the Cold-Start problem, because it can still recommend items, even though it lacks ratings (Tkalcic & Chen, 2015). However, the good music recommender systems must be able to recommend relevant songs to the users that can deal with the Cold-Start problem and the systems must be able to provide the novelty in the recommended songs which means the ability of a recommender system to recommend new songs that the user did not know about before in different contexts.

1.1. Motivation

The MRS methods that are used most frequently are Collaborative Filtering (CF), which is the process where the system analyses a user's preferences from his or her historical usage data and Content-Based (CB) techniques which analyse the item descriptions to identify songs that are of particular interest to the user (Duan, 2015; Zhao et al., 2017). CF recommends songs to people that are based on other users who share the same musical preferences as them. In contrast, CB uses the metadata and content of a song, such as the album, the artist and the genre in order to recommend songs. It has been found that CF generally gives better recommendations than CB. However, this is only true if there is usage data available, such as the ratings given to previous tracks. If this is not the case, then it will not prove accurate results and, consequently, suffer from the Cold-Start problem, which includes two categories of problems – new items and new users. The first problem refers to the new items that are meant to be recommended, but the information that is associated with them, e.g. the ratings, is insufficient. The second occurs when a new user joins a system and not much is known about him or her (Crane, 2011). Consequently, the recommendation system is unable to give recommendations that have been personalised for users until they begin to rate different items. CB is not usually as sensitive to the Cold-Start problem, as it has the capacity to still recommend items, even if it does not have enough ratings. CB has its own issues, however, and so is not without problems (Tkalcic & Chen, 2015). The biggest of these is that it will recommend songs the user is familiar with already. The objective of this thesis is to put forward a solution to solve the Cold-Start problem in MRS and also offer users new music recommendations they have never heard before.

Therefore, the motivation for this thesis is to increase the performance of a music recommender system which includes the diversity and the novelty by reducing the effects of the Cold-Start problem through combining three recommendation techniques into hybrid approaches. Thus, people will be provided with a solution that will enable them to receive more accurate and better recommendations that are based on their own music preferences.

The thesis will focus on the Cold-Start problem (new users and new items) in addition to the existing users. Specifically, it will recommend both new artists and songs that the user will not have listened to before and it will recommend songs to the existing users who already have a listening history. These recommendations will be based on users' existing preferences using MRS, in addition to information from a number of other sources. This thesis will, therefore, answer the following research questions:

i. Which of the factors in MRS can be utilised to tackle the new category of the Cold-Start problem?
ii. How can these factors be utilised to improve the recommendations that are given when the Cold-Start new song problem occurs?
iii. How can the system recommend relevant songs to new users who have just registered?
iv. How can the system recommend new songs to existing users who have listening histories?

1.2. Aims and Objectives

The aim is to provide both new and existing users with an easy way to discover new songs and provide personalised recommendations that will successfully work out what the user's musical preferences are. By analysing how they interact with the system, it will be possible to work out which group or artist the user will want to hear at any given time. Users do not wish to listen to the same genres or artists all the time, as users need to be surprised sometimes and enjoy something new.

The objectives of this research are:

- Develop a generic framework that delivers "contextual recommendations" that are based on the combination of previously gathered user feedback data (i.e. ratings and listening history), context data for supporting the development of context-aware music recommendation systems.

- Support the recommendation systems' algorithms by using a multi-strategy approach that combines the results of different base recommenders and generic recommenders into a final recommendation.

1.3. Contributions

The following contributions are made by this thesis:

i. It develops user-based hybrid algorithms for music recommendation systems to provide both new and existing users with an easy way to discover new songs and provide personalised recommendations by using the multi-strategy approach by combining three recommendation techniques: Collaborative filtering, content based and context based in order to address the Cold-Start problem.

ii. It systematically develops Context-Aware Personalised Music (CAPM) Framework for supporting the development of context-aware music recommendation systems. CAPM supports the representation, indexing by retrieving only songs that are estimated as relevant for the users, based on the user's profile, i.e., a representation of the user's music preferences and user's context.

1.4. Thesis Outline

The thesis is composed of a total of six chapters. These have been structured in the following way:

- Chapter 2 introduces both the research background and the related work and theories about recommendation systems.
- Chapter 3 describes the proposed CAPM framework; communication between the different components are expounded; design and technological choices are justified; and the data domain is explained.
- Chapter 4 illustrates the multi-strategy approach which is the recommendation algorithms.
- Chapter 5 presents the results of the experiments and explains the selection of the metrics used for evaluation of the developed system.

- Chapter 6 forms the conclusion of the work described in this thesis, along with suggestions for possible future work.

Chapter 2

Background and Related Work

This chapter presents the background and the related work on the recommendation systems techniques and context-aware music recommender systems. It discusses how this research relates to and differs from previous research works by others. Section 2.1 presents the main RS approaches, as well as the main recommendation system techniques that been used to address the issue of information overloading and cold start problem. Section 2.2 discusses previous studies about the acquisition of user profiling representation and related concepts. Section 2.3 provides an overview of the music recommendation systems while section 2.4 illustrates the most important factors that affect music recommendation system. Section 2.5 presents the mood categories. Finally, section 2.6 introduces the linked open data (LOD) technique and shows how using this technique can enhance the recommendation accuracy.

2.1. Recommender systems: main approaches and challenges

Recommender systems (RS) have developed to be a vital field of research since the emergence of the first literature on collaborative filtering during the mid-1990s (Gunawardana & Shani, 2015; Guy, 2015; Masthoff, 2010). Generally, the definition of RS is described as the supporting systems which assist users to discover products, information, or services like books, music, movies, websites, digital products, and TV programs by gathering and analysing recommendations from other users, which translates to reviews from different consultants, and user characteristics (Celma, 2010; Negre, 2015).

The objective of creating RS is to moderate overload of information by recovering the most important information and services from a gigantic amount of data, consequently providing tailored services. The most vital element of a recommender system is its capability to guess the preferences and interests of the user by analyzation of their behaviour and/or that of other users to produce tailored recommendations (Tkalcic & Chen, 2015). Electronic service personalization methods are represented by RS and have gained much consideration in the past two decades (Celma, 2010).

With the growth of recommendation techniques and approaches, ever more RS (software) have been instigated and numerous real-life recommender system software have been created. In recent times it has been pointed out that application study is the key focus of research of contemporary recommender system research, particularly in the present-day age of big data (Negre, 2015). The uses of RS include endorsing music, movies, television programs, documents, books, websites, tourism scenic spots, conferences, and learning materials. Moreover, they encompass the areas of e-commerce, e-learning, e-library, e-business, and e-government services. Thus, this section focuses on the existing literature on RS, with a particular focus on Music Recommender Systems (MRS) (Hidasi et al., 2016).

Recent years have witnessed the boom of MRS, thanks to the arrival and realization of online streaming services, which today make all music almost available in the world at the fingertip of the user. Despite the fact today's MRS significantly assist users in finding interesting music in these large catalogues, MRS literature is still struggling with substantial challenges. Particularly, when it comes to create, incorporate, and assess recommendation approaches that incorporate information beyond standard user, item interfaces or content-based descriptors, but go deeper into the very core of listener preferences, needs, and ideas, MRS research develops to be a big endeavour and interrelated publications quite meager (Jannach and Adomavicius 2016).

2.1.1. Collaborative Filtering Models

Collaborative filtering is the terminology applied to methods that scrutinize the connections between users and items in a large set of data and make recommendations centred on existing links between nodes (Delannay & Verleysen, 2008; Chen et al., 2015). One of the common methods in collaborative filtering is the application of existing links to make decisions about similar users and products. The comparison is only determined through history, for instance, there can be a similarity between two users if they have bought many of similar products, so the rating of one user can be used to assume a rating for another.

Collaborative filtering has savoured a prolonged popularity in tasks of recommendations. It was originally used commercially in a method called Tapestry in 1992 to endorse newsgroup messages to readers (Duan, 2015). In this method, reaction and annotations from existing relationships of user document are employed to select documents that are exciting for other users. This technique first uses the term collaborative filtering to designate that people indirectly cooperate by recording their feedbacks to documents, empowering others to make judgments based on those feedbacks.

There are mainly two broad groups of collaborative filtering: memory-based and model-based methods. Memory-based techniques merely memorize the matrix rating and issue recommendations centred on the link between the quizzed user, item and the rest of the rating matrix. Model-based techniques fit a model parameterized to the given matrix rating and then provide recommendations centred on the model fitted (Liang, 2016). Furthermore, the most common memory-based CF systems are neighbourhood-based systems, which envisage ratings by making a reference to users whose ratings are alike to the user queried, or to items that are equivalent to the item queried. This is driven by the supposition that if two users have equivalent ratings on various items they will have equivalent ratings on the remainder of items. Moreover, if two items have equivalent ratings by a proportion of the users, then the two items must have equivalent ratings by the users remaining (Oramas et al. 2017).

Particularly, CF user-based techniques detect users that are comparable to the user queried and estimate the rating desired to be the ordinary ratings of these comparable users. Likewise, item-based CF detects items that are comparable to the item queried and approximate the rating desired to be the ordinary of the ratings of these comparable items (Zhao et al., 2017). For example, in a neighbourhood-based model of similarity, users are likened to each other to identify their nearest neighbours based on their past. At that juncture, to make a likelihood for user *u's* perception on product *p*, the technique looks at the perception of the neighbours of *u* concerning *p*. Furthermore, an additional similarity technique is the item-based model, which scrutinizes product similarity rather than user similarity. This method has been the core of recommendation engine used by online giant store Amazon (Zhao et al., 2017). Its benefit is that similarities of products can be calculated offline, and once a user requires a recommendation of a product, the system executes a fast lookup of similar items to ones in the history of the user. This speediness has been beneficial for accessibility in Amazon's huge purchase network (Vasile et al. 2016).

Collaborative Filtering systems have numerous pros, like the ability for taking an item quality or defect into consideration when suggesting items, especially in unequivocal customer ratings. For instance, a local music band may fall into an identical genre of music like a popular rock band would in all over the globe, but this does not promise that they may have the same level of quality. This reveals that items quality documentation is obviously an advantage for Collaborative Filtering. Moreover, Collaborative Filtering can prevent deficient recommendation and suggestions by taking the preference of clients which are true into an account. The second advantage in which the algorithms of Collaborative Filtering are predominantly useful and applicable in areas where the scrutiny of content is difficult or very expensive, like film and music suggestion, without challenging any field of knowledge (Vekariya & Kulkarni, 2012).

While the algorithms have numerous pros and the algorithms quality level develop over time, the most significant drawback is the point of a startup in recommendation technique, as there are several items and objects provided in the system whereas there are limited customers and few or nonexistence of rankings. This drawback termed "cold start" and means that the system of recommendation cannot generate any recommendations or suggestion for a new user (Schein et al, 2002). Solutions for resolving this drawback have varied over time. Some have included seeding the system by usage of other data sets, and utilization of algorithms of system recommendation that are dissimilar in start-up stage which fail to suffer from "cold start" issue.

The simplest way to mitigate the cold-start user issue and make a quick profile of a new user has been to ask for explicit ratings through the presentation of items to the user. This has been found to stimulate initial data about the new user through a quick and short interview. Once some items have been presented to the new user, this process is ended and, while in the user-item matrix the new user row is not empty, the normal level of a recommender system is entered by the new user. The CF recommender system should utilize these ratings to compute resemblance between existing and new users (Messina et al. 2018).

While a few effective techniques to cope with the cold-start problem have been suggested, it is yet not a cornerstone. During the selection of items, the newly received ratings of other users are not taken into account. Hence, a future direction can be creating a new method, which will fill this identified gap and acclimatize to the earlier ratings provided by other users.

2.1.2. Content-Based Models

In this model of filtering, systems implementing the model approach scrutinize a set of descriptions or documents of items rated previously by a user and construct a profile or model of user interests centred on the characteristics of the items rated by that user (Aggarwal, 2016). Consequently, the profile is a structured depiction of user interests, approved to recommend novel interesting objects. The process of recommendation essentially consists in matching up the features of the user profile compared to the characteristics of a content object (Rao & Talwar, 2011). The outcome is a relevance judgment that signifies the level of the user interest in that object. Furthermore, if a profile reflects the user preferences accurately, it is of remarkable advantage for the efficiency of an information access procedure (Zhou & Li, 2010). For example, it could be applied to filter search results by determining whether a user is attracted to a particular Web page or not and, in the negative scenario, inhibiting it from being displayed. Algorithms that give content-based (CB) recommendations do not need user ratings except for the target user. Thus, these techniques can be utilised in cold start scenarios if there is information available regarding the preferences of the user. In the most drastic cases, when a new item has been included in the catalogue, CB methods enable recommendations as they are capable of using features they have extracted from this new item and utilise them to give recommendations (Mizgajski and Morzy 2018).

Content-based Filtering systems require suitable techniques for generating the user profile and representing the items, and some approaches for comparing the profile of the user with the product representation. Thus the content based recommender system needs a high-level architecture. In this architecture, the process of recommendation is accomplished in three phases, with each being handled by a distinct component; Content Analyser - When data has no structure (for example text), some sort of pre-processing phase is required to extract relevant structured information. The key duty of the component is to epitomize the content of items (for instance documents, news, Web pages, product descriptions, among others) coming from sources of information in a form appropriate for the next phases of processing. Items of data are scrutinized by feature procedures of extraction so as to transfer a representation of an item from the original space of information to the target one (for instance Web pages characterized as keyword vectors). This component is the input to the profile learner and filtering component. Profile learner – This component collects information representative of the preferences of the user and attempts to generalize this documents, in order to generate the user profile (Mizgajski and Morzy 2018). Normally, the generalization approach is achieved through machine learning systems (Mahata, Saini, Saharawat, & Tiwari, 2017), which have the ability to gather a model of user interests beginning from items disliked or liked in the past. For example, the profile learner recommender for a Web page can implement a relevance method of feedback in which the technique of learning combines vectors of negative and positive examples into a sample vector which represents the user profile (Pazos-Arias, Vilas & Rebeca, 2012). Training illustrations are Web pages on which a negative or positive feedback has been offered by the user (McAuley et al., 2015).

Many different techniques of content-based filtering have been suggested that have been intended for reducing the cold-start issue in content-based filtering. A research from Princeton University in the U.S. suggests a method that employs CBF to cope with the cold start problem (Wang & Blei, 2011). They desired to create a machine learning algorithm for suggesting scientific articles to online community users. Their algorithm employs two sets of data: the libraries of articles of the community users, and the content of each article. The objective of the system is to both suggest older papers that are vital to others within the community with alike article taste, and simultaneously propose new papers that would be appropriate to the user. When proposing papers from other users with comparable taste, they use CF centred on latent factor strategy. This technique works well for proposing papers that are popular, nevertheless cannot be used to propose articles that have not been read yet. To cope with this they employ content-analysis centred on probabilistic topic modeling, and can subsequently recommend papers with comparable content as the papers loved by the user in the past. The subject modeling of the papers provides a subject representation of the items to determine the key themes in each paper and consecutively helps the system create intelligent recommendations for papers prior to anyone rating them.

These two techniques are combined subsequently in a probabilistic model where the choice of which item to propose is centred on the conditional anticipation of hidden variables. The anticipation is influenced equally by the content from the papers and the libraries of all the users, nonetheless, in scenarios where the item is new, the suggestions are centred on the content. This technique deals with new items and the issues associated. However, it does not mitigate the problems of new users. Stern, Herbrich, and Graepel (2009) suggest coping with the challenge through the usage of meta-data about each user to propose items popular in the demographic group of the user such as gender, age, and occupation.

2.1.3. Hybrid Recommender systems

These systems are a mix of single recommendation techniques as sub-components. This hybrid technique approach was introduced to hack the issue of conventional recommendation. Two core issues have been addressed in this field by researchers, the issue of cold-start and plasticity versus stability (Aggarwal, 2016). The cold-start problem arises when learning based methods like collaborative and content-based algorithms are used. Their stages of learning are established on users' information, in general, a user has to place their preferences or ratings manually and consequently makes the information gathering hard to achieve (Duzen & Aktas, 2016). Stability/plasticity issue means that at times it is hard to change reputable users' profiles that have been already proven after a given period of time via the systems. The cold-start issues may be resolved using the hybrid methodology since various types of recommendation methods like knowledge-based algorithm may be less affected by the issue. Among the solutions for the stability/plasticity issues is a temporal discount, whereby older ratings are made to have less influence (Hernandez-Rubio et al. 2018).

Consequently, various techniques of hybrid recommendation have been created and tested (Li & Kim, 2003). The hybrid systems are constructed on four major traditional recommendation techniques; content-based, collaborative filtering, demographic, and knowledge-based. Contrasting the first three which utilize learning algorithms, the knowledge-based exploits domain information and makes suggestions about users' preferences and needs. Hybrid recommendation techniques can yield outputs which outperform solitary component techniques through a combination of multiple techniques. The most popular hybridizing approach is merging different systems of different types, for instance, mixing content-based and collaborative filtering. Nevertheless, it is as well possible to combine dissimilar systems of the same type, like k-nearest neighbour content-based and naive Bayes content-based. Similarly, combining the same type of methods with dissimilar datasets can be possible (Fortes, Freitas & Gonçalves, 2017).

Burke and his associates attempted to compare the performance and effectiveness of this numerous types of hybrid systems of recommendation (Burke, 2002). They had instigated a system of recommendation known as Entrée, a recommendation system for a restaurant created by the concept of case-based reasoning. This system practices the usage of interactive critiquing interchange between the user trying to find out appropriate restaurants and the system. This technique is not like searches, which attempt to narrow through the addition of contents, but shifting the emphasis in the feature space, almost like browsing. They provide also Entrée dataset which as well is available publicly. In this dataset, the interchanges represent negative or positive user ratings like the critiques are negative rankings, while entry and ending point is positive rating. Consequently, this dataset comprises generally negative ratings and its size is rather small, which can be considered as shortcomings (Burke, 2002).

According to the results of the experiments, the hybrids exhibited dominance over conventional recommendation techniques. This interaction was established under scenarios like with smaller session dimension, the sparse density of recommendation. This outcome means that hybridization can mitigate cold start problem which was inherent in some conventional systems of recommendation. The best hybrids were cascade hybrid and feature augmentation. Feature augmentation permitted a contributing recommender to create positive influence without meddling with the better algorithm performance. Cascade hybrids have been established to be rare in literature but demonstrated to be effective for coalescing recommender with various strengths. Moreover, the knowledge-based technique was established to be good for contributing or secondary components and may be combined severally to create hybrids. Lastly, the literature indicates that diverse hybrid components have relative consistency and accuracy and their features should be meticulously considered so as to create effective hybrids (Said, 2010).

Noia et al. (2013) demonstrate that Linked Open Data (LOD) has the prospect to be used effectively in content-based recommender systems that successfully overcome the cold-start problem. They further specify a content-based recommender structure that utilizes LOD datasets, for example, DBpedia, LinkedMDB, and Freebase to recommend movies. Moreover, they use these LOD datasets to collect contextual data about movies such as directors, actors, and genres and then employ a content-based recommendation technique to produce recommendation results. Similarly, Ostuni et al (2013) suggest a hybrid LOD-based system of recommendation that exploits users' embedded feedback and is constructed on top of DBpedia. Semantic data about items in the profile of the user and items in DBpedia are combined into an amalgamated graph to mine path built features for the algorithm of recommendation. Congruently, Ostuni et al (2013) as well propose a content-based recommender system that produces semantic item likenesses using DBpedia. The semantic likeness between items is computed through a neighbourhood-based graph kernel that identifies the items local neighbourhoods. Evidently, there is a very vigorous research community concentrating on the application of LOD sources to RSs.

2.2. Music Recommendation Systems

One of the key problems facing Music Recommender systems MRSs, like all other RS, is the cold start problem. In this case, when a new user makes registration to the system or a new song is added to the collection and the system has insufficient data linked with these users/songs. In such a scenario, the system fails to suitably recommend existing songs to a new user (i.e. new user-problem) or suggest a new song to the existing users (new song problem) (Adomavicius &

Tuzhilin, 2005; Elahi et al., 2016; Kaminskas & Ricci, 2012; Schein et al., 2002). An additional sub-problem of cold-start is the earlier discussed sparsity problem which in this case refers to the element that the amount of given ratings is much inferior compared to the possible number ratings, which is most likely when the amount of users and songs is large (Elahi et al., 2017).

A number of techniques have already been suggested to mitigate the cold-start problem in the MRS domain, these have included the earlier discussed approaches focusing on content-based, cross-domain recommendation hybridization, and active learning. Besides the aforementioned techniques, active learning has demonstrated promising results as far as MRS are concerned in coping with the cold-start problem. Active learning tackles this problem at its foundation by eliciting and identifying (high-quality) data that can signify the user preferences better than by what they themselves provide (Elahi et al., 2016, Rubens et al., 2015). Such a system thus interactively requires specific user response to maximize the enhancement of system presentation. Given that these approaches were discussed earlier in this chapter, this section will focus on their limitations as far as MRS are concerned.

Nowadays, the majority of the commercial music recommender systems are approximately using the users' preferences to deal with the long tail problem (the large no. of songs). The common characteristics in these systems are constant when using users' preferences compared with users' context (location, mood, weather, etc.). For instance, in the library when people are sitting there maybe they need quiet and melodious music to listen according to the environment where they are in. Last.fm, Allmusic, Spotify, Pandora and Shazam are commercial music recommendation systems which are considered to be excellent systems by focusing on the music already played in order to help the users to find more music. Users are able to connect to a web-based music streaming service to access the recommendations. All the tracks that are played on this stream are recommended. As with a "random" broadcast, the users are able to tell the system if they are interested in the track being played or they want to ban it. The two types of recommendations streams are for subscribers and for non-subscribers. The precision of the recommendation algorithm will vary, depending on whether the user is a subscriber or not. For non-subscribers, all the tracks are chosen according to a similar profiles group. The music stream is broadcast for subscribers and its content is governed by the user profile only. The tracks on the personalised stream are expected to match the user's preferences more closely. An open source project called Audioscrobbler.com uses complicated functionality and expensive infrastructure to act as the data harvester for Last.fm (Elahi et al., 2016, Rubens et al., 2015). These websites provide a unique platform to retrieve rich and useful information for user studies but they are not enough to get users' satisfactions (Song et al., 2012). These systems are considered as music catalogue providing users with personalized recommendations based on their taste in music. MyStrands (desktop app which allows the recommendations of similar songs, albums, and artists) is a great music recommender system which based on songs/artists uploaded either from iTunes playlists or added as favourites on the site. It is Based on songs or artists which users either upload from your iTunes playlists or add as favourites on the site where users start managing their library of music with tags and keep tracking of the music the friends who listening to and getting multiple recommendations per song played. Additionally, this app filters recommendations by decade, genre, and popularity, as well as builds fabulous playlists (Song et al., 2012). Many researchers have paid great attentions to contextual information gathered from the sensors attached to the mobile smart phones and utilized these information in music recommender systems to satisfy the users' needs (Jannach et al., 2017).

Kim et al. (2010) used collaborative tagging employed as an approach in order to grasp and filter users' preferences for items and they explored the advantages of the collaborative tagging for data sparseness and a cold-start user (they collected the dataset by crawling the collaborative tagging delicious site). Weng et al. (2008) combined the implicit relations between users' items preferences and the additional taxonomic preferences to make better quality recommendations as well as alleviate the cold-start problem.

Khrouf et al., (2013) presented EventMedia which is a web-based environment that exploits real-time connections to deliver rich content describing events associated with media and interlinked with the Linked Data cloud to help user to attend upcoming events, and inter lined with the Linked Data cloud. It is a pure service to satisfy daily users and developers. Kharouf used the semantic web and produced a novel hybrid recommender system and also used LD to avoid the sparsity in the recommendations. Zangerle et al. (2012) showed that tweets can be exploited to build a corpus for music recommendations. Therefore, Ash (2012) proposed a Twitter Music which is a smart phone app pulls the music from iTunes Radio, Rdio and Spotify where the app retrieves the most popular artists from the follower lists and analyses them to give the user the best possible music. This music app provides users artist and song suggestions based on who the user followed on the Twitter. On the other hand, a recommendation system named HAAPL was designed to gain a very high accuracy and novelty. Onuma et al. (2009) proposed a TANGENT, a recommendation algorithm which considered serendipity. It is a “surprise me” query that gives a user a recommendation which is related but not regular. This proposed treated the recommendation as a node selection on a graph and its goal to select nodes that not only connect to nodes with high scores but also connect to unrelated ones. Wang et al. (2014) proposed a hybrid recommendation method to find suitable music according to the user’s context by using last.fm, yahoo local, twitter and Lyricwiki and linked these data resources by using Linked Open Data (LOD) technology and used the serendipity of the music recommender system to increase the accuracy of the recommendation. This way is considered to be a novel in recommendation system to avoid the cold start and sparsity problems and also it is new way to discover songs and artists which closed to the user preferences (Lu et al. 2018; Zhang et al. (2017).

Meanwhile, many researchers have used social media (Twitter & Facebook) to identify user’s mood (tension, depression, anger, vigor, fatigue, confusion) and also identify user’s personality (openness, conscientiousness, extraversion, agreeableness, neuroticism) where these are very important factors which influence on user’s music taste (Wang et al., 2014; Roberts et al., 2012; Pandarachalil et al., 2015; Ross et al., 2009; Bachrach et al., 2012; Back et al., 2010) and also contextual features (location & event) can lead to different emotional effects due to objective features of the situation or subjective perceptions of the listeners (Scherer et al., 2001). Music lyrics are also considered to be one of emotional presentation because they include some kinds of implicit thinking, thus we can fully understand emotions and their associated thinking in each song (Nunes and Jannach, 2017; Tintarev and Masthoff, 2008).

Cano et al. (2017) mentioned that there is a strong relation between the user mood and listening to the music. The people may want to listen to music which has the same mood of them when they are in specific mood and in contrast the people want to listen to different kind of music which encourage them to enhance their mood and this thing depend on the psychological studies and therefore, the author produced a contextual mood-based music recommender system which is able to regulate the driver’s mood and also try to put the driver in a positive mood when driving because listening to the music while driving has always been one of the most favourite activities carried out by people.

Finally, similarly, active learning approaches suffer from various limitations. Firstly, the usual active learning approaches recommend to the users the songs with the highest forecast ratings so as to provoke the true ratings. This certainly is a default approach in all RS as users have a habit of rating what has been suggested to them. Furthermore, users normally browse and rate stimulating songs which they would like. Conversely, it has been demonstrated that doing so generates a strong preference in the dataset and increases it excessively with high ratings. This consequently may influence substantially the algorithm for prediction and reduce the accuracy of recommendation (Elahi, Ricci & Rubens, 2013). Furthermore, not all approaches to active learning are necessarily personalized. The users very much have a difference in the amount of data they have about the songs, their preferences, and decision-making process. Thus, it is undoubtedly inept to appeal all the users to rate the same set of songs, since it is likely that many users may have quite a limited knowledge base, ignore many songs, and not correctly give ratings for these songs. Properly created active learning approaches should take this into account and suggest different songs to different users to rate. This can be very beneficial and escalate the chance of obtaining ratings of higher quality (Elahi, 2011).

2.3. Factors that Affect Music Recommendation

2.3.1. Music-related factors

It has been found that listeners across a wide range of ages prefer a fast and lively tempo in a range of musical genres, from classical and jazz, to pop and folk (Teo, 2003). It has further been identified that tempo preference interacts with other variables, such as affective association of tempo, musical styles, the ability to discriminate tempo, performance medium, and subdivision of beats. Additionally, music with a clearly defined rhythm and a regular meter and consistent and easily identifiable pulse is generally preferred to music with an irregular and harder to follow rhythm. Music with a moderately complex rhythm is also usually preferred to that with a rhythm perceived as being overly complex or simple. In terms of pitch, studies have shown that pitch preference is closely related to the ability to determine it. A further important correlating factor is pitch intensity.

The term 'timbre' refers to the sounds of particular instruments, which have a significant effect on listeners. Thus far, there is no known scientific explanation for why people prefer the sounds of particular instruments, though it is considered likely that preferences vary across cultures and musical genres, and that there is a high level of individual difference (McDermott, 2012). Nevertheless, studies have found a general preference for instrumental rather than vocal timbre, particularly in classical and non-western traditional music. One notable exception is pop music, where, by contrast, listeners tend to prefer the vocal timbre.

In addition to its auditory and perceptual features, music also has the ability to conjure referential meaning (McDermott, 2012), whereby the melody, rhythm, harmony, and mode of a piece of music generally convey particular emotional content and ideas, perceived by the listener (Koelsch et al., 2004). These messages can ultimately direct the recognition and enjoyment of music's aesthetic value (Finnas, 1989). Its emotional content is frequently argued to be one of the primary reasons listeners consume music, and they will generally prefer music that invokes an emotional response over that which does not. In this way, musical preferences are affected by both the emotional content of the music itself, and the emotional response of the listener, and listeners can use music to achieve mood regulation, to enhance, or even alter, emotional state.

Another key factor influencing musical preference is complexity, whereby musical stimuli that are perceived by the listener to be overly simple or overly complex may be less aesthetically pleasing; generally, moderate complexity is associated with the greatest appreciation in listeners (McDermott, 2012; North & Hargreaves, 1995). Generally speaking, there is a U-shaped correlation between complexity and preference in regard to musical stimuli (Berlyne, 1974). Studies have measured complexity based on a range of factors, including degree of syncopation, number of chords, human ratings, and temporal correlation of melodic sequences. Understandably, then, the perception of complexity, and its relation to preference, will depend on the listener and their level of expertise or musical training. Research has found that listeners with greater musical ability are more likely to prefer music of greater complexity, and also that musical expertise diminishes the impact of complexity on musical preference, giving greater influence to other aesthetic features. Furthermore, familiarity can influence appreciation, where repeated listening can increase the enjoyment of more complex music.

Rentfrow et al. (2011) identified five latent dimensions of musical preference, in a study that measured listeners' reactions to musical excerpts across a large variety of styles and genres. Based on the results, the researchers identified five dimensions: 1) mellow (smooth, relaxing styles); 2) unpretentious (sincere, 'rootsy' music, e.g. singer-songwriter genres, country music); 3) sophisticated (opera, classical, jazz, and world music); 4) intense (energetic, loud, and forceful styles); and 5) contemporary (typically percussive and rhythmic, such as rap, acid jazz, and funk). These dimensions are based on both psychologically-oriented (complex, aggressive, intelligent, inspiring, romantic, sad, relaxing) and acoustic (distorted, instrumental, dense, fast, loud, electric, percussive) attributes.

Lastly, as already mentioned, prior exposure to, or familiarity with the music has a significant impact on preference (Finnas, 1989; McDermott, 2012; North & Hargreaves, 2008), where a listener is typically more likely to prefer music they have heard previously, and to dislike unfamiliar music. Familiarity is particularly able to explain cultural variation in musical preference, where listeners will typically prefer music produced within/by their own culture. Nevertheless, upon first listen a listener may still dislike a particular track within a familiar genre and culture, but they are more likely to develop appreciation for it after repeated listening. Generally, familiar music is preferred to unfamiliar music, across genres, however repeated listening can increase enjoyment of unfamiliar music from a similar genre and/or culture.

2.3.2. Listener-related factors

Research has identified an apparently strong association between listener's age and music preference, where it is suggested that the importance of music to one's life increases until adolescence, before gradually decreasing over one's lifespan. Chamorro-Premuzic et al. (2010) further found that age has a negative impact on consumption of music. The correlation between age and music preference has been analysed in more depth by Holbrook and Schindler (1989), who found that listeners will continue to prefer music they were exposed to in the 'critical' period of their life, which is identified by the researchers to culminate at approximately 23.5 years old. This is confirmed in other studies that similarly identify the critical period as between 20 and 25 years of age. The effect of this period can possibly be explained by certain developmental experiences of the individual, such as social activities, any problems coped with, or identification with particular artists that were formative in shaping music preference.

Music preferences have also been associated with certain lifestyle preferences. For instance, North and Hargreaves (2007a, b, c) carried out a large-scale study of 25, 32 participants measuring the relationship between musical preferences and lifestyle factors. Participants were asked to provide information relating to their living arrangements, interpersonal relationships, education level, financial position, employment status, ethical and political stances, criminal behaviours, hobbies, media preferences, travel history, music consumption, and health-related factors such as drinking and smoking habits. The study found that participants' music preferences could be used to distinguish certain lifestyle choices. Specifically, the researchers were able to broadly categorise stimuli, i.e. literature, leisure, media, and music preferences, into either intellectually undemanding (termed ‘low art’), or intellectually promising (or ‘high art’). Those who consumed more high art music (e.g. opera, classical music) showed a similar preference for high art in other areas of their life; and the reverse was also true, where consumers of low art music tended to prefer low art literature, media, and leisure interests. The researchers then related these findings to social status, and found that high art music was closely associated with upper-middle to upper class consumers, and low art music was more closely associated with lower-middle to lower class consumers. A liberal-conservative dichotomy was also identified. However, such a dichotomy is likely over-simplified, and there is a need for studies applying more complex approaches to analysing the relationship between music preference and lifestyle. Perkins (2008) and Schäfer (2008) have conducted extensive reviews of the extant research on this topic.

Lastly, it is necessary to emphasise the relevance of musical training, as briefly introduced earlier in relation to musical complexity. In terms of the way in which individuals listen to music, Kemp (1996) differentiates between objective-analytics and affective listening strategies, where the former is applied by musically trained listeners and includes technical or objective responses, whereas the latter is a more emotional, and typically a more musically naïve, appreciation. There is a correlation between these different listening strategies and musical preference, whereby music experts and novices respond to different features of a piece of music, rather than to the same features but at different levels.

2.3.3. Listener context-related factors

Music listening is often described as a social activity, which implies that music preference is a social phenomenon (Lonsdale & North, 2011; North & Hargreaves, 2008). Extant research supports the link between social ties and music preference, and the consumption of music as a form of individual social expression (Finnas, 1989; MacDonald et al., 2002; North & Hargreaves, 1999; Rentfrow & Gosling, 2006). This indicates that individuals might view music preferences to be an indicator of particular personality traits, even as an aspect of identity, signalling their personality to others, and enabling them to identify and connect with a particular social group; this is especially the case amongst adolescents, who discuss their music preferences more than other topics (North & Hargreaves, 1999; Rentfrow & Gosling, 2003).

It is interesting to note also that personal music choices are often markedly influenced by the opinions of others (McDermott, 2012). For instance, studies of online music distribution systems have revealed that user track ratings are highly dependent on the ratings of other users, even where users are unknown to each other. This conformity in music preference can be explained by the phenomenon of compliance, whereby people have a desire to belong to particular social groups that hold similar values and opinions, or enjoy similar activities. An individual might thus adjust their music tastes to those expressed by the members of a particular group (e.g., a group of friends), in order to gain membership and/or acceptance. A further possible explanation is the phenomenon of informational influence, also known as the ‘prestige effect’, identified by North and Hargreaves (1999). This effect occurs where individuals form a preference for unfamiliar music based upon the opinions of others, or on contextual information about that music, for instance a description of the artist.

Beyond specific social groups, the musical preferences of an individual can be more broadly dependent on their cultural environment (MacDonald et al., 2002; McDermott, 2012; Schäfer, 2008). A listener’s ethnicity or cultural background can impact on their perception of the particular aesthetic qualities of certain genres of music, or of specific pieces. In addition, the specific listening context or situation, such as presence or not of others, location, and any ongoing activities, can have a significant effect on music appreciation (Schäfer, 2008).

2.3.4. Music use-related factors

Music is utilised by listeners to serve a variety of human needs, whether emotional, sociocultural, physiological, or cognitive (Schäfer, 2008; Schäfer & Sedlmeier, 2009). In particular, music has been found to have substantial social communication and self-reflection functions. An individual might use music to change or strengthen social relations, to alter or enhance one’s own mood (Schäfer, 2008; Ter Bogt et al., 2010), for self-socialization (especially amongst adolescents), personal reflection, and potential alleviation of personal problems (Schwartz & Fouts, 2003). Rentfrow and Gosling (2003) argue that, based on an understanding of music’s wide-ranging functions, the key to understanding individual listening behaviour may be the individual benefit derived from it. However, current experimental research has not provided any clear conclusions regarding the use of and role of music in individuals’ lives, indicating the complexity of this issue (Schäfer & Sedlmeier, 2009), in which the short-term context of the listener may be an important factor.

Notably, a study carried out by Chamorro-Premuzic et al. (2010) found that motivation for listening rather than demographic or personality differences in individuals is a better predictor of music consumption. The researchers developed and applied the Uses of Music Inventory to evaluate three different motives for listening to music, as follows: emotional use (i.e. mood inducing); cognitive use (i.e. enjoyment of intellectual/rational analysis of music); and background use (i.e. enjoyment alongside another activity, e.g. socializing, studying, working). The findings showed that all music uses had a positive effect on consumption. Similarly, a study by Ter Bogt et al. (2010) proposed a Typology of Music Listeners based on the listener's degree of involvement with music and four music usage types: mood enhancement; coping with problems; defining personal identity; and marking social identity. Of these, emotional use of music, i.e. for mood enhancement, was found to be the most prevalent amongst all listeners across all levels of musical involvement.

Zillmann (1998) acknowledged that a broad collection of information consumption from music, news, movies, and documents are impacted by user's mood. The concept has been further scrutinized in the research community of mood management (Knobloch-Westerwick, 2006). In specific, selection of music is characterized by self-indulgent motivations to either mend their negative mood or preserve their positive mood in terms of both duration and intensity. In regard to this, a user's emotional state serves as a significantly useful predictor of their decisions of music. While a selection of genres can possibly drive entertainment choice, tragic or sad contents are perhaps more likely to be circumvented by a large number of users; while funny or light-hearted music is mostly sought after (Oliver, 2008; Schaefer et al., 2005). Similarly, Tesser et al (1998) established three key motivations activating users' movie-going behaviours: entertainment, self-escape, and self-development. The former two appear to be consistent with self-indulgent considerations for users to mend or preserve positive mood; while the latter appears to be the least correlated to self-indulgent considerations, in distinction, reflecting users' eudemonic motivations in pursuing greater meaningfulness and insight towards life for self-reflection (Waterman, 1993). Consequently, a wider collection of movies have been confirmed to establish the connections between users' mood states and preferences of entertainment (Oliver, 2008).

2.4. Mood Categories

The cataloguing of music can be a challenging task given the emotional response between listeners can be fairly dissimilar for a given track (Eerola, 2012). Majority of the current classification for music is established on an artist's general genre, rather than on the emotion created by a song. Attempting to classify music through techniques of engineering is challenging, however, can possibly assist to minimize these incongruities between users in the sorting process. Moreover, categorizing the mood of a track automatically would be tremendously useful for organizing large groups of digital music like those of Spotify or iTunes (Bhat et al., 2014). The mood of a song could as well enhance algorithms for categorizing similar songs for MRS, basing the comparisons on the song's mood instead of similar artists. Furthermore, breaking a track down into computable musical components such as harmony, rhythm, and timbre can permit for the matching of tracks to specific groupings based upon anticipated data for each category of mood.

In the majority of the existing techniques of music mood categories, the song moods are divided conferring to Robert Thayer's traditional mood model (Bhat et al., 2014). This model distributes music along the lines of stress and energy, from sad to happy and energetic to calm, in that order (Bhat et al., 2014). The eight categories established by Thayer's model comprise the immoderations of the two lines in addition to each of the conceivable intersections of the lines (for example sad-calm or happy-energetic).

Faster tempos are connected with high-energy tracks, while slower tempos are linked to lower energy, sadder songs. Furthermore, intensity or loudness of a track can be linked with anger, whereas softer songs would advocate sadness, tenderness, or fear (Bhat et al., 2014). Moreover, the overall higher pitch can be an indicator of carefree, happy, and light moods within a tune, whereas lower pitch, suggests a darker, serious, and sad tone. On the other hand, timbre, the tonal element of a song produced by harmonics, is an inquisitive indicator of mood. As pointed out by a group of scholars from the BNM Institute of Technology in Bangalore, India, timbre kindles human energy levels with no regard to harmonic or rhythmic saturation (Bhat et al., 2014). Additionally, sources of sound that have guileless vocal profiles have timbres that are 'darker' and have a tendency to soothe human feelings (Bhat et al., 2014). This same group of scholars formed a correlation table of timbre, intensity, rhythm, and pitch in identifying numerous moods.

In a different research by Hu, Downie, & Ehmann (2009), an examination on what part lyric text can play in the improvement of audio music mood grouping. The authors proposed a new method to construct a huge ground truth set of 5,585 tracks and 18 categories of mood centred on social tags so as to mirror a convincing, user-centred perception. A comparatively comprehensive set of lyric features and models of representation were scrutinized. The best performing set of the lyric feature was as well compared to a prominent audio-based system. In a combination of audio and lyric sources, sets of hybrid feature created with three dissimilar feature methods of selection were as well examined. In comparison to function words and Part-of-Speech, Bag-of-Words was established to be the most convenient feature type. Nevertheless, there was no noteworthy difference between the choice of not stemming or stemming, or amongst the four text depictions on regular accuracies across all groupings (Hu, Downie, & Ehmann, 2009). Their comparisons of audio, lyric, and combined features identified patterns in conflict with preceding studies. They established that particular lyric features solely can outperform audio components in groupings where samples are scanter or when semantic denotations taken from lyrics relate well to the mood grouping. Similarly, they established that combining audio and lyrics features improves performances on the majority, however not all, categories. Tests on three diverse feature selection techniques revealed that too many features of the text are indeed noisy or redundant and the combination of audio with the most relevant text features may contribute to higher precisions for the majority of mood categories (Hu, Downie, & Ehmann, 2009).

2.5. Summary

This chapter has laid out the background into three main research areas developed in this thesis: recommender systems which include the main approaches and the challenges, music recommender systems and the factors effect on that and the mood categories and how the user current mood can be a good factor in the music recommendation systems. It has discussed how this research is related to previous researches in context-aware music recommendation systems.

In almost all the existing recommendation systems personal information must be collected in advance in order to build the recommender system, which creates a time-consuming and problematic issue known as the 'Cold-Start' problem. The Cold-Start problem in relation to music recommendation is where the system is unable to recommend songs to new users who do not yet have a listening history; they must first consume a number of songs by listening to them. On the other hand, as most existing music recommender systems recommend the songs that are most liked by the majority of users, it is difficult to recommend new songs to users because these songs are new and therefore do not have enough ratings to be recommended. By contrast, the current music recommendation systems are more than sufficient for recommending similar music but this similar music cannot meet consumers' taste. Clearly, developing the appropriate techniques to address Cold-Start problem and improve recommendation accuracy are needed. We will focus on how to achieve this goal in the rest of this thesis.

Chapter 3

Context-Aware Personalised Music (CAPM) Framework

Music has always played an essential role in human entertainment. With the increase in digital music and Internet technologies, a significant amount of music content has become available to tens of millions of customers around the world. With many thousands of artists and songs on the market, it is becoming increasingly difficult for customers to search for and discover interesting and novel musical content. Furthermore, the huge quantity of music data available has opened up possibilities for researchers working on track statistics retrieval and advice to create new practicable services that guide track navigation, discovery, and sharing, and the formation of consumer communities. The demand for such services – commonly known as song recommender structures – is high, and therefore the conceivable market for online music content is huge. Music recommender systems are decision support tools that reduce the information overload by retrieving only the items that are expected to be relevant to the user, based on the user's profile.

As mentioned in the literature review in this thesis that there are many music recommender systems currently available but these systems fail in addressing the cold start problem in addition to fail in providing music according to the users individual needs. Therefore, this chapter presents a Context-Aware Personalised Music (CAPM) framework which is a generic framework for supporting the development of context-aware music recommendation systems, which uses user-profiling method to automatically extract a user's contextual information from multiple sources. The motivation underlying this approach is to help users find novel music while avoiding the Cold-Start problem associated with recommender systems.

3.1. System Framework and Design

CAPM framework is proposed to address the Cold-Start problem, in addition to dynamically providing personalised song recommendations to the existing users depending on their context.

Figure 3.1 outlines the major components of CAPM framework, which is built on a client-server architecture.

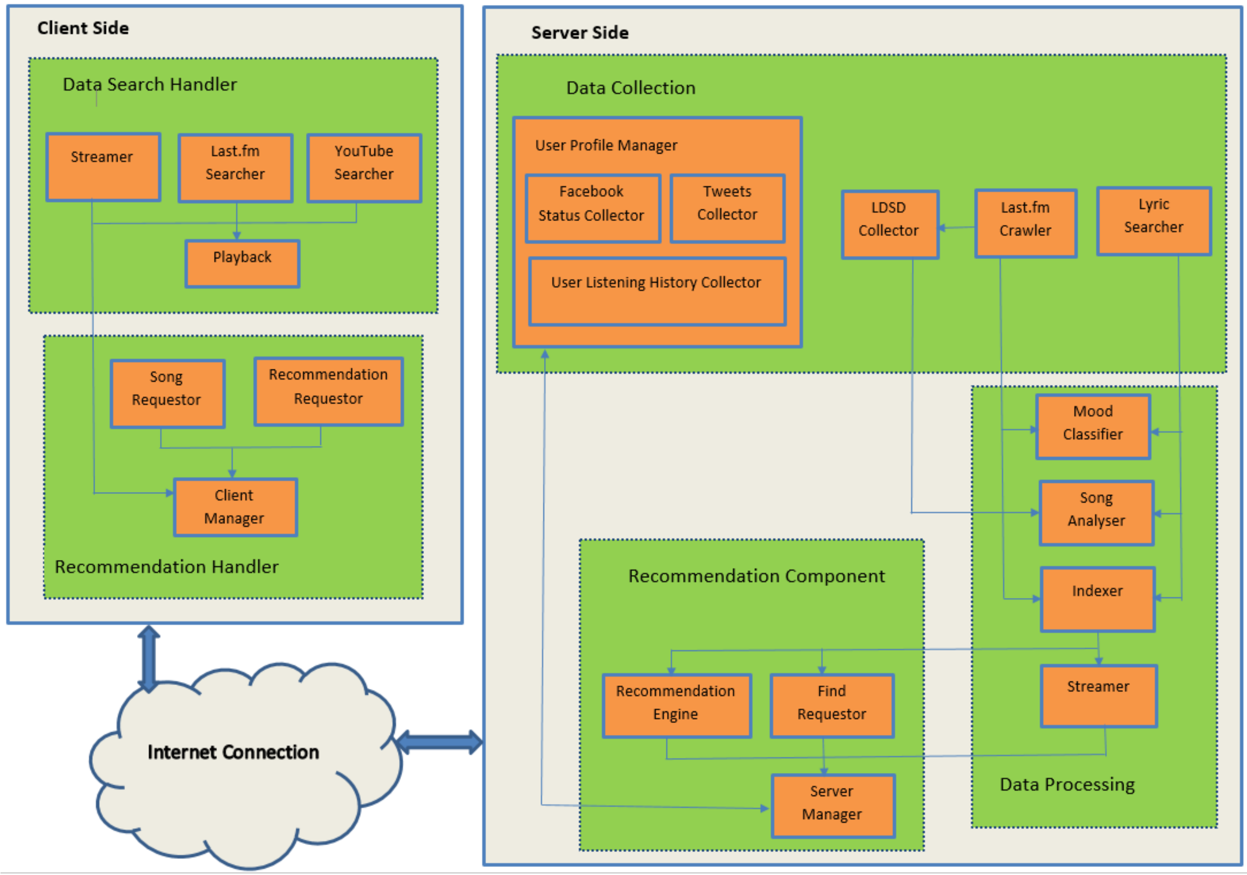


Figure 3.1: CAPM Framework

In the server side, there are three components: data collection, data processing and recommendation component. Data collection is to collect relevant data from different sources, for instance, collecting lyrics from chartlyrics, collecting music related information like artists from Last.fm and collecting semantic distance between artists from dbpedia. User profile manager is to dynamically extract and update the user context from Twitter and Facebook and also collect the user listening history. Data processing is a key component, which consists of mood classifier, song analyser, indexer and streamer. Recommendation component recommends songs to the users using the hybrid recommendation algorithms developed in this research.

On the client side, Data search handler is to dynamically download relevant lyrics and stream the music. Recommendation handler connects to the server to recommend music to the user.

Each component in the server side and the client side has a number of modules transferring the data to accomplish the work.

The CAPM framework helps users discover new artists, albums or songs making this music information accessible. The dynamic characteristics of the interface allows the user to browse music collections while listening to a song, album, or playing a video. The users will receive information related to their interaction patterns (profiles) as personalised recommendations of items, which probably they would like, while they use the application.

3.2. Server Side

The CAPM framework runs on a client-server system. The server side is made up of different components as seen in figure 3.1.

3.2.1. Data Collection

The data are considered the key component of any recommendation system. These data can be collected by several means, such as individuals' ratings of items, and reviews/feedback from customers. These data are used as the foundation of recommendations to users. The data collection in this framework relies on three modules: a Last.fm crawler, a lyrics searcher, LDSD Calculator and a user profile manager. Each module is responsible for a specific task, as shown below.

3.2.1.1. Last.fm Crawler

Last.fm is an online digital music service available since 2002 (Vigliensoni, 2017). It offers a public API that provides researchers with a good opportunity to build their own programs using Last.fm data. The Last.fm API also allows calling methods that respond in REST style XML or JSON. The purpose of Last.fm crawler module is to collect all forms of data from Last.fm, such as artists, songs, albums, and tags which have the following main features:

- Artist tags retrieval
- Artist albums retrieval
- Artist tracks or songs retrieval
- Artist basic statistics retrieval

The task of the Last.fm crawler module is to keep a recent list of all the music files, updated by reading the tags field from .mp3 files that populate the song library. This gathers data on all artists and their associated information from Last.fm and redirects that information to the specific databases present in the database server. It also identifies and interprets any missing or incorrect information relating to a song by using the Last.fm API, and using the crawler to locate information about the track by locating its metadata.

The additional information relating to songs, such as the date of release and the album details can help to provide more accurate song suggestions to users. The essential function of this component on the server side is to provide the user with complete information relating to a song. The user credibility is increased, as better recommendations can be provided for the user through a system that can help provide quality content and suitable music choices for the user. The system uses 'track.getInfo' to obtain song information, which can function as user metadata.

3.I. Last.fm Crawler

1. While there are songs to be updated
2. Go through the .mp3 file tags
3. Establish a connection to the Last.fm via Last.fm API
4. If the tags are empty then
5. Collect the tags from Last.fm and update the songs tag
6. End if
7. End while

3.2.1.2. Lyric Searcher

Lyric information plays a very important role for identifying music, as there are no two songs with exactly the same lyrics. Access to song lyrics is provided by various websites, such as Lyricwiki, Gracenote Lyrics, and EvilLyrics, amongst others. Previous researchers have found that the greatest difficulties in automatically retrieving lyrics are related to the display, formatting, and content of the lyrics (Knees, 2005).

ChartLyrics is a free website that is considered a good source from which researchers can obtain reliable lyrics for any song, by any artist (Cavallaro, 2010). The task of the Lyric Searcher module is to find the lyrics of a given song. When song lyrics are searched, ChartLyrics sitemap will be queried for the current song, and if it matches, the Lyric Searcher module will be connected to the matching URL. When it is connected to the web page, the lyrics are retrieved and then reserved in the index. However, the recommendation algorithms in this research were developed with the objective of providing song lyrics of the song, therefore it is necessary to identify the lyrics of all the tracks stored in the system. The accuracy of the results is increased by using ChartLyrics, as this site is able to provide the lyrics to approximately 18,000,000 different songs. Any search made of this source thus has a high chance of detecting the correct lyrics. The Lyric Searcher module will search the lyrics of every song in the music list. There is a chance that it will be unable to find the lyrics, for instance for instrumental music, however the user relevancy can be identified with reference to information related to the album in order to produce recommendations. Furthermore, the module automatically connects to the URL if any match for the lyrics is found. The lyrics of the song are stored in the index after they have been extracted from the source.

3.II. Lyric Searcher

```
1.  Establish connection to ChartLyrics sitemap
2.  Obtain the list of song sitemap
3.  While there is still song in ChartLyrics sitemap do
4.      Download the song sitemap
5.      Include song sitemap in the list of sitemaps
6.  End while
7.  While lyrics for song S in the song list, do
8.      Obtain a connection to the specified URL by the sitemap list
9.      Download the lyrics from the website and index it in S
10.     Otherwise
11.         Remove S from song list
12. End while
```

3.2.1.3. LDSD Calculator

The linked data semantic distance (LDSD) plays an important role in the music recommender system. This module is responsible for extracting the links between the artists from DBpedia ontology (Passant, 2010) and determining the LDSD between the artists in the database, to be used in the next stage in the data processing when the similarity between the songs is calculated. However, DBpedia already provides a large database for such resource recommendations, as there are more than 39,000 distinct instances of dbpedia-owl:MusicalArtist or dbpedia-owl:Band provided by DBpedia. Essentially, the proposed algorithm uses a seed URI as input to calculate the linked data semantic distance between this URI and all other resources from the dataset. The distance is computed using the LDSD algorithm and is indexed. In this module, two types of links are taken, dbpedia-owl:MusicArtist, and dbpedia-owl:Band. These links are vocabularies in dbpedia ontology to denote the connection between artists. They are specifically chosen because Content-Based musical recommender system is implemented to recommend musical artists or bands based on the popular LOD dataset. Furthermore, the relevant results in music recommender systems have been achieved in previous research when using these two links (dbpedia-owl:MusicArtist, and dbpedia-owl:Band) (Passant, 2010).

3.III. LDSD Calculator

1. While there exists a song S in the song list, do
2. Generate vector for S artist name
3. While there are song to compare Sc in song list do
4. Generate vector for Sc artist name
5. Establish a connection to DBPedia sitemap
6. Collect the direct Band and MusicalArtist
7. Collect the indirect Band and MusicalArtist
8. Index the direct and indirect links
9. Calculate the LDSD (Equation 3.3)
10. End while
11. End while

In order to fully understand how the LDSD is calculated, there is a need to explain the direct and indirect distances between resources as shown below:

I. Linked Data Semantic Distance (Direct)

The links between resources in the graph can show relatedness, and the more links between the resources, the more relatedness indication there is. Thus, a direct distance between two resources exists when there is a distinct direct link (directional edge) between these two.

Definition 1: *"C_{direct} is a function that computes the number of direct and distinct links between resources in a graph G. $C_{direct}(l_i,r_a,r_b)$ equals 1 if there is an instance of l_i from resource r_a to resource r_b, 0 if not. By extension C_{direct} can be used to compute (1) the total number of direct and* *distinct links from r_a to r_b ($C_{direct}(n,r_a,r_b)$) as well as (2) the total number of distinct instances of the link l_i from r_a to any node n ($C_{direct}(l_i,r_a,n)$)"* (Passant, 2010).

The Linked Data Semantic Distance (Direct) can be determined as the total number of distinct direct links between two resources as shown in the equation 3.1:

$$LDSDdirect(ra,rb) = \frac{1}{1+Cdirect(n,ra,rb)+Cdirect(n,rb,ra)} \quad (3.1)$$

The direct connectivity between r_3 and r_2 is two (C_{direct}(r2, r3) = 2) because they are connected by l_1 and l_4, and the direct connectivity between r_2 and r_3 is zero ($C_{direct}(r_3, r_2) = 0$) as there are no direct links originating from r_2 as shown in figure 3.2

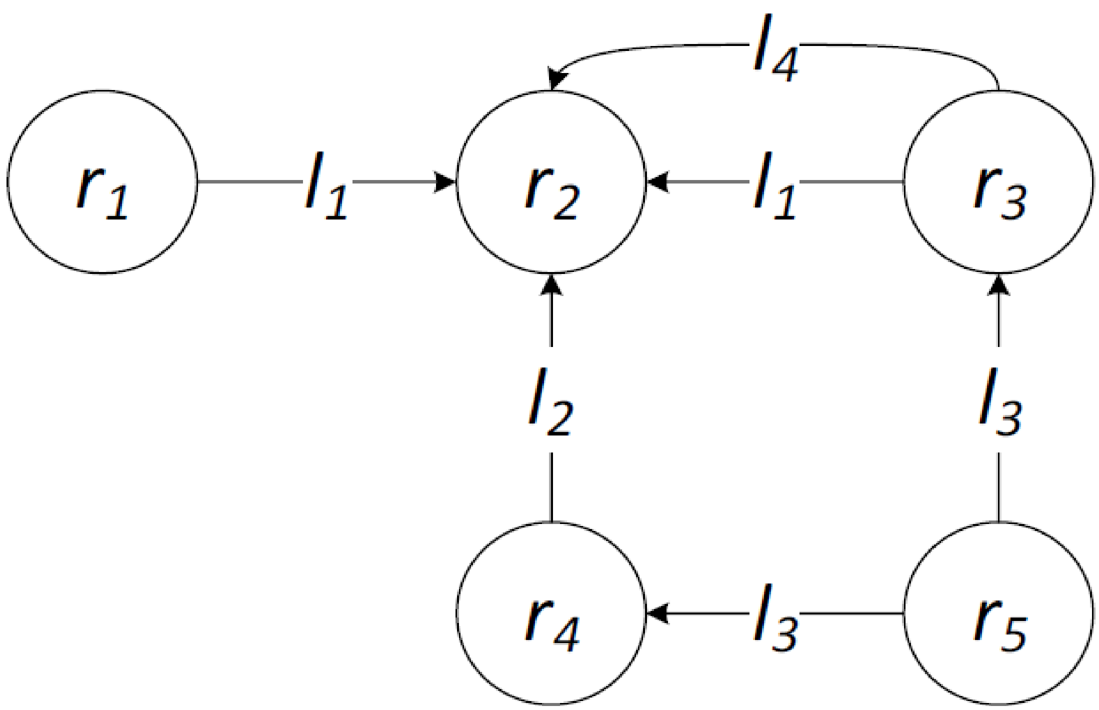


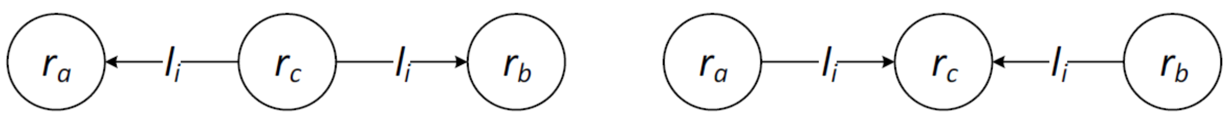
Figure 3.2: Sample graph

II. Linked Data Semantic Distance (Indirect)

Resources can also be indirectly connected to other resources through the indirect distance between two resources which happens when they are linked via another resource, and these connections are either both incoming or both outgoing through the intermediate resource.

Definition 2: *"$C_{inoutcom}$ and $C_{inincom}$ are functions that compute the number of indirect and distinct links, both outcoming and incoming, between resources in a graph G. $C_{inoutcom}(l_i,r_a,r_b)$ equals 1 if* *there is a resource n that satisfy both <l_i,r_a,n> and < l_i,r_a,n >, 0 if not. C_{ii} (li, ra, rb) equals 1 if there is a resource n that satisfy both <li, n, ra> and <li, n, rb>, 0 if not. By extension $C_{inoutcom}$ and $C_{inincom}$ can be used to compute (1) the total number of indirect and distinct links between ra and rb ($C_{inooutcom}$(n,r_a,r_b) and $C_{iniincom}$(n,r_a,r_b), respectively outcoming and incoming) as well as (2) the total number of resources n linked indirectly to ra via li ($C_{inoutcom}$ (li,ra,n) and $C_{inincom}$ (li,ra,n), respectively outcoming and incoming)"* (Passant, 2010).

Therefore, there are two types of indirect connections: incoming and outcoming. An incoming indirect connection between two resources *ra* and *rb* exists if there is a resource *rc* such that *rc* is directly connected to both *ra* and *rb* as in part A of figure 3.2 Likewise, an outcoming indirect connection between two resources *ra* and *rb* exists if there is a resource *rc* such that both *ra* and *rb* are directly connected to *rc* as in part B of figure 3.2



A: Indirect Incoming Links B: Indirect Outcoming Links

Figure 3.3: Linked Data Semantic Indirect Distance

However, by combining the $C_{inoutcom}$ and $C_{inincom}$, the $LDSDindirect(ra, rb)$ will be calculated according to the equation 3.2:

$$LDSDindirect(ra, rb) = \frac{1}{1+Cinoutcom(n,ra,rb)+Cinincom(n,rb,ra)} \qquad (3.2)$$

III. Linked Data Semantic Distance (LDSD)

The Linked Data Semantic Distance (LDSD) is a result of adding the $LDSDdirect(ra, rb)$ and the $LDSDindirect(ra, rb)$ as shown in the equation 3.3:

$$LDSD(ra, rb) = \frac{1}{1+LDSDdirect(ra,rb)+LDSDindirect(ra,rb)} \qquad (3.3)$$

Let's take an example to explain how the LDSD is calculated. Suppose we have four artist names and these are called set of resources "R" (Adele, J.Lo, Arianna Grande, and Shakira) and we have different types of links "L" (Band and MusicalArtist) connect these artists as shown in the figure 3.2, these links are retrieved from DBPedia.

Resources	Cdirect		Cindirect			
			Incoming		Outcoming	
	Band	MusicalArtist	Band	MusicalArtist	Band	MusicalArtist
Adele-J.Lo	1	1	1	1	1	1
Adele-Arianna Grande	2	0	4	2	2	2
Adele-Shakira	1	1	1	0	1	0
J.Lo-Adele	1	1	1	1	0	1
J.Lo-Ariana Grande	2	2	2	0	3	2
J.Lo-Shakira	1	2	3	2	3	2
Ariana Grande-Adele	2	0	4	2	2	2
Ariana Grande-J.Lo	2	2	2	0	3	2
Ariana Grande-Shakira	1	3	2	3	2	3
Shakira-Adele	1	3	1	0	1	0
Shakira-J.Lo	1	2	1	1	1	1
Shakira-Ariana Grande	2	3	2	3	2	3

Table 3.1: Example for direct and indirect links between the artists

$$LDSDdirect(Adele, J.Lo) = \frac{1}{1+Cdirect(Band,Adele,J.Lo)+Cdirect(MusicalArtist,Adele,J.Lo)+Cdirect(Band,J.Lo,Adele)+Cdirect(MusicalArtist,J.Lo,Adele)}$$

$$LDSDdirect(Adele, J.Lo) = \frac{1}{1+1+1+1+1} = 0.2$$

$$LDSDindirect(Adele, J.Lo) = \frac{1}{1+Cinoutcom(Band,Adele,J.Lo)+Cinoutcom(MusicalArtist,Adele,J.Lo)+Cinincom(Band,Adele,J.Lo)+Cinincom(MusicalArtist,Adele,J.Lo)}$$

$$LDSDindirect(Adele, J.Lo) = \frac{1}{1+1+1+1+1} = 0.2$$

$$LDSD(Adele, J.Lo) = \frac{1}{1+LDSDdirect(Adele,J.Lo)+LDSDindirect(Adele,J.Lo)}$$

$$LDSD(Adele, J.Lo) = \frac{1}{1+0.2+0.2} = 0.714$$

Table 3.2 shows the direct, indirect and the total linked data semantic distance between the two artists which indicate that (LDSD) can be applied to measure relatedness between resources from the Web of Data and they can be applied to provide resources recommendations.

Resources	LDSDdirect	LDSDindirect	LDSD
Adele-J.Lo	0.2	0.2	0.714
Adele-Arianna Grande	0.2	0.091	0.775
Adele-Shakira	0.143	0.333	0.678
J.Lo-Adele	0.2	0.25	0.670
J.Lo-Ariana Grande	0.111	0.125	0.809
J.Lo-Shakira	0.143	0.091	0.810
Ariana Grande-Adele	0.2	0.091	0.775
Ariana Grande-J.Lo	0.111	0.125	0.809
Ariana Grande-Shakira	0.111	0.091	0.832
Shakira-Adele	0.2	0.333	0.652
Shakira-J.Lo	0.143	0.2	0.745
Shakira-Ariana Grande	0.111	0.091	0.832

Table 3.2: Semantic distances between the artists

3.2.1.4. User Profile Manager

User profiles play an important role in recommendation processes since their models represent the user's information needs. Most personalisation recommender systems need to build a user profile or a model of user preferences in order to identify the needs of individual users. The initial step in providing personalised recommendations is to learn about user interests and preferences in order to generate a user profile. Users' preferences can be gleaned from their past interactions with the systems in question. These user interactions consist of either explicit or implicit information about the user's preferences or interest in items. The user profile allows users to be modelled, which can be described as the process of building personal preferences. In other words, the user model is generally represented in the form of a user profile which captures the personal preferences of the users in terms of the user's knowledge about the object or subject in which they are interested.

In another way, the user profile is a structured construct containing data both directly and indirectly pertaining to a user's preferences, behaviour, and context. Effective personalisation needs services to make and maintain correct models of a customer's preferences, interests, and background through a user profile.

However, with the increasing number of music services such as Google Play Music, iTunes, Spotify, etc. there is a demand to engage the customers to their sites. Recommending songs to the user is one way of meeting the demand. The user profiles are first formed by extracting features of the data items, which have been accessed in the past. Based on the user profiles, the system recommends only the data items that are highly relevant to the user profiles by computing the similarities between the features of the data items and the user profiles.

The task of the User Profile Manager is to dynamically extract and collect the user information that will be used later on in the recommendation engine. The recommender system collects both kinds of information to generate the user profile. This profile stores information not only about the user likes, also information about the user itself, current personal needs, age, and so on. The information stored within is also a determinant factor in the recommender algorithm design. The User Profile Manager contains three modules: Tweets Collector, Facebook Collector and User Listening History Collector. This thesis aims to build a user profile using the information obtained from Twitter and Facebook and taking into account the user listening history to provide a personalised music recommendation service. With Spotify, the user has to listen to at least one song for 30 seconds to consider this song in his/her listening history in order to build user profile so that Spotify can recommend new songs to the user. In CAPM framework, the user does not need to listen to any song because the user tweets and Facebook status will be used by extracting the important keywords that relate to the song lyrics to build the personal profile and then CAPM can recommend songs to the user even though the user has no listening history (no new user Cold-Start problem).

3.2.1.4a. Tweets Collector

Twitter is a social information network where short messages or tweets are shared among a large number of users through a very simple messaging mechanism.

One of the problems of recommender systems is the Cold-Start problem, where the system does not have enough information about an item or a user to make accurate recommendations. This problem will be mitigated in this thesis by gathering information from Twitter. Twitter is an extensive source of information about real-time events, but because of considerable noise, it can be difficult to find the information relevant for a recommender system. The main concern in the music recommendation systems is that the recommendation should happen based on user lifestyle and manner rather than friends' preferences. For this purpose, we need history or some data of user. To overcome this Cold-Start, the system will connect to a user's twitter account and retrieve posts. After collecting the user's post, keywords will be extracted and unwanted ones will be removed. Then after doing some analysis, the system will compare these extracted keywords with song lyrics to calculate which song are similar to users' posts (Wang, 2014).

The ability to collect large amounts of digital traces of human behaviour through Twitter and other social media platforms represents a new pattern for social science research in the area of data collection (Mccormick, 2017). In this thesis, a Twitter Collector is presented which is capable of gathering and organizing information from Twitter in a way that makes it usable for a music recommender system.

The Twitter Collector (TC) connects to the users' twitter account through the Twitter API, which allows easy access to users' tweets by means of their username. After gaining access to users' tweets, the TC then checks the tweets for musical information that can be used for recommendations. Information such as artists, songs, album, musical groups and genres are collected by the TC and passed to the Recommendation Engine through the Server Manager. The TC was included in this framework due to the volume of users of Twitter and the potential musical information that can be gained from tweets of users. A single tweet from a user can reveal the preferred genre or artists of the user. For example, if a user tweet contains a song or songs from *The Beatl*es, known for their songs in the rock genre, the Recommendation engine is able to identify the song as belonging to *The Beatles* and recommend other songs performed by *The Beatles* to the user. It can also identify the song as belonging to the rock genre and recommend other songs in the same genre by different artists. The information gained from a single tweet by a user can be used in different ways by the recommendation engine to provide personalised recommendations to the user. A song can be recommended to the user based solely on his or her past tweets. However, in the case of absence of the user's more recent tweets, the system will resort to the user's tweets from recent months. In the case that the user lacks a Twitter account, the system can revert to Facebook (as discussed in the next module).

3.IV. Tweets Collector

1. If the current user has a Twitter handle account ID, then
2. Look for the user's tweets in the past days and include them in the index
3. If no tweets can be found linked to the user, then
4. Look for tweets of the user T that span within the past week and index them
5. If there are no tweets for the past week, then
6. Look for tweets spanning to the past months and index
7. End if
8. End if
9. End if

3.2.1.4b. Facebook Collector

Social media sites provide a rich source of user-generated content revealing the latent characteristics of their users. Effective and efficient mining of this data can facilitate the detection of emotions and personality of users without the need for open, user-informed data collection. Consequently, this improves the application potential of bespoke or customised services that depend on such knowledge for personalisation.

Facebook is a social networking platform launched in 2004. Its users have profiles that display personal information, including their interests, and photos. Users can 'friend' or 'like' other users of the platform, and can engage in various activities, for instance writing on the 'walls' of other users, sharing and commenting on links, 'liking' brands or public figures, and participating in discussions in 'groups'. Through these activities, Facebook users can build and maintain social capital, keep up with the lives of their social circle, communicate with other users, and consume news and entertainment. Research on Facebook has tended to focus on the functionality of the site, and its norms (Papacharissi, 2009), the way in which it is used, and for what purpose (Debatin et al., 2009; Ellison, Steinfield, and Lampe, 2007), and self-presentation and identity management via the platform (Labrecque, Markos, and Milne, 2011; Papacharissi, 2009; Tom Tong et al., 2008; Smith et al., 2012; Zywica & Danowski, 2008).

A typical Facebook profile contains certain information, such as a user's bio, photos, friends, likes, interest, hobbies, groups and so on. The task of the Facebook Collector (FC) module is to gather the user's Facebook status information. The FC searches a user's profile to get information such as the user's musical taste, favoured artists, musical groups and other musical preferences. The statuses that contain relevant musical information are then collected and indexed. The Facebook computer program interface offers a straightforward means of collecting the statuses of an individual if their username is provided in a supported time context; therefore, suggesting a song to the user is strictly limited to his or her past statuses. Also, the FC can be used to determine the mood of a user at a given point in time. The *Feeling/Activity* feature on Facebook can indicate a user's current mood such as happy, sad, excited, etc. This mood information can also be collected by the FC and used to provide personalised song recommendations to the user. This kind of context-aware framework, though rarely used in the past, can be used effectively to provide personalised recommendation for users of the framework.

3.V. Facebook Collector

1. If the current user has a Facebook account ID, then
2. Look for the user's statuses in past days and include them in the index
3. If no statuses can be found linked to the user, then
4. Look for statuses of the user S that span within the past week and index them
5. If there are no statuses for the past week, then
6. Look for statuses spanning the past month and index
7. End if
8. End if
9. End if

3.2.1.4c. User Listening History Collector

Music listening histories are considered to be an interesting subject for research, and provide a timeline of a user's listening events. Many researchers tend to analyse the user listening histories because they provide a clear idea about the user's music consumption, and what kind of music he/she is enjoying or not enjoying over time. The User Listening History Collector in the CAPM framework is intended to track and store users' listening history (what songs the user listens to) and save this information for use in the recommender system when selecting the user's 'neighbours' based on listening history. This module builds on users' listening history to provide users with new pieces of music that meet their taste and give them novel music that leads to make the recommender system effective and attractive to new users. By establishing users' listening history, the CAPM is able to extract helpful information with which to understand and predict users' preferences.

CAPM learns from the users' past listening history and recommends them songs which they would probably like to hear in future. Users can benefit from the recommender system without going through the pain of making decisions on what to listen. Through studying users' listening history, we will be able to extract useful information and predict users' preference.

Information that can be collected through a user's listening history include the user's favourite songs, albums, genres and artists. For instance, a user who has listened to mostly R&B songs in the past is most likely a fan of R&B music. Therefore, songs that will be recommended to such a user will most likely be ones from the R&B genre.

3.VI. User Listening History Collector

1. **While** there are users in the system **do**
2. Get user info and index it
3. **if** User U plays a Song S **then**
4. Get Song ID and Song rating
5. Number of times a song has been played and index it
6. **End if**
7. **End while**

3.2.2. Data Processing

Data processing typically starts with pre-processing, normalizing, and filtering the raw data, then transforming the data to a format that can be input to the algorithms being used or calculations being performed, and lastly computing the metrics to facilitate the decision making.

After collecting the data from multiple sources, the collected data must be processed. This makes data organised and understandable, and makes it easy to retrieve specific information at any given time. The data processing stage relies on four components: song analyser which is important for compiling playlists, mood classifier which helps to recommend songs based on the mood of the listener, streamer for playback function, and the indexer which helps to facilitate quick querying. These will be discussed below.

3.2.2.1. Song Analyser

Content-Based music recommender systems analyse the music features or metadata such as information on genres, artists, and lyrics to find similar items (Bogdanov et al., 2012). Unlike the Collaborative Filtering methods, this approach does not face the Cold-Start problem or popularity bias.

Therefore, the previous research has shown that song characteristics (lyrics, artist, album, etc.) play an important role in playlist composition (Lee, 2011), and the analysis of these song characteristics can provide helpful information about the song itself that can be used in recommendation systems. By using Content-Based filtering, the attributes of the music will be analysed, such as the genre, tone, theme of the lyrics, artists, album etc. (Dieleman, 2014). The analysed data of the music will be collected as track-related information, which then serves as the basis for making recommendations. The Song Analyser module in the CAPM framework was designed for this purpose, and the similarity between all of the songs stored in the CAPM database will be calculated when making a recommendation. However, all songs, as mentioned above, have unique sets of characteristics, including lyrics, artist, album, title, and year of release. By giving each characteristic a different weighting and adding these together, the song similarity can be calculated. The weightings are set empirically to get the best results that meet user's need. The combinations of weights were tested by 53 users, who are students at University of Portsmouth. The final weights used are 55% lyrics similarity, 20% LDSD between artists, 20% title similarity, 5% same year of production. Song analyser is important as songs with similar characteristics to the user's listening history or the user's song preferences can be recommended to the user. This can only be possible when the song analyser has found sufficient links between songs based on their characteristics.

In order to determine the lyrics similarity, the cosine similarity has been used (equation 3.4). Given two lyrics vector (A, B) and the similarity between them is determined using equation (3.4) and then the result is multiplied by the weighted value of 55% which mentioned in the above.

Subsequently, the linked data semantic distance (LDSD) determined in the LDSD calculator module will be retrieved and then multiplied by 20% as a weighting. Then, the title similarity will be calculated again using cosine similarity, and the value multiplied by a 20% weighting. At the next stage of calculation, the last part of the weighting which is (5%) is given to the year of song production; if the songs have the same year of production then the weight will be 5% and if not, the weight given is 0%. The above procedure is demonstrated in the algorithm shown below. The last stage of the process is to sort tracks in descending order.

$$\text{Cosine similarity (A, B)} = \frac{A.B}{|A||B|} \quad (3.4)$$

$$\text{Score} = \text{LS} * 0.55 + \text{LDSD} * 0.2 + \text{TS} * 0.2 + \text{ if(production years are similar, } 0.05, 0) \quad (3.5)$$

Where:

LS: Lyrics Similarity, LSDS: Linked Data Semantic Distance and TS: Title Similarity.

3.VII. Song Analyser Algorithm

1. **While** there exists a Song S in the songlist, do
2. Compose a vector for S lyrics
3. **While** there exists a song to be compared Sc among the song list, do
4. Proceed to compose a vector for Sc
5. Obtain Lyric Similarity (equation 3.4) between S and Sc vectors then multiply the result by 55%
6. Obtain the value of Linked Data Semantic Distance (LDSD) algorithm then multiply the result by 20%
7. Obtain Title Similarity (equation 3.4) between Song S title and Sc title vectors then multiply the result by 20%
8. **If** Sc and S year of production match, then
9. Increase the score by multiplying by 5%
10. **End if**
11. Include similar song with the same score in similar song list
12. **End while**
13. Arrange similar song list in a descending order depending on the score
14. Add similar song list to S in the index
15. **End while**

In order to understand how to measure the lyric similarity and the title similarity we will take an example as shown below:

Suppose we have song A with lyrics [Let the rhythm change your world], and song B with lyrics [the rhythm so high make me feel the world], therefore we will use the following vector representation

Distinct words from both songs	Word frequency in song A	Word frequency in song B
Let	1	0
The	1	2
Rhythm	1	1
Change	1	0
Your	1	0
world	1	1
So	0	1
High	0	1
Make	0	1
Me	0	1
feel	0	1

Table 3.3: Example for measuring the cosine similarly between two songs

Vector A (song A) = [1 1 1 1 1 1 0 0 0 0 0]

Vector B (song B) = [0 2 1 0 0 1 1 1 1 1 1]

$$\text{Cosine similarity (A, B)} = \frac{A.B}{|A||B|}$$

$$\text{Cosine similarity (song A, song B)} =$$

$$\frac{1\times0+1\times2+1\times1+1\times0+1\times0+1\times1+0\times1+0\times1+0\times1+0\times1+0\times1}{\sqrt{1^2+1^2+1^2+1^2+1^2+1^2+0^2+0^2+0^2+0^2+0^2}\times\sqrt{0^2+2^2+1^2+0^2+0^2+1^2+1^2+1^2+1^2+1^2+1^2}} = \frac{4}{\sqrt{6}\times\sqrt{9}} = 0.544$$

3.2.2.2. Mood Classifier

Researchers have found that music can be effective in improving mood, raising energy and reducing tension (Thayer, Newman, & McClain, 1994). For example, athletes typically prefer up-tempo, conventional, intense, rebellious, energetic, and pulsing music over reflective and complex music. Furthermore, psychologists have found that users' music preferences are related to their temperament. Such studies highlight that music recommenders are not solely a tool for relaxation, but also act as good tools to satisfy desires within completely different contexts. Designing an individual music recommender is difficult, as it is challenging to fully and accurately capture users' desires and meet their needs. Selecting the most appropriate music under different mood conditions is an important contribution nowadays for people who want to listen to music.

This component recommends music to users based on the current user's mood as apart from the other recommendation factors in order to address the Cold-Start problem in the music recommendation industry. Recommender system takes the current user's mood and then CAPM gives the user appropriate music fits with that current mood.

The mood classifier in the framework helps to improve recommendation to users as songs will be grouped into different mood categories so that suitable songs can be recommended to users based on their current mood.

CAPM has 18 mood categories. This is based on (Hu, 2009) division of mood into 18 mood tag group categories utilising a ground truth dataset of 5,585 songs. The research used 186 mood related words as tags in the classification of songs. For instance, the 'Calm' mood category has tags such as calm, comfort, quiet, serene, etc., while the 'Sad' category has tags such as sad, sadness, unhappy, melancholic and so on as shown in table 3.4.

Mood number	Mood-Based Words
Mood 1 (Calm)	calm, comfort, quiet, serene, mellow, chill out, calm down, calming, chill-out, comforting, content, cool down, mellow music, mellow rock, peace of mind, quietness, relaxation, serenity, solace, soothe, soothing, still, tranquil, tranquility
Mood 2 (Sad)	sad, sadness, unhappy, melancholic, melancholy, feeling sad,
Mood 3	happy, happiness, happy songs, happy music, glad

(Happy)	
Mood 4 (Romantic)	romantic, romantic music
Mood 5 (Gleeful)	upbeat, gleeful, high spirits, zest, enthusiastic, buoyancy, elation,
Mood 6 (Depressed)	depressed, blue, dark, depressive, dreary, gloom, darkness, depress, depression, depressing, gloomy
Mood 7 (Angry)	anger, angry, choleric, fury, outraged, rage, angry music
Mood 8 (Grief)	grief, heartbreak, mournful, sorrow, sorry, doleful, heartache, heartbreaking, heartsick, lachrymose, mourning, plaintive, regret, sorrowful
Mood 9 (Dreamy)	Dreamy
Mood 10 (Cheerful)	cheerful, cheer up, festive, jolly, jovial, merry, cheer, cheering, cheery, get happy, rejoice, sunny
Mood 11 (Brooding)	brooding, contemplative, meditative, reflective, broody, pensive, pondering, wistful
Mood 12 (Aggressive)	aggression, aggressive
Mood 13 (Confident)	confident, encouraging, encouragement, optimism, optimistic
Mood 14 (Anxious)	angst, anxiety, anxious, jumpy, nervous, angsty
Mood 15 (Earnest)	earnest, heartfelt
Mood 16 (Hopeful)	desire, hope, hopeful
Mood 17 (Pessimism)	pessimism, cynical, pessimistic, weltschmerz, cynical, sarcastic
Mood 18	excitement, exciting, exhilarating, thrill, ardor, stimulating, thrilling, titillating

(Excitement)	

Table 3.4: Mood Categories (Hu et al., 2009)

The songs in CAPM library were classified according to the mood classifier algorithm, where a song's tags alongside its frequency are collected from Last.fm using the method track.getTopTags. Then, the tags for each song are compared with the mood-based words that belong to each mood category (see table 3.1). Finally, the song's mood factor will be determined according to equation 3.6 below:

$$\text{Mood Factor} = \sum(MT \times TFreq) \tag{3.6}$$

Where:

MT is the matched tags belonging to a specific mood category (1 if matched, 0 otherwise)

TFreq is the tag frequency.

3.VIII. Mood Classifier

1. **While** there are songs S in songs database, **do**
2. Get the song tags with tag frequency from Last.fm
3. Compare the collected tags for each song with mood standard table
4. Save the matched tag for each mood category
5. Calculate the mood factor for each song (equation 3.6)
6. **End while**

It is helpful to use an example to explain the algorithm: the tags for the song "Don't You Remember" for the artist "Adele" were retrieved from Last.fm with the tag frequency, as shown in the table 3.5:

Tag	Frequency	Tag	Frequency	Tag	Frequency
hi	1	downtempo	1	soft	1
emotive	1	music	1	amour	1
awful	1	lost	1	ad	1
diaries	1	English	1	epic	1
fucking	1	heartache	3	it	2
linedance	1	Adele	1	easy	1
beautiful	3	10	3	-	1

rb	1	21	1	James	1
at	1	Duffy	1	pop	2
10s	1	love	7	remember	1
5	2	heartbreak	1	rock	3
vocals	1	artist	1	top	1
8	1	6	1	lady	1
that	1	UK	1	me	1
hello	1	chillout	1	vocalist	1
favorite	5	r&b	1	2010s	1
memories	1	cry	1	bittersweet	1
a	1	stars	4	eargasm	1
vocalists	1	blunt	1	vampire	1
time	1	breaks	1	of	4
listening	1	drama	1	soul	4
make: 1	1	import	1	poprock	1
sad	2	n-a	1	tragically	1
and	2	faves	1	hurts	1
blues	2	mellow	1	my	3
best	1	favorites	1	ballad	1
2011	1	indie	1	all	1
vocal	1	foxy	1	female	1
no	1	neo-soul	2	up	1
don’t	1	heart	3	awesome	1
break	1	the	4	00s	1
from	1	straight	1	blue	1
emotional	1	overrated	1	heartbreaking	2
one	1	To	1	singer-songwriter	1
l	1	first	1	heartbroken	1
t	1	masterpiece	1	melancholic	1

songs	5	own	1	feel	1
jazz	1	British	2	good	1
listen	1	melancholy	1	meaningful	1
neo	1				

Table 3.5: The collected tags "Don't You Remember – Adele" from Last.fm

All the collected tags in the above table were compared with the standard table (table 3.1) and the matched tags were used in equation 3.6 to determine the mood factor, as shown below:

- Mood: 1, number of matched tags: 2 [tag: mellow, frequency: 1; tag: chillout, frequency: 1],

  Mood Factor = 1 * 1 + 1 * 1 = 2
- Mood: 2, number of matched tags: 3 [tag: sad, frequency: 2; tag: melancholic, frequency: 1; tag: melancholy, frequency: 1]

  Mood Factor = 1 * 2 + 1 * 1 + 1 * 1 = 4
- Mood: 6, number of matched tags: 1 [tag: blue, frequency: 1]

  Mood Factor = 1 * 1 = 1
- Mood: 8, number of matched tags: 3 [tag: heartbreak, frequency: 1; tag: heartache, frequency: 3; tag: heartbreaking, frequency: 2]

  Mood Factor = 1 * 1 + 1 * 3 + 1 * 2 = 6

Therefore, the song "Don't You Remember" for the artist "Adele" will be classified as follows: Mood: 1 (Calm), Mood: 2 (Sad), Mood 6: (Depressed) and Mood 8: (Grief) and so on, with the other songs in the CAPM database. The song will appear in the four categories listed above. Most songs in the database of the framework will belong to more than one mood category depending on their tags. A song might carry a tag that represents mood 3, the happy mood, and carry a tag that represents mood 18, which is the excitement mood. These two categories are closely related and most songs in the happy mood category will most likely belong to the excitement mood category.

3.2.2.3. Indexer

All the collected data are indexed to enable quick querying facilitating first result release to the user in as short a time span as possible. The index will contain the fields given below:

- Song and Artist ID

- Song Title
- Artist name
- Album name
- Date of production
- Lyrics
- Song similarities
- Mood classifications
- User listening history

The song and artist ID and mood classifications will lead to the quick identification of the song, eliminating the need to search through the titles and artists and map the matches. Instead the CAPM will be able to revert to the n^{th} artist as well as the n^{th} song, resulting in faster and more efficient functioning of the system.

For the song title, artist, and album names the Term Frequency (TF) and Inverse Document Frequency (IDF), are obtained using equations 3.7 and 3.8 (shown below), respectively. Through these calculations, the TF-IDF can be obtained for every term in the fields and retained as a post in the Inverted File Index.

$$TF = \frac{Number\ of\ counts\ of\ term\ appearance\ in\ the\ document}{number\ of\ terms\ found\ in\ the\ document} \tag{3.7}$$

$$IDF = \log \frac{sum\ of\ the\ documents}{documents\ with\ term\ t} \tag{3.8}$$

The year of production can be obtained in various formats. Hence there is a need for the formats to be uniform in order to speed up the process of generating a recommendation.

The song similarities and user listening history will be sorted in descending order for quick discovery of the most similar track.

3.2.2.4. Streamer

In order to download the songs from the server side to the client side, the Streamer module is used to facilitate the downloading of the song for playback. When the user makes a request to the server, the track is downloaded to his or her device along with the preceding and subsequent track. This feature enables the user to either fast-forward or rewind the song without having to let the song play out against their will. Before terminating the song being played, a request for the next song would be made, making it available as the next play in order to enable continuous playback.

The song is downloaded with the relevant information, including the artist, date of release, title, and album - if all of these details are available. The streamer is essential to the framework as it ensures that songs are retrieved from the server and made available to the user for continuous playback. Without the streamer, the user will be unable to enjoy continuous playback. This means that the user will get the songs one by one. This can be strenuous and time consuming and may results in the user becoming disillusioned with the CAPM.

3.2.3. Recommendation component

A recommender engine is a tool and predicts user preferences for songs that users have not expressed any preference for. Therefore, the recommendation section contains a number of components to handle with real-life data sets, which can construct a customised recommender system and work together in order to implement the recommendation algorithms. These components are: recommender manager, find requester and the server manager.

3.2.3.1. Recommendation Engine

In order to provide personalised recommendations to the users, a series of stages will be implemented. A different score will be calculated at each stage depending on the weighting of that stage. The recommendation engine will be discussed in detail in Chapter 4.

3.2.3.2. Find Requester

The CAPM recommends songs to the user according to their listening history as well as their tweets, Facebook statuses, and mood. In the case that the user has no relevant data (no listening history, no tweets, no Facebook status, and no mood selected), the user must perform a search. This is another important aspect of music exploration, where recommendations are presented to the user so that they can play a song; the recommendation algorithms will then be implemented according to the played song. The Find Requester module is used for this purpose, by searching the title and lyrics of a song in order to return relevant results. The Find Requester is like a fail-safe feature in the framework. The framework is designed to work even when the user has no tweets, Facebook status or current mood to make recommendations. This is where the Find Requester module comes into play.

The Find Requester module is a fast means of searching through the index file and returning the most relevant outcome. The recommendation algorithms use the Finder Requester in the process of looking for lyrics and titles correlating to the songs stored in CAPM database and returning the results that are calculated to be the most relevant. Their relevancy is calculated by multiplying term frequency with the inverse document frequency as derived from the inverted index while obtaining the TF-IDF weight w. The Vector Space Model becomes the basis for ranking the obtained results.

$$TF - IDF\ weight = \frac{\sum_i^n (TF_{i,j} \times TF_{i,d})}{\sqrt{\sum_i^n TF_{i,j}}^2 \times \sqrt{\sum_i^n TF_{i,d}}^2} \quad (3.9)$$

Where:

n: is the no. of the results, : the term frequency weight for word i in the document j, : the term frequency weight for word i in the document d.

For initial setup, the user has to make a request. To begin with, the Find Requester would provide a list of songs that can be selected and played. It does not directly provide tracks, though the tracks are logically anchored once obtained from the database. This implies that once the list has been collected, tracks can be anchored for easy searching. The Find Requester uses these anchored details to return values that correspond to the user's search.

3.2.3.3. Server Manager

In order to manage all the requests sent by the CAPM and to keep the connection continually updated, a Server Manager is needed. The internet connection will be used to send all requests between the client and the server.

The main task of the Server Manager is to make available server services to counter requests made by CAPM. The returned results are forwarded via a TCP socket that must be assigned to the right section of the server. CAPM will have multiple users who will be logging in, and who may request recommendations or may just search for a track. The Server Manager will collect the user's recommendations and present them to the user.

3.3. Client Side

The client-side of the system will be an application with a user interface that is integrated into a music listening website or application. This application gathers the information from users, investigates some actions of the users, and provides the connection with the server. This application is the client-side interface of the CAPM framework. The client side contains several components: data search handler and recommendation handler, and each of them contain a number of components as shown below:

3.3.1. Data Search Handler

Modern recommendation systems have a search function which allows users to find all information such as web pages, products, or news. Therefore, The Data Search Handler section is responsible for getting correct information about any recommended song. When a song is recommended, the data search handler section will check the song for any incorrect or incomplete information and correct it. It also has the responsibility of identifying and presenting the correct video for the recommended song. It is also responsible for providing the streaming and playback function to the user. The components in the section are the YouTube Searcher, Last.fm Searcher, Streamer and Playback.

3.3.1.1. YouTube Searcher

YouTube Searcher is a module that aims to help identify the correct video for the music track currently being played. With this, the user will be able to watch the video as well as listen to the track. With the help of a YouTube API, this searcher queries the track name as well as the artist, guaranteeing an almost perfect video return. The contribution of this component to the framework is that while users are getting recommendation to songs, they can also get to watch the video to the songs. It is a complimentary component that is meant to enhance the users' experience while using the framework.

3.3.1.2. Last.fm Searcher

This is used to find the album art and show it to the users while playing the song. The artwork is then stored on the device to use again without the need to download the image. Album art constitutes the album cover, which is displayed when a song is played on the device. However, there are a large number of songs that do not have any album art, which makes the screen appear dull and unattractive to the user. This module therefore connects to Last.fm through the Last.fm API in order to identify the details of songs and uses this to search for an appropriate image or album art for the song to be displayed on the user screen. The feature helps to create a more interactive interface for the users.

This module, like the YouTube Searcher module, is a component that is meant to improve the users' experience of the framework. With this, users can get a vibrant and attractive interface, making the framework interesting to use.

3.3.1.3. Streamer

As previously explained, the streamer module enables a user to download a song to his or her device for continuous playback.

3.3.1.4. Playback

This component has the task of displaying the details provided by the song's .mp3 tags and the album artwork as provided by Last.fm, and enabling the playback of the track. Without the playback module, users can't listen to recommended songs. With the playback function, users can play a song and perform a couple of other options such as pause and stop. In addition, the playback displays important information about the song such as the title, the artist, album, year and lyrics.

3.IX. Playback Algorithm

1. If play is started, then
2. If song is playing, then
3. Stop the song currently being played
4. End if
5. Show Title, Artist, and Album from the Song S .mp3 Tags
6. Show the Album Art that has been downloaded via Last.fm Crawler for Song S
7. Play Song S
8. Else if paused, then
9. Stop the track being played
10. Else if forward is pressed, then
11. Increment S and go back to line 1

12. Else if rewind is initiated
13. Decrement S and go back to line 1
14. End if

3.3.2. Recommendation Handler

The Recommendation Handler component is responsible for obtaining search terms and requesting recommendations on behalf of the user. It consists of the Recommendation Requester, Song Requester and Client Manager. Requests for songs and recommendations are sent to the Recommendation Section through the Client Manager. The Client Manager then receives the response and sends it to the Data Search Handler.

3.3.2.1. Recommendation Requester

This request asks for a recommendation from the server handler; in response, a list is reproduced and then downloaded and stored in the client's repository.

3.3.2.2. Song Requester

This component is responsible for obtaining the search terms imputed by users and transferring it to the client manager module for further processing.

3.3.2.3. Client Manager

This module is responsible for communicating with the server and dealing with the communications protocol and interpretations. The module is designed to understand and to process all communications with the server. It offers interpretations for server return or replies, and prepares client requests to be in a perfect state that the server can understand and deduce exactly what the request concerns. Issues that this module deals with include getting the addresses for the server, and parsing the server's return in order to determine the implications and content of the message. The message may be encrypted; hence the Client Manager must have the cipher key in order to decrypt the message for the client to interpret.

After the search request has been made, the client server is kept alert in order to listen for any communication from the server, and keep the user informed of any transactions taking place.

3.4. Summary

This chapter presented the details of the proposed generic Context-Aware Personalised Music (CAPM) framework, which provides song recommendations to users based on the contextual information associated with their profile. CAPM consists of different components that work effectively to support context distributions and personalised music recommendation and to address the Cold-Start problem, which pertains to the difficulty of providing high quality recommendations to new users and new songs where little information exists. CAPM supports the representation, indexing, sharing and delivery of context information and provides modular components that are common across applications.

Chapter 4

A hybrid multi-strategy recommendation approach

In CAMP framework, recommendation engine is a key component. It employs a hybrid multi-strategy approach to better capture users' preferences and augment the accuracy of the recommendations. In this chapter, multiple recommendation algorithms are explained and discussed in details.

4.1. General approach

The lack of necessary information causes a 'Cold-Start' problem in music recommender systems, preventing them from making a good recommendation. While CF is the most commonly used music recommendation system, it suffers not only from the problem of recommending songs to new users where the system has no existing knowledge of their music taste (the 'new user' problem), but also from the problem of recommending new songs, about which the system does not have enough information, i.e. ratings (the 'new song' problem).

Unlike CF, CB recommends songs with similar content to the songs that the user has preferred in the past (Chen, 2001). The recommendation quality of Content-Based systems is based on the fact that CB largely uses acoustic features, such as rhythm, timbre, and other musical features to determine the similarity between two songs (Song et al., 2012); using this method, the system can recommend new songs even if they have no ratings. Consequently, CB systems do not suffer from the new song problem, although the Cold-Start problem (new user) is still present. Thus, a hybrid method, which is a combination of the CF and CB approaches, can be used to enhance the recommendations and to address the Cold-Start problem (Yoshii et al., 2006).

In order to provide personalised recommendations to the users, a multi-strategy approach is implemented in the recommendation engine, and scores are calculated at each stage depending on the weighting of that stage. Best weightings were obtained through experiments: Collaborative Filtering (CF) (K-nearest Neighbour) – 20%, Content-Based (CB) – 50%, Context-Based – 30% which contains (Twitter & Facebook – 15% and Mood – 15%).

4.2. User-Based Collaborative Filtering Algorithm

The great quantity of music content available online has increased interest in music recommender systems. However, some important problems must be addressed in order to give reliable recommendations. Many approaches have been proposed to deal with Cold-Start and first-rater drawbacks; however, the problem of generating recommendations for gray-sheep users has been less studied. Most of the methods that address this problem are Content-Based, hence they require item information that is not always available. Another significant drawback is the difficulty in obtaining explicit feedback from users, necessary for inducing recommendation models, which causes the well-known sparsity problem. In this work, a recommendation method based on playing coefficients is proposed for addressing the above-mentioned shortcomings of recommender systems when little information is available. The results prove that this proposal outperforms other Collaborative Filtering methods, including those that make use of user attributes.

Therefore, the user-based Collaborative Filtering is used to measure the similarities among users using the cosine similarity and select the top five user neighbours depending on the user listening histories. After that, the user's ratings for each song for the top five neighbours are weighted (if the song rating is 5, the weighting will be 20; while if the song rating is 4 or 3 the weighting will be 15 and 10 respectively; ratings of 2 and 1 are ignored in addition to the user current listening history will not be recommended again to the user). However, if the user has not given a rating to the songs, the default rating is used and a rating of 3 is given to the song, in order to reduce the sparsity and to improve the efficiency of the recommender system (Lampropoulos, 2015).

4.I. User-Based Collaborative Filtering Algorithm

1. Get vector for User U
2. **While** there are Users to compare Uc, **do**
3. Get vector for Uc
4. Calculate Cosine Similarity between U and Uc (Equation 3.4)
5. Sort the nearest neighbour list
6. **For** the top 5 nearest neighbours, **do**
7. **While** there are Songs S in listening history, **do**
8. **If** S rating is 5, **then**
9. Add S to recommendation with score of 20

Else if S rating is 4 **then**

Add S to recommendation with a score of 15

Else if S rating is 3 **then**

Add S to recommendation with score of 10

Else

Ignore S

End if

End while

End for

To illustrate 4.I algorithm, let's take this example: suppose we have five users as shown in the table 4.1 in below.

User 1			User 2			User 3			User 4			User 5		
Song	No. of times the song has been played	User Rate	Song	No. of times the song has been played	User Rate	Song	No. of times the song has been played	User Rate	Song	No. of times the song has been played	User Rate	Song	No. of times the song has been played	User Rate
Song 1	2	3	Song 1	2	3	Song 5	3	3	Song 7	2	3	Song 9	3	3
Song 2	2	5	Song 2	3	5	Song 6	3	5	Song 8	3	5	Song 10	3	5
Song 3	2	5	Song 3	3	4	Song 17	3	5	Song 25	2	4	Song 33	2	4
Song 4	2	3	Song 4	2	3	Song 18	2	3	Song 26	1	3	Song 34	2	3
Song 5	3	3	Song 11	2	3	Song 19	1	3	Song 27	1	3	Song 35	1	3
Song 6	3	3	Song 12	2	3	Song 20	1	2	Song 28	1	3	Song 36	2	3
Song 7	1	2	Song 13	2	3	Song 21	1	2	Song 29	2	3	Song 37	2	3
Song 8	3	4	Song 14	1	4	Song 22	1	4	Song 30	1	4	Song 38	1	3
Song 9	3	5	Song 15	1	3	Song 23	1	3	Song 31	1	2	Song 39	1	3
Song 10	3	4	Song 16	2	3	Song 24	1	4	Song 32	1	2	Song 40	1	2

Table 4.1: Example of measuring the nearest neighbours for a user

To measure the cosine similarity between user 1 and user 2, a vector will be created between these two users depends on the listening history and how many times that the user has listened to that song (term frequency).

Vector A (user 1) = [2 2 2 2 3 3 1 3 3 3 0 0 0 0 0 0]

Vector B (user 2) = [2 3 3 2 0 0 0 0 0 0 2 2 2 1 1 2]

$$Cosine\ Similarity\ (A, B) = \frac{A\,.B}{|A||B|}$$

$$Cosine\ Similarity\ (user\ 1, user\ 2) =$$

$$\frac{2\times2+2\times3+2\times3+2\times2}{\sqrt{2^2+2^2+2^2+2^2+3^2+3^2+1^2+3^2+3^2+3^2}\times\sqrt{2^2+3^2+3^2+2^2+2^2+2^2+2+1^2+1^2+2^2}} = 0.3394$$

$$Cosine\ Similarity\ (user\ 1, user\ 3) = 0.3758$$

$$Cosine\ Similarity\ (user\ 1, user\ 4) = 0.2689$$

$$Cosine\ Similarity\ (user\ 1, user\ 5) = 0.3708$$

Based on the above, the nearest users to the user 1 are user 3, user 5, user 2 and user 4. After that the songs that have rate smaller than 3 will be neglected and the rest of ratings will be selected and weighted and the top ten songs will be recommended to the user according to their weight.

4.3. Content-Based Recommendation Algorithm

In the music domain, Content-Based filtering ranks songs based on how similar they are to a seed song according to some similarity measure, which focuses on an objective distance between items and does not include any subjective factors. This makes it possible to recommend new items that do not have any user ratings associated with them. Additionally, there is no popularity bias as all items are considered to be of equal importance because user-generated data is not used to measure similarity (Celma, 2010; Kaminskas & Ricci, 2012).

The Content-Based strategy in the recommendation engine is the similar song recommendation algorithm which is finding songs similar in content to those in the user listening history. For each song in the current user's listening history there will be a number of similar songs, which are identified using the similar song score equation (equation 3.5 – Chapter 3). The top ten similar songs are selected and normalised to the scale of 100% and the result multiplied by 50% as weighting.

However, a problem will be encountered if two recommended songs have two or more common similar songs. If the scores were to be added then the results would be greater than 1 and it becomes difficult to be normalised to scale 100. Or if the average was taken any outliers would bring the score down. So a method was designed to combine the scores without skewing the results as shown in algorithm 4.II by removing the outliers.

4.II. Content-Based Song Similarity Algorithm

1. **While** a number of songs So in listening history, **do**
2.   **For** the similar songs SX get the top 10 and **do**

3. Put the SX in the recommendation list
4. Add the variance to list of variances in the recommendation list equation 4.1
5. Keep running total of variance squared and sum of variance recommendation
6. **End for**
7. **End while**
8. **While** there are recommendations R, **do**
9. Calculate standard deviation from equation 4.2
10. Remove the outliers (equation 4.3)
11. Set score to average score of song S
12. Add weighting onto score from equation 4.5
13. **End while**

$$Variance = Similar\ Song\ Score - Average\ Song\ Score \tag{4.1}$$

$$Standard\ Deviation = \sqrt{\frac{\sum Variances^2}{Number\ of\ similar\ song - 1}} \tag{4.2}$$

$$Outliers = Average\ Song\ Score \mp 2 \times Standard\ Deviation \tag{4.3}$$

Then, the final score will be determined using equation 4.3

$$Weighting = \frac{100 - Maximum\ Score}{Maximum\ number\ of\ occurrences} \times Number\ of\ occurrences \tag{4.4}$$

$$Final\ Score = Weighting + Average\ Score \tag{4.5}$$

Where the maximum number of occurrences is the maximum number of times a song appears in a recommendation list, maximum score is the highest recommendation score and number of occurrences is how often the song being recommended appears in the recommendation list. The final score is then multiplied by 100% and taken 50% as weighting.

To illustrate 4.II algorithm, let's take this example: suppose a user has five songs in his/her listening history (song A, song B, song C, song D and song E) and for each song there are ten similar songs with their score which is found according to the equation (equation 3.5 – Chapter 3) as shown in table 4.2:

Song A	Song B	Song C	Song D	Song E
Song 1 = 90%	Song 1 = 80%	Song 1 = 70%	Song 1 = 82%	Song 1 = 61%
Song 2 = 70%	Song 2 = 75%	Song 2 = 68%	Song 2 = 73%	Song 33 = 60.4%
Song 3 = 65%	Song 3 = 60%	Song 3 = 65%	Song 25 = 65.3%	Song 34 = 59.4%
Song 4 = 63%	Song 11 = 59.1%	Song 18 = 63.2%	Song 26 = 63.3%	Song 35 = 58.4%
Song 5 = 62%	Song 12 = 58.1%	Song 19 = 62.2%	Song 27 = 62.3%	Song 36 = 55.4%
Song 6 = 61%	Song 13 = 57.1%	Song 20 = 61.2%	Song 28 = 61.3%	Song 37 = 54.4%
Song 7 = 60%	Song 14 = 56.1%	Song 21 = 60.2%	Song 29 = 60.3%	Song 38 = 52.4%
Song 8 = 59%	Song 15 = 55.1%	Song 22 = 59.2%	Song 30 = 59.3%	Song 39 = 51.4%
Song 9 = 57%	Song 16 = 53.1%	Song 23 = 57.2%	Song 31 = 57.3%	Song 40 = 50.4%
Song 10 = 52%	Song 17 = 50.1%	Song 24 = 52.2%	Song 32 = 52.3%	Song 41 = 49.4%

Table 4.2: Example to illustrate the algorithm 4.II

By applying the equations (4.1) and (4.2) to measure the variance and the standard deviation for 50 songs (all the similar songs); the average score and the standard deviation are 60.002, 8.292 respectively and the outliers are 76.586 and 43.418. After removing the outliers, the songs with their scores will be as shown in table 4.3 below:

Song A	Song B	Song C	Song D	Song E
		Song 1 = 70%		Song 1 = 61%
Song 2 = 70%	Song 2 = 75%	Song 2 = 68%	Song 2 = 73%	Song 33 = 60.4%
Song 3 = 65%	Song 3 = 60%	Song 3 = 65%	Song 25 = 65.3%	Song 34 = 59.4%
Song 4 = 63%	Song 11 = 59.1%	Song 18 = 63.2%	Song 26 = 63.3%	Song 35 = 58.4%
Song 5 = 62%	Song 12 = 58.1%	Song 19 = 62.2%	Song 27 = 62.3%	Song 36 = 55.4%
Song 6 = 61%	Song 13 = 57.1%	Song 20 = 61.2%	Song 28 = 61.3%	Song 37 = 54.4%
Song 7 = 60%	Song 14 = 56.1%	Song 21 = 60.2%	Song 29 = 60.3%	Song 38 = 52.4%
Song 8 = 59%	Song 15 = 55.1%	Song 22 = 59.2%	Song 30 = 59.3%	Song 39 = 51.4%
Song 9 = 57%	Song 16 = 53.1%	Song 23 = 57.2%	Song 31 = 57.3%	Song 40 = 50.4%
Song 10 = 52%	Song 17 = 50.1%	Song 24 = 52.2%	Song 32 = 52.3%	Song 41 = 49.4%

Table 4.3: The result of table 4.3 after removing the outliers

By looking to the table 4.3, we will choose the maximum score and the maximum number of occurrence as in below:

Max. score = the highest recommendation score = 75%

Max. no. of occurrences = 4 (because song 2 has appeared 4 times)

By applying equations (4.3) and (4.4)

$$Final\ Score = \frac{100-Maximum\ Score}{Maximum\ number\ of\ occurrences} \times Number\ of\ occurrences + Average\ score$$

$$Final\ Score\ (song\ 1) = \frac{100-75}{4} \times 2 + \frac{70+61}{2} = 78$$

$$Final\ Score\ (song\ 2) = \frac{100-75}{4} \times 4 + \frac{70+75+68+73}{4} = 96.5$$

$$Final\ Score\ (song\ 3) = \frac{100-75}{4} \times 3 + \frac{65+60+65}{3} = 82.083$$

Therefore, the output of the algorithm 4.II for the above example will be the ten similar songs as shown in table 4.4 below:

Top ten recommended songs
Song 2 = 96.5%
Song 3 = 82.083
Song 1 = 75%
Song 25 = 65.3%
Song 26 = 63.3%
Song 18 = 63.2%
Song 4 = 63%
Song 27 = 62.3%
Song 19 = 62.2%
Song 5 = 62%

Table 4.4: The output for the algorithm 4.II for the table 4.2

4.4. Context-Based Recommendation Algorithms

The user context-based is the third strategy of the recommendation process implemented in the recommendation engine module; when the user has no listening history, it checks the user's context situation by checking the user's tweets, Facebook statuses, and current mood. For tweets and Facebook, all the terms are gathered and the cosine similarity between them and the song lyrics which are already collected from Chartlyrics and stored in the database is determined; then, the results are normalised to 100% and then take 15% as weighting. For the user's current mood, the user needs to first select the mood; the mood algorithm is then implemented and a score assigned, as shown in table 4.6. The final stage is to remove the songs where the year of release was before the user's date of birth (i.e. restrict songs to the user's lifetime). Research by Knobloch (2002) proposes mood management theory, which states that respondents in a 'bad mood' prefer exposure to highly energetic and joyful music over music low in these qualities to a higher degree than did respondents in a 'good mood'. Respondents in a neutral mood also preferred exposure to highly energetic and joyful music over music low in these qualities, to a greater extent than respondents in a good mood.

The CAPM system uses mood management theory to make recommendations of songs to users according to their mood. However, the proposed system has 18 mood categories therefore, a questionnaire (provided in Appendix C) was developed to capture users' listening tastes in relation to these 18 mood categories, according to mood management theory. In the questionnaire study, 96 students from the University of Portsmouth (54 female, 42 male) were asked about their music preferences when they felt calm, sad, happy, romantic, gleeful, and other moods. The results of this questionnaire are shown in figure 4.1 below

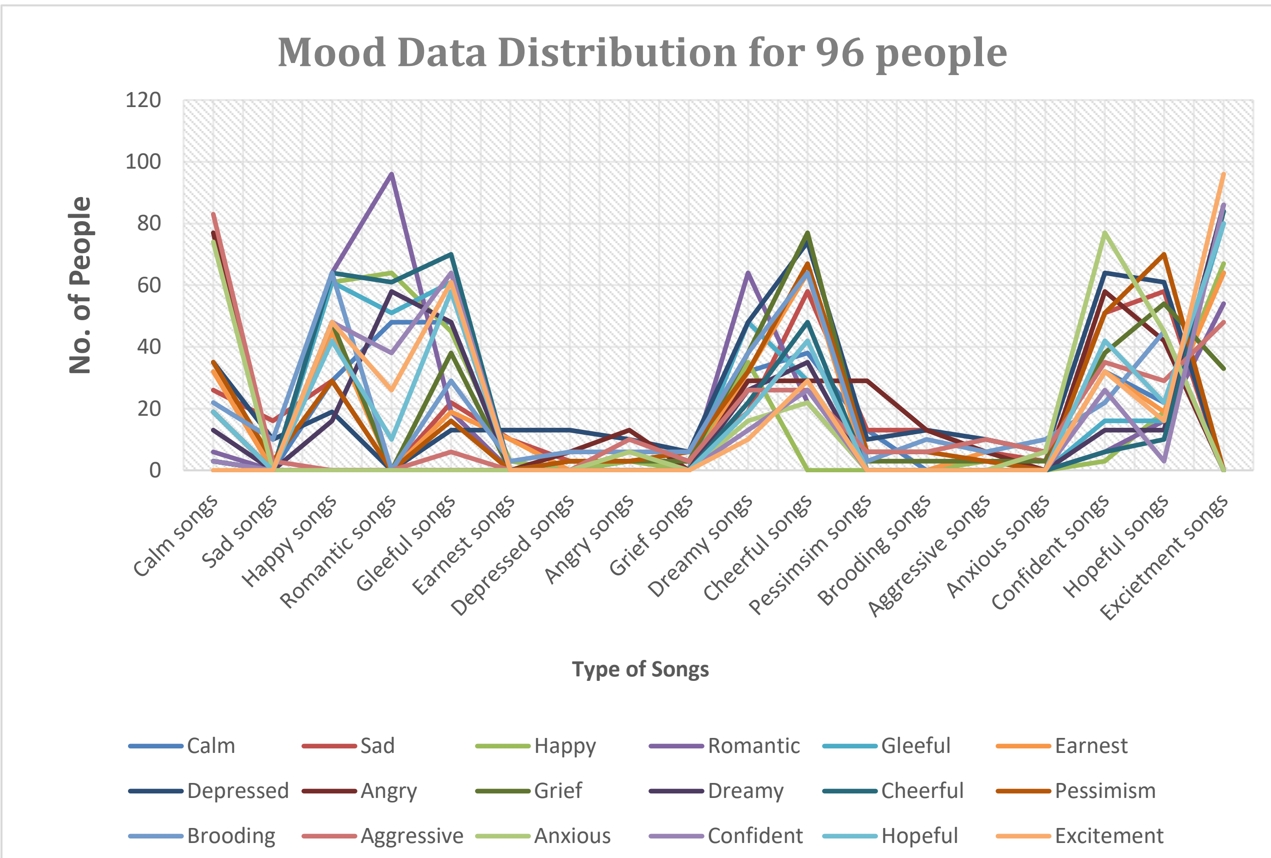


Figure 4.1: User songs preferences according to the mood situation

The graph above shows the preferences of the 96 respondents in terms of what type of music they preferred to listen to when they experience different moods. For instance, 64 and 48 respondents preferred to listen to exciting and gleeful songs, respectively, when their mood is calm, while 68 and 64 respondents would prefer to listen to exciting and romantic songs, respectively, when they are in a happy mood.

In the CAPM framework, the highest three values for each mood are chosen, and each is given a score. For instance, a user in calm mood will be recommended three types of songs, exciting, romantic, and gleeful, with scores of 15, 10, and 10, respectively, and so on with the other mood categories, as shown in table 4.5 in below:

Mood Number	Mood Name	Normalize Moods								
		Normalize Mood-1			Normalize Mood-2			Normalize Mood-3		
		Name	Value	Score	Name	Value	Score	Name	Value	Score
Mood 1	CALM	EXCITEMENT	64	15	ROMANTIC	48	10	GLEEFUL	48	10
Mood 2	SAD	CHEERFUL	58	15	HOPEFUL	58	15	CONFIDENT	51	10
Mood 3	HAPPY	EXCITEMENT	67	15	ROMANTIC	64	10	HAPPY	61	5
Mood 4	ROMANTIC	ROMANTIC	96	15	HAPPY	64	10	DREAMY	64	10
Mood 5	GLEEFUL	EXCITEMENT	80	15	HAPPY	61	10	GLEEFUL	61	10
Mood 6	EARNEST	EXCITEMENT	64	15	CHEERFUL	64	15	HAPPY	45	10
Mood 7	DEPRESSED	CHEERFUL	74	15	CONFIDENT	64	10	HOPEFUL	61	5
Mood 8	ANGRY	CLAM	77	15	CONFIDENT	58	10	HOPEFUL	42	5
Mood 9	GRIEF	CHEERFUL	77	15	HOPEFUL	54	10	HAPPY	48	5
Mood 10	DREAMY	EXCITEMENT	86	15	ROMANTIC	58	10	GLEEFUL	48	5
Mood 11	CHEERFUL	EXCITEMENT	84	15	GLEEFUL	70	10	HAPPY	64	5
Mood 12	PESSIMISM	HOPEFUL	70	15	CHEERFUL	67	10	CONFIDENT	51	5
Mood 13	BROODING	HAPPY	64	15	CHEERFUL	64	15	HOPEFUL	45	10
Mood 14	AGGRESSIVE	CALM	83	15	EXCITEMENT	48	10	CONFIDENT	35	5
Mood-15	ANXIOUS	CONFIDENT	77	15	CALM	74	10	HOPEFUL	45	5
Mood 16	CONFIDENT	EXCITEMENT	86	15	GLEEFUL	64	10	CHEERFUL	48	5
Mood 17	HEPEFUL	EXCITEMENT	80	15	GLEEFUL	42	10	CHEERFUL	42	10
Mood 18	EXCITEMENT	EXCITEMENT	96	15	GLEEFUL	61	10	HAPPY	48	5

Table 4.5: Mood normalization and scores

4.III. Mood-Based Recommendation Algorithm

1. Get the user's current mood
2. **While** there are songs S in the song list, do
3. Get the song mood type
4. **If** the song mood type fits with the normalised mood table, **then**
5. Add the song to the recommended list
6. **End if**
7. **End while**

4.5. Recommendation Engine Algorithms

When gathering the Collaborative Filtering, the Content-Based and the context-based, the recommendation engine algorithm is as shown below:

4.IV. Recommender Engine Algorithm

1. **If** the listening history of user U is not empty, **then**
2. Determine the user-based Collaborative Filtering algorithm to collaboratively add recommendations
3. Determine the Similar Songs for each song in the Listening History
4. Add highest ranked similar songs to the recommendations list
5. Get songs related to the user's context (Facebook, tweets, mood)
6. **Else**
7. Get songs related to the user's context (Facebook, tweets, mood)
8. **End if**
9. Give the recommended songs to the user

However, there is a need for a recommendation requestor module in order to ask for a recommendation from the server handler; in response, a list is reproduced and then downloaded and stored in the client's repository. This list will be displayed for the user to pick which songs to play.

4.V. Recommendation Requester

1. Request Recommendation
2. Read how many recommendations have been made
3. Initialise counter at 0
4. While there exists Recommendations R to be downloaded from Server, do
5. Save the Recommendations to a recommendation list
6. Show Recommendations in panel counter
7. Increment counter
8. End while
9. Display Recommendation view

4.6. Summary

This chapter described the hybrid multi-strategy approach to address the Cold-Start problem (new user and new item). The multi-strategy approach combined three recommendation methods –. The key contribution of the approach is an automatic prediction technique to determine user's preference. This prediction technique involves three stages: content- based filtering, collaborative filtering and context-based recommendation – with different scores calculated at each. While there are several profiling and recommendation approaches to determine user's need and recommend appropriate items, our approach is simple and customized for music recommendation tasks.

Chapter 5

CAPM Implementation and Evaluation

A vital component of recommendation systems research and development is the evaluation, which means measuring whether the recommended items are useful and effective. In order to demonstrate and to evaluate the use of CAPM Framework and proposed recommendation algorithms, a context aware personalised music (CAPMusic) recommender application was developed for mobile devices. In this chapter, we describe experimental settings appropriate for testing the recommendation algorithms. We started by publishing the CAPMusic application on Google Play App where real users can easily download it and interact with the system. In this case, we described types of questions (questionnaire) that could be answered to be used later on in the evaluation metrics measurements. We then benchmarked CAPMusic with Last.fm and Spotify and compared the results obtained from the same users who used CAPMusic application.

5.1. CAPMusic

CAPMusic was built based on CAPM framework and implemented using Java programming language and Android studio software. The functions of the CAPMusic client are:

- Playme a Song
- Song Chart
- Loved Song
- Find Song
- Mood
- Playlists

These functions were successfully implemented and provide the framework needed for context-aware music recommendations. CAPMusic user interface is shown in Appendix A which shows CAPMusic functions:

5.1.1. Playme a Song

If the user wants to discover new songs, he/she needs to use "Playme a song". By selecting this function, the recommendation engine will be running (algorithm 4.5a – step no. 6 – chapter 4). If the user has no listening history then the results will be the top ten recommended songs according to the user's tweets and Facebook status but when the user plays at least one song the results will be according to the user's listening history in addition to the user's tweets and Facebook status.

5.1.2. Song Chart

Sometimes the users want to tell their friends about songs that have been recommended to them while using the music app. This function displays the songs that have been most recently listened to by the user and enables the user to play the songs again by watching them as a video clip via YouTube.

5.1.3. Loved Song

This function displays the user's top ten favourite songs based on their ratings in addition to how many times a song has been actively played.

5.1.4. Find Song

The recommendation system in this research is based on the user's listening history, tweets, Facebook status, and the user's current mood. However, if the user has no data (no listening history, no tweets, no Facebook status, and no selected mood) the system will be unable to recommend any songs to the user unless the user searches for a song. The "Find Song" function is used for this purpose, in order to give the user the opportunity to search by song name and to display the top ten songs results for that search (algorithm 4.5a – step no. 1 – chapter 4).

5.1.5. Mood

This function enables the users to select their mood manually. There are 18 mood categories; the users select their mood, and then the system will recommend the top ten songs associated with that mood, taking into account the other recommendation factors (listening history, tweets, and Facebook status) (algorithm 4.5a – step no. 1 – chapter 4).

5.1.6. Playlists

The purpose of playlist function is to generate a music sequence that matches the user preferences. The user can use this function to create a playlist according to their preferences. Users can create their own playlists by searching for songs manually and they can add them to their custom playlist. Users can also create a name for their playlist and write a description. After that and by playing a number of the songs in the playlist, the system will have all the needed information (i.e. user listening history and user context) and the system will provide the user with more accurate results.

5.2. CAPMusic online publication

CAPMusic was published on the Google Play App Store at the time of writing the PhD thesis as shown in Appendix A; the App can be located through the following links:
[https://play.google.com/apps/testing/com.etj.CAPMusic1]
However, beta testing was carried out which enables developers to test out new versions of software and apps before they are released publicly by inviting groups of people for the purposes of evaluating the App performance.
For this study, 53 students from the University of Portsmouth tested CAPMusic App. A questionnaire was also created (as shown in Appendix B) so that users could test the system and

give ratings for the recommended songs based on whether the song was new to them, and if the user liked the song. This was carried out first without the mood selection, and second, with the user selecting a mood. The table below shows a sample of the 53 test users.

User ID	No. of Rec. (without mood)	Average of Rec. (without mood)	Average Rating (without mood)	No. of Rec. (with mood)	Average of Rec. (with mood)	Average Rating (with mood)	App Rating
1	4	3.0 3.0 3.2 3.3	3.125	4	3.4 3.5 4.5 4.6	4.0	4
2	3	3.9 4.1 4.3	4.1	4	4.2 4.1 4.1 4.1	4.125	5
3	4	3.1 3.3 3.8 4.4	3.65	4	3.0 3.3 3.6 4.3	3.55	3
4	3	3.5 3.5 3.5	3.5	3	3.4 3.4 3.9	3.567	4
5	3	3.0 3.3 3.5	3.267	3	3.3 3.7 4.1	3.7	4

Table 5.1: Recommendation Testing Sample

It is clear that the majority of users began by giving ratings with low values to the recommended songs when these were recommended without the mood selection; when users were starting to select their mood, they began to give better ratings, which indicates that the user's approval of the recommendations improved. The average recommendation ratings across the 53 users was 3.578 (without the mood) and 3.684 (with the mood selection), which shows that most of the recommended songs were liked by the test users.

The results were analysed and showed positive findings. As has already been mentioned, CAPMusic is based on the user's music listening history, in addition to other factors; therefore, when the system makes a recommendation to the user it is the intention that the accuracy of and user satisfaction with the recommendations will increase over time (Knijnenburg et. al., 2012). Evidence to support this is very clear in the graph below, which shows that the song ratings increased as the participants used the recommendation request more times.

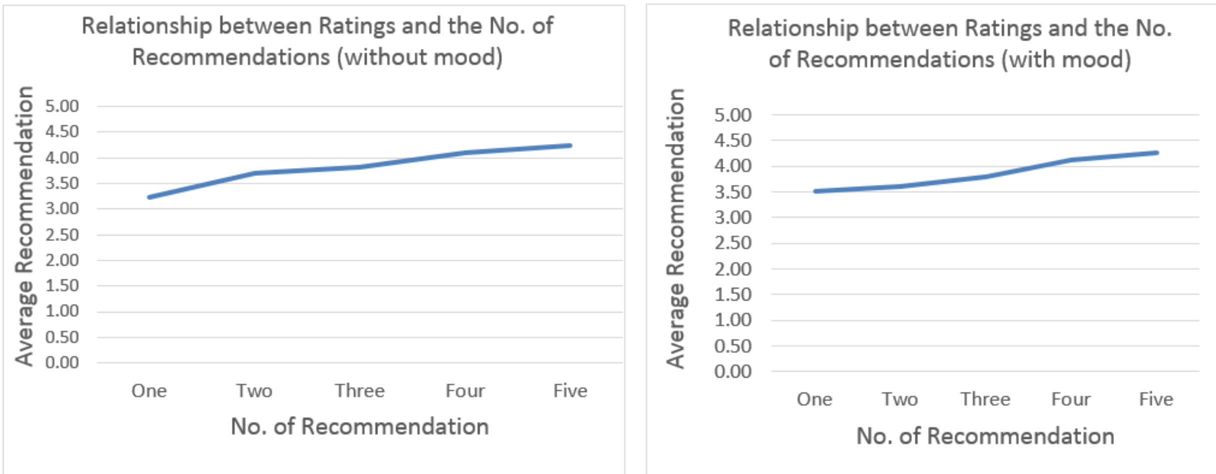


Figure 5.1: Relationship between ratings and the no. of recommendations (with & without mood)

5.3. Evaluation Metrics

Evaluation metrics for recommender systems can be divided into two major classes: 1) accuracy metrics, 2) recovery and novelty measurements metrics. Accuracy metrics try to assess the successful decision-making capacity (SDMC) of recommendation algorithms. They measure the amount of correct and incorrect classifications as relevant or irrelevant items that are made by the recommender system and are therefore useful for user tasks such as finding good items. A rank accuracy or ranking prediction metric measures the ability of a recommender to estimate the correct order of items concerning the user's preference, which is called the measurement of rank correlation in statistics. Therefore, this type of measure is most adequate if the user is presented with a long ordered list of items recommended to him. A rank prediction metric uses only the relative ordering of preference values so that is independent of the exact values that are estimated by a recommender. The recovery and novelty measurements metrics are about measuring of a piece of information which refers to how the recommended items are accurate being ranked and being different based on “what has been previously seen”. Evaluating recommender systems requires a definition of what constitutes a good recommender system, and how this should be measured.

5.3.1. Accuracy Metrics

The RS intention is to offer choice preferences associated with the context of the environment, as well as being based entirely on user preferences and actions. However, user satisfaction can fluctuate according to what the user wants to achieve. This section will examine the most established assessment of user satisfaction metrics.

5.3.1.1. Accuracy of Estimated Ranking

The accuracy of the recommended items positioned close to the top of the recommendation list is vital for various information retrieval systems, such as recommendation systems or search engines, where the majority of users of these systems focus only on browsing or choosing amongst the top *k* items from the list of recommended items. In the past, different methods have been developed to measure the quality of estimated rankings. One of these methods is Spearman's ranking correlation, which is a method based on calculating the correlation between the predicted and the true ranking. In this case, the predicted values are the system ranking values, while the true ranking is the ranking that the user themselves gives to the recommended items.

$$r = 1 - \frac{6}{n\,(n^2-1)} \sum_{i=1}^{n} (x_i - y_i)^2 \tag{5.1}$$

Where:

r: Spearman's ranking correlation, n: no. of recommended items, x_i: ranks of item i output by RS, y_i: ranks of item i provided by user.

The following example shows how to calculate the Spearman's ranking correlation for three users.

User 1		User 2		User 2	
Rec. List	User Ranking	Rec. List	User Ranking	Rec. List	User Ranking
Song A1	4	Song B1	1	Song C1	3
Song A2	7	Song B2	3	Song C2	1
Song A3	5	Song B3	2	Song C3	2
Song A4	3	Song B4	4	Song C4	4
Song A5	6	Song B5	6	Song C5	5
Song A6	1	Song B6	5	Song C6	7
Song A7	2	Song B7	10	Song C7	6
Song A8	8	Song B8	8	Song C8	8

Song A9	9	Song B9	7	Song C9	9
Song A10	10	Song B10	9	Song C10	10

Table 5.2: Example for calculating the accuracy of estimated ranking

$$r\ (for\ user\ 1) = 1 - \frac{6}{10\ (100-1)}[(1-4)^2 + (2-7)^2 + (3-5)^2 + (4-3)^2 + (5-6)^2 + (6-1)^2 + (7-2)^2 + (8-8)^2 + (9-9)^2 + (10-10)^2] = 0.454545$$

$$r\ (for\ user\ 2) = 1 - \frac{6}{10\ (100-1)}[(1-1)^2 + (2-3)^2 + (3-2)^2 + (4-4)^2 + (5-6)^2 + (6-5)^2 + (7-10)^2 + (8-8)^2 + (9-7)^2 + (10-9)^2] = 0.89091$$

$$r\ (for\ user\ 3) = 1 - \frac{6}{10\ (100-1)}[(1-3)^2 + (2-1)^2 + (3-2)^2 + (4-4)^2 + (5-5)^2 + (6-7)^2 + (7-6)^2 + (8-8)^2 + (9-9)^2 + (10-10)^2] = 0.95151$$

Based on the above, the average Spearman's ranking correlation for the three users is 0.765655. For the questionnaire (shown in Appendix B), 53 students from the University of Portsmouth were asked to test the recommendation list and rank the recommended songs according to their preferences. Based on the questionnaire results, the average Spearman's ranking correlations across all 53 students, without and with mood selection, were 0.68173 and 0.86384, respectively. This indicates that the recommendations given where mood was selected produced the best results.

5.3.1.2. List Relevance

In the ideal information retrieval system, items should be ranked in order of probability of their relevance or usefulness. Most IR and RS follow this principle, and the results are presented to the user in the form of a list. There are several methods that have been developed and used in the past to measure relevance. One of these methods is 'average precision' (AP); AP is the average of the precision values obtained from the set of the top k items existing after each relevant item is retrieved for single query (for one recommendation list). If there is a set of queries (many recommendation lists), it is necessary to determine the mean average precision (MAP).

$$Average\ Precision\ (AP) = \frac{1}{M}\sum_{i=1}^{n} rel(k) \times P_{rec}@k \quad (5.2)$$

$$Mean\ Average\ Precision\ (MAP) = \frac{1}{m}\sum_{m} AP_m \quad (5.3)$$

Where:

M: total number of relevant items

n: the list length

$rel(k)$: 1 if relevant, otherwise 0

$P_{rec}@k$: precision at each rank at a value of 3 and above

m: number of queries

Returning to the table in Section 5.2.1.1, the average precision is determined as shown below:

User 1		User 2		User 2	
Rec. List	Song Rate	Rec. List	Song Rate	Rec. List	Song Rate
Song A1	3	Song B1	3	Song C1	3
Song A2	2	Song B2	3	Song C2	3
Song A3	3	Song B3	4	Song C3	3
Song A4	3	Song B4	3 (new)	Song C4	3
Song A5	3	Song B5	3	Song C5	3
Song A6	4 (new)	Song B6	3	Song C6	2
Song A7	3	Song B7	2	Song C7	3
Song A8	2	Song B8	3	Song C8	2
Song A9	2	Song B9	2	Song C9	2
Song A10	2	Song B10	3	Song C10	2

Table 5.3: Example for calculating the List Relevance in the recommended list

$$average\ precision\ for\ user\ 1 = \left[\frac{\frac{1}{1} + 0 + \frac{2}{3} + \frac{3}{4} + \frac{4}{5} + \frac{5}{6} + \frac{6}{7} + 0 + 0 + 0}{6}\right] = 0.81786$$

$$average\ precision\ for\ user\ 2 = \left[\frac{\frac{1}{1} + \frac{2}{2} + \frac{3}{3} + \frac{4}{4} + \frac{5}{5} + \frac{6}{6} + 0 + \frac{7}{8} + 0 + \frac{8}{10}}{8}\right] = 0.959375$$

$$average\ precision\ for\ user\ 3 = \left[\frac{\frac{1}{1} + \frac{2}{2} + \frac{3}{3} + \frac{3}{4} + \frac{5}{5} + 0 + \frac{6}{7} + 0 + 0 + 0}{6}\right] = 0.97619$$

$$mean\ average\ precision\ for\ user\ 1, 2, 3 = \left[\frac{0.81786 + 0.959375 + 0.97619}{3}\right] = 0.91781$$

Thus, the average precisions for all 53 students, without and with mood selection, are 0.70026 and 0.72483 respectively. This means that the recommendation list produced with the mood selection produced the best result.

5.3.1.3. Accuracy of Ranking Position

The best recommender systems should be able to rank the relevant items in the top of the recommended list. When the mean reciprocal rank (MRR) calculation is applied to the recommender system, it can be used to determine whether the system positions the user's favourite items at the top of the list. The MRR is calculated as follows:

$$MRR = \frac{1}{|D_{rel}|} \sum_{i=1}^{N} \frac{rel_i}{i} \quad (5.4)$$

Where:

$|D_{rel}|$: the set of items preferred by the user, N: the length of the recommendation list, rel_i: whether the item is preferred by the user (1 or 0), i: the item rank in the recommended list.

Again returning to the table from Section 5.2.1.1, the mean reciprocal rank (MRR) can be determined as shown below:

User 1		User 2		User 2	
Rec. List	Song Rate	Rec. List	Song Rate	Rec. List	Song Rate
Song A1	3	Song B1	3	Song C1	3
Song A2	2	Song B2	3	Song C2	3
Song A3	3	Song B3	4	Song C3	3

Song A4	3	Song B4	3 (new)	Song C4	3
Song A5	3	Song B5	3	Song C5	3
Song A6	4 (new)	Song B6	3	Song C6	2
Song A7	3	Song B7	2	Song C7	3
Song A8	2	Song B8	3	Song C8	2
Song A9	2	Song B9	2	Song C9	2
Song A10	2	Song B10	3	Song C10	2

Table 5.4: Example for calculating the accuracy of ranking position in the recommended list

$$MRR\ for\ user\ 1 = \left[\frac{\frac{1}{1}+\frac{0}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\frac{1}{6}+\frac{1}{7}+\frac{0}{8}+\frac{0}{9}+\frac{0}{10}}{6}\right] = 0.34881$$

$$MRR\ for\ user\ 2 = \left[\frac{\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{0}{5}+\frac{1}{6}+\frac{0}{7}+\frac{1}{8}+\frac{1}{9}+\frac{1}{10}}{8}\right] = 0.32326$$

$$MRR\ for\ user\ 3 = \left[\frac{\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\frac{0}{6}+\frac{1}{7}+\frac{0}{8}+\frac{0}{9}+\frac{0}{10}}{6}\right] = 0.40437$$

$$average\ RMM\ for\ user\ 1,2,3 = \left[\frac{0.34881+0.32326+0.40437}{3}\right] = 0.35881$$

Thus, the average RMMs for all 53 students without and with mood selection are 0.53423 and 0.58364, respectively. Again, this indicates that mood selection produces the best result.

5.3.2. Recovery and Novelty Measurements

In order to be able to evaluate the overall performance of the recommendation algorithms regarding whether or not they produce an accurate ranking of the recommended items, the recovery and novelty measurements are employed. The recommender system is considered to be good if it assigns a higher ranking to items that are relevant to the users. The relevant items are those that the user assigns a "Like" rating (3, 4 and 5). Therefore, the recovery measurement R can be determined as follows:

$$R = \frac{\sum_{u \in u_{TestSet}} \frac{1}{L_u} \sum_{i=1}^{L_u} \frac{r_i}{C_u}}{|u_{TestSet}|} \tag{5.5}$$

Where Cu is the number of recommended items in the recommendation list, Lu is the number of relevant items in the recommendation list, ri is the position for item i in the ranked list for user u, and |uTestSet| is the number of users in the test dataset. According to this definition of recovery, the lower R is the more accurate system.

The recommended songs become more useful and novel when the system provides relevant and new songs to the users (not popular). Several methods of evaluating the novelty of recommendations have been introduced. This research utilised the self-information-based novelty measurement method, which measures novelty relative to the popularity of items. According to this measure, popular items provide less novelty. Self-information-based novelty can be calculated as follows:

$$S(N) = \frac{\sum_{u \in u_{TestSet}} \frac{RS_{rel\,(new)}}{RS}}{|u_{TestSet}|} \tag{5.6}$$

Where: $S(N)$: self-information-based novelty, $RS_{rel\,(new)}$: how many songs in the recommendation list are relevant and new for each user, RS : the number of recommended songs in the recommendation list and $|u_{TestSet}|$: the total number of users testing the system.

In the questionnaire (provided in Appendix B), users were asked to identify if a song was new or not, and if they liked it or not. The table below, the same table provided in Section 5.2.1.1, shows a sample of three users who tested the CAPMusic App and gave feedback that was used to calculate the recovery and novelty values, as shown below:

User 1		User 2		User 2	
Rec. List	Song Rate	Rec. List	Song Rate	Rec. List	Song Rate
Song A1	3	Song B1	3	Song C1	3
Song A2	2	Song B2	3	Song C2	3
Song A3	3	Song B3	4	Song C3	3
Song A4	3	Song B4	3 (new)	Song C4	3
Song A5	3	Song B5	3	Song C5	3
Song A6	4 (new)	Song B6	3	Song C6	2
Song A7	3	Song B7	2	Song C7	3
Song A8	2	Song B8	3	Song C8	2
Song A9	2	Song B9	2	Song C9	2

Song A10	2	Song B10	3	Song C10	2

Table 5.5: Example for calculating the recovery and novelty in the recommended list

$$R = \frac{\left(\frac{1}{10}+\frac{3}{10}+\frac{4}{10}+\frac{5}{10}+\frac{6}{10}+\frac{7}{10}\right)+\left(\frac{1}{10}+\frac{2}{10}+\frac{3}{10}+\frac{4}{10}+\frac{6}{10}+\frac{9}{10}+\frac{10}{10}\right)+\left(\frac{1}{10}+\frac{2}{10}+\frac{3}{10}+\frac{4}{10}+\frac{5}{10}+\frac{7}{10}\right)}{3} = 2.76667$$

$$S(N) = \frac{\frac{1}{10}+\frac{1}{10}+\frac{0}{10}}{3} = 0.06667$$

The recovery (R) value for all 53 students without mood selection is 0.54582, and with mood selection is 0.32681. This indicates that mood selection produced the best result. The novelty (S(N)) values for the 53 users without and with mood selection are 0.62435 and 0.66782 respectively. This also indicates that mood selection produced the best result.

5.4. Benchmarking

Comparison is considered to be an effective means of obtaining more accurate evaluation scores for recommender systems. However, it is difficult to compare the results obtained from different recommender systems due to the various alternatives in this thesis and the implementation of an evaluation approach. Therefore, 53 real users have used the CAPMusic app. Some users had common songs but the majority of users had entirely different listening histories. Each song in the listening history has certain attributes (song title, artist name, song ID, artist ID, album name, year of release, song rating, and play count). Benchmarking was carried out in several different ways, as explained below, and in each step the results obtained are compared with two well-known music recommender websites (Last.fm and Spotify) to measure the similarity between CAPMusic's recommendations and those of the established sites, in order to ensure that the recommendations are successful.

The same 53 users who tested CAPMusic also tested Last.fm and Spotify and all 53 users completed the questionnaire. From the pie charts shown below, it is clear that a large portion of the users who used CAPMusic (without mood) liked the recommended songs - 53% of the recommendations were rated 3, while 28% of the recommendations were rated 4; 10% and 7% of recommendations were rated 5 and 2, respectively, and only 2% were rated 1.

Moreover, the figure 5.2 below shows that the recommendation results improved after the mood selection stage was added to the recommendation, where the majority of the users (45%) rated the recommendation as 4. However, 35% rated the recommendation 3, and there was a noticeable increase in ratings of 5 from before (17%), and 2%, and 1% of the recommendations were rated 2 and 1, respectively.

If the ratings 5, 4, and 3 are combined, it is evident that the users preferred CAPMusic to Spotify and to Last.fm. In total, 97% of users liked CAPMuisc (with mood), and 91% of them liked it without mood selection, whereas 89% and 88% of the 53 users liked Spotify and Last.fm, respectively.

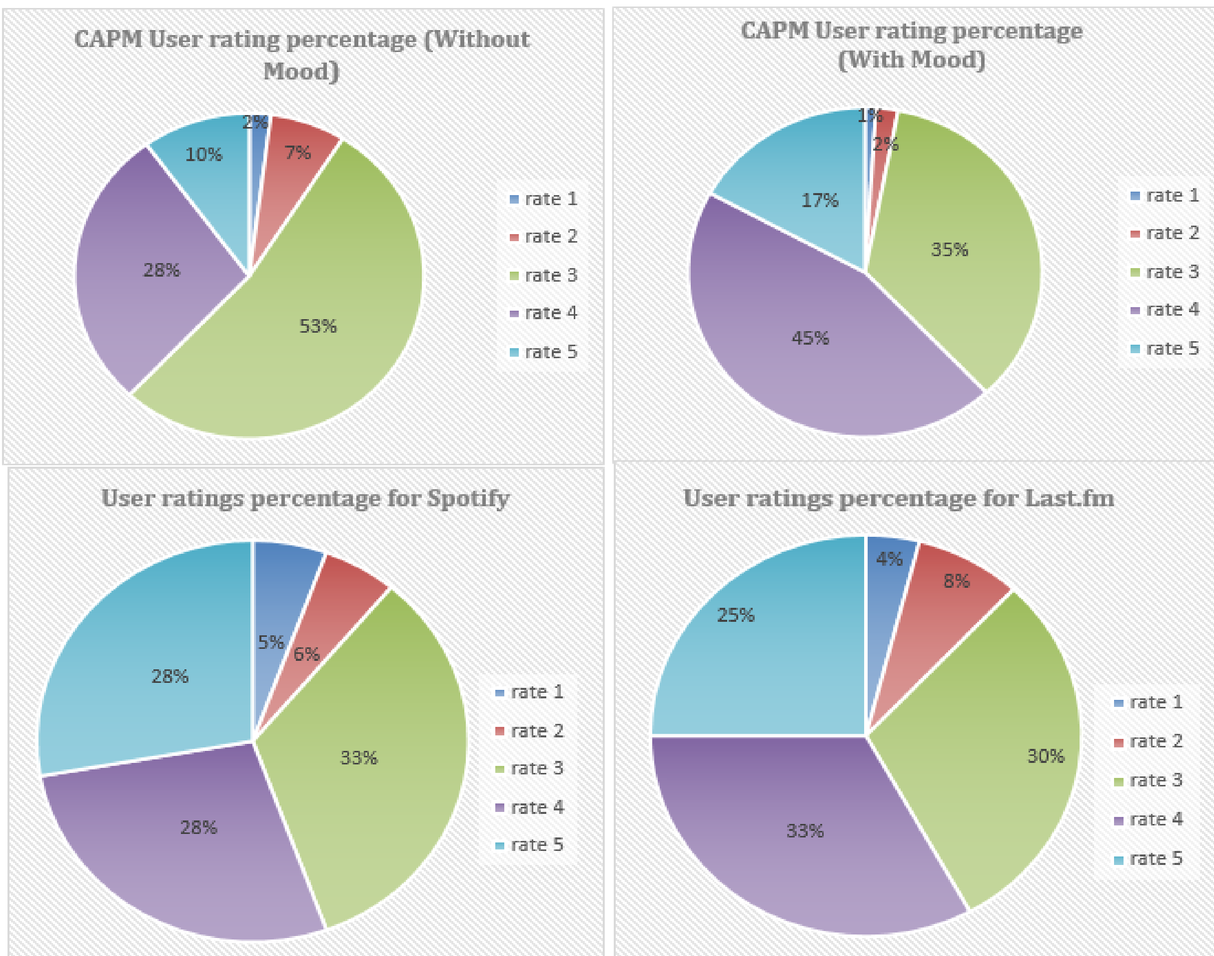


Figure 5.2: Recommendation results for CAPMusic, Spotify and Last.fm

Furthermore, the evaluation metrics were calculated for Spotify and Last.fm and compared with those of CAPM; the results are shown in the table below.

Measurement Metrics	CAPMusic		Last.fm	Spotify
	Without Mood	With Mood		
Spearman's ranking correlation	0.68173	0.86384	0.57621	0.66841
Mean average precision (AMP)	0.70026	0.72483	0.43268	0.62865
Mean reciprocal rank (MRR)	0.53423	0.58364	0.51268	0.52364
Recovery (R)	0.54582	0.32681	0.51429	0.55432
Novelty (N)	0.62435	0.66782	0.56248	0.57634

Table 5.6: Measurement metrics comparison between CAPMusic, Last.fm and Spotify

The table above shows that the Spearman's ranking correlation, the mean average precision (AMP), the mean reciprocal rank (MRR), and the novelty (N) of CAPMusic (with mood) are greater than those of CAPMusic (without mood), Last.fm, and Spotify. On the other hand, the recovery (R) of CAPMusic (with mood), at 0.32681, is significantly lower than for CAPMusic (without mood), Last.fm, and Spotify, at 0.54582, 0.51429, and 0.55432, respectively. This implies that CAPMusic (with mood) produces the best results - as has already been explained, the lower the value, the better the system.

5.4.1. Last.fm

1- Initially, it was assumed that the user has one song in their listening history; this song was chosen based on its popularity. In the first instance, the user's current mood is not selected; then, the user's current mood is selected with the same song, and the user's current mood is "Happy", as shown in table 5.5 below. The table below shows the songs recommended by both CAPMusic and Last.fm for the specific song.

No.	Song Listened To	CAPMusic Rec. (without mood)	CAPMusic Rec. (with mood)	Last.fm Rec.
1	Cheap Thrills - Sia	Snow In California - Ariana Grande	Snow In California - Ariana Grande	The Greatest - Sia
		Sparks Fly - Taylor Swift	Sparks Fly - Taylor Swift	Chained to the Rhythm - Katy Perry
		Poker Face - Lady Gaga	Entertainment – Sean Paul	Swish Swish - Katy Perry

		I Kissed A Girl - Katy Perry	I Kissed A Girl - Katy Perry	Aura - Lady Gaga
		A Place In This World - Taylor Swift	A Place In This World - Taylor Swift	Worth It - Fifth Harmony
		Aura - Lady Gaga	Blue – Calvin Harris	Rockabye - Anne-Marie
		If It's Lovin' That You Want - Rihanna	Dreams For Plans – Shakira	Applause - Lady Gaga
		We Ride - Rihanna	We Ride - Rihanna	Snow In California - Ariana Grande
		Shake It Off - Taylor Swift	Shake It Off - Taylor Swift	I Got You - Bebe Rexha
		Thinking Of You - Katy Perry	Thinking Of You - Katy Perry	Shake It Off - Taylor Swift

Table 5.7: The common songs between CAPMusic & Last.fm for one song

The items highlighted in yellow are the common recommended songs present in both CAPMusic and Last.fm's recommendations lists.

As the diversity of the recommender system determines whether the system can recommend various types of items to users, the difference between the two recommendation lists can be calculated using equation (5.7):

$$d_{i,j} = 1 - \frac{c_{i,j}}{N} \qquad (5.7)$$

Where $d_{i,j}$ is the distance between the two recommendation lists, $c_{i,j}$ is the common songs, and N is the size of the recommendation lists. The recommendation diversity is the average of the distance. Therefore, the distance between the two recommended lists without and with mood selection is equal to 0.85 and 0.9, respectively.

This experiment was repeated ten times with ten different songs, and the distance as calculated for each iteration; the results are shown in the table 5.6 in below.

No.	Song Name	Distance Between Last.fm and		No.	Song Name	Distance Between Last.fm and	
		CAPM without mood	CAPM with mood			CAPM without mood	CAPM with mood
1	One Kiss – Dua Lipa	0.8	0.8667	6	New Rules – Dua Lipa	0.8571	0.8571
2	Nice for What - Drake	0.6667	0.75	7	No Tears Left to Cry – Ariana Grande	0.7857	0.8571
3	Shape of You – Ed Sheeran	0.5833	0.5833	8	Delicate – Taylor Swift	1.0	1.0
4	Havana - Camila	0.8235	0.8235	9	Symphony – Zara Larson	0.8462	0.9231
5	Back to You - Selena Gomez	0.7647	0.8235	10	Hello - Adele	1.0	1.0

Table 5.8: The distance calculations between CAPMusic & Last.fm for specific songs

From the above table, it is clear that there is significant diversity between CAPMusic (without and with mood) and Last.fm, as the distance between Last.fm and CAPMusic is 1.0 (Hello – Adele) and (Delicate – Taylor Swift) which means there are no common songs. The diversity in the recommendations was calculated for the above results, yielding 0.81272 (without mood) and 0.84843 (with mood).

2- Step 1 was then repeated, this time based on the assumption that the user has five songs in their listening history; the test was run once when the user had not chosen their current mood, and then again, with the same listening history, but with the current mood selected as “Sad”. The results are shown in the table below, enabling a comparison between the recommendations made by CAPMusic and Last.fm.

No.	Song Listened To	CAPMusic Rec. (without mood)	CAPMusic Rec. (with mood)	Last.fm Rec.
1	Honeymoon Avenue – Ariana Grande	Love Story - Taylor Swift	The Climb - Miley Cyrus	Hate That I Love You - Rihanna
		Fearless - Taylor Swift	Blown Away - Carrie Underwood	Don't Stop the Music - Rihanna
2	You Belong with me – Taylor Swift	The Climb - Miley Cyrus	Love Story - Taylor Swift	Crazy in Love - Beyonce
		Blown Away - Carrie Underwood	Fearless - Taylor Swift	Promiscuous - Nelly Furtado
3	Shut Up and Drive – Rihanna	Breakin' - Rihanna	Paparazzi - Lady Gaga	Halo - Beyonce
		Toxic - Britney Spears	Breakin' - Rihanna	Daydreamin' - Ariana Grande
4	Poker Face – Lady Gaga	Sweet Dreams - Beyonce	Toxic - Britney Spears	Baby I - Ariana Grande
		Gimme More - Britney Spears	Sweet Dreams - Beyonce	The Climb - Miley Cyrus
5	Whenever, Wherever - Shakira	Paparazzi - Lady Gaga	Gimme More - Britney Spears	Worth It - Fifth Harmony
		Bad Romance - Lady Gaga	Bad Romance - Lady Gaga	The Heart Wants - Selena Gomez

Table 5.9: The common songs between CAPMusic & Last.fm for no. of songs

The distance between the Last.fm and CAPM (without and with mood) recommendations is 0.95. This experiment was repeated ten times with ten different songs, and the distance was calculated each time; the results are shown in the table 5.8 in below.

No.	Distance between Last.fm and		No.	Distance between Last.fm and	
	CAPMusic without mood	CAPMuisc with mood		CAPMusic without mood	CAPMusic with mood
1	0.5833	0.5833	6	0.7857	0.8571
2	0.7647	0.8235	7	0.8462	0.9231
3	1.0	1.0	8	0.8	0.8667
4	1.0	1.0	9	0.5833	0.5833
5	0.8571	0.8571	10	0.6667	0.75

Table 5.10: The distance calculations between CAPMusic & Last.fm

However, the diversity of recommendations was calculated for the above results to be 0.7887 (without mood) and 0.82441 (with mood).

5.4.2. Spotify

1- Step 1 in 5.2.1 was repeated, this time using Spotify. The Spotify API enabled the researcher to retrieve the recommended songs list for a specific song played by the user.

No.	Song Listened To	CAPMusic Rec. (without mood)	CAPMusic Rec. (with mood)	Spotify Rec.
1	Cheap Thrills - Sia	Snow In California - Ariana Grande	Snow In California - Ariana Grande	This Is Acting - Sia
		Sparks Fly - Taylor Swift	Sparks Fly - Taylor Swift	Midnight Decisions, Artist - Sia
		Poker Face - Lady Gaga	Entertainment – Sean Paul	Lady Wood - Tove Lo
		I Kissed A Girl - Katy Perry	I Kissed A Girl - Katy Perry	WTF Love Is - Tove Lo
		A Place In This World - Taylor Swift	A Place In This World - Taylor Swift	The Fame - Lady Gaga
		Aura - Lady Gaga	Blue – Calvin Harris	Poker Face - Lady Gaga

		If It's Lovin' That You Want - Rihanna	Dreams For Plans – Shakira	Animal - Kesha
		We Ride - Rihanna	We Ride - Rihanna	Dreams For Plans – Shakira
		Shake It Off - Taylor Swift	Shake It Off - Taylor Swift	The Truth About Love - Pink
		Thinking Of You - Katy Perry	Thinking Of You - Katy Perry	Try - Pink

Table 5.11: The common songs between CAPMusic & Spotify for one song

Based on the above, the distance between the two recommended lists is 0.95 both without and with the mood selection. This experiment was repeated ten times with ten different songs, and the distance was determined each time; the results are shown in the table 5.10 in below.

No.	Song Name	Distance Between Spotify and		No.	Song Name	Distance Between Spotify	
		CAPMusic without mood	CAPMusic with mood			CAPMusic without mood	CAPMusic with mood
1	One Kiss – Dua Lipa	0.7647	0.8235	6	New Rules – Dua Lipa	0.7857	0.8571
2	Nice for What - Drake	0.8462	0.9231	7	No Tears Left To Cry – Arianna Grande	0.7857	0.8571
3	Shape of You – Ed Sheeran	1.0	1.0	8	Delicate – Taylor Swift	1.0	1.0
4	Havana - Camila	0.7857	0.7857	9	Symphony – Zara Larson	1.0	1.0
5	Back To You - Selena Gomez	1.0	1.0	10	Hello - Adele	1.0	1.0

Table 5.12: The distance calculations between CAPMusic & Spotify for specific songs

From the above table, it is clear that there is a big different between CAPMusic (without and with mood) and Spotify, as the distance between Spotify and CAPMusic is 1.0 such as in (Back To You - Selena Gomez) and (Delicate – Taylor Swift) etc., which means there are no common songs. The diversity of recommendations was calculated for the above results and yielded values of 0.8968 (without mood) and 0.92465 (with mood).

2- Step 1 was repeated, but this time based on the assumption that the user has five songs in their listening history; the table below shows a comparison between the CAPMusic and Spotify recommendations.

No.	Song Listened To	CAPMusic Rec. (without mood)	CAPMusic Rec. (with mood)	Spotify Rec.
1	Honeymoon Avenue – Ariana Grande	Love Story - Taylor Swift	The Climb - Miley Cyrus	Yours Truly - Ariana Grande
		Fearless - Taylor Swift	Blown Away - Carrie Underwood	Piano - Ariana Grande
2	You Belong With Me – Taylor Swift	The Climb - Miley Cyrus	Love Story - Taylor Swift	Camila - Camila Cabello
		Blown Away - Carrie Underwood	Fearless - Taylor Swift	In the Dark - Camila Cabello
3	Shut Up and Drive – Rihanna	Breakin' - Rihanna	Paparazzi - Lady Gaga	All Your Fault - Bebe Rexha
		Toxic - Britney Spears	Breakin' - Rihanna	Big Sean - Nick Jonas
4	Poker Face – Lady Gaga	Sweet Dreams, Artist - Beyonce	Toxic - Britney Spears	Picture to Burn - Taylor Swift
5	Whenever, Wherever - Shakira	Gimme More - Britney Spears	Sweet Dreams, Artist - Beyonce	Last Year Was Complicated - Nick Jonas
		Paparazzi - Lady Gaga	Gimme More - Britney Spears	Breakin' - Rihanna

		Bad Romance - Lady Gaga	Bad Romance - Lady Gaga	All Of You - Colbie Caillat

Table 5.13: The common songs between CAPMusic & Spotify for no. of songs

The distance between the Spotify and CAPMusic (without and with mood) recommendations is 0.95. This experiment was repeated ten times with ten different songs, and the distance was calculated each time; the results are shown in the table below.

No.	Distance between Spotify and		No.	Distance between Spotify and	
	CAPMusic without mood	CAPMusic with mood		CAPMusic without mood	CAPMuisc with mood
1	0.8667	0.8235	6	0.6667	0.8571
2	0.6667	0.5833	7	0.8462	0.9231
3	1.0	0.9231	8	0.8	0.8
4	1.0	1.0	9	0.5833	1.0
5	0.5833	0.5833	10	0.7857	1.0

Table 5.14: The distance calculations between CAPMusic & Spotify

The diversity of recommendations was calculated for the above results, yielding values of 0.77986 (without mood) and 0.84934 (with mood).

5.5. Summary

In this chapter, the results obtained from CAPMusic have been experimentally evaluated by real users. The experiments consisted of two parts: evaluation metrics that measure the performance of CAPMusic in recommending songs; and benchmarking, which involves comparing the CAPMusic recommendations with very famous music recommendation systems in the music streaming services market (Last.fm and Spotify). In the evaluation metrics section, a number of measurements were calculated relating to the accuracy of estimated rankings using Spearman's ranking correlation, the degree of list relevance using mean average precision, list accuracy based on ranking position using mean reciprocal ranking, and recovery and novelty measurements using their respective formulas. In the benchmarking, the powerful features of CAPMusic have been illustrated by comparing its advantages with (Last.fm and Spotify) when the user shares the same listening histories with these recommendation systems. However, the same evaluation metrics applied to CAPMusic were also applied to Spotify and Last.fm.

The results of both stages of evaluation show that CAPMusic is an effective music recommender system, and leads to an improvement in recommendation accuracy (the relevance and the novelty) and user satisfaction.

Chapter 6

Conclusions and Future Work

6.1. Conclusions

In this thesis, we have presented a novel solution to address the Cold-Start problem encountered in Music Recommendation Systems (MRSs). The solution proposed in this study is able to offer both new and existing users an easy way to discover new music and provide personalised recommendations, i.e., recommending songs fit with the users' preferences. We have developed Context-Aware Personalised Music (CAPM) framework which is a generic framework for supporting the development of context-aware music recommendation systems. The framework includes the following key components: "data collection" to extract the user-context useful information from multiple data sources in order to use this information to build a dynamic user profile; "data processing" to analyse the collected data step by step preparing the data for the recommendation engine where the recommendation algorithms are implemented. In this framework, user-based hybrid recommendation algorithms were implemented which integrate users' social information in order not only to be able to recommend new songs to the new users who don't have any listening histories but also to be able recommend new songs to the existing users.

In order to evaluate the proposed framework and the recommendation algorithms, we developed the CAPMusic mobile application based on CAPM framework. The results show that users tend to agree with the recommended songs using our proposed approach. We intend to collect the user feedback and further explore the effects of such matching on users' preferences. Our experimental results show that CAPMusic can provide relevant music recommendations based on individual preferences; the results showed that overall user satisfaction was 97% and 91% with and without mood selection, respectively. This indicates that our approach is able to offer improved recommendation accuracy and, consequently, user satisfaction.

The main conclusion of this thesis is that Content-Based, Collaborative Filtering and context-based techniques are complementary ones. Content-Based techniques help to solve problems when the users' preferences data is inadequate or non-existent and Collaborative Filtering techniques can obtain relations between users who share same listening histories and the context-based techniques can use the user context to keep the recommendation list updated and adaptive with the user status.

CAPM is able to obtain better results using hybrid recommendation methods instead of pure ones. This conclusion is a direct observation of the results obtained from our experiments and answers our main questions.

6.2. Future Work

In future work, the relationship between individual characteristics, and user perception and interaction with ratings of recommended tracks, will be explored in more detail. In addition, the way and extent to which different levels of interaction affect performance and cognitive load in information filtering tasks will also be investigated.

For this research, the total music sample was only 910 songs, which is not considered a large data set; however, the success at this stage provides motivation for the future development of this research, in a more comprehensive study with a larger music sample. In addition, future research might aim to make further improvements, including:

- System performance: for the present study, the music recommendation library was manually sampled; the system will be more effective if an auto-sampling module is used.
- Examining combinations of personalities, and the various categories of music that these personalities might prefer to consume.
- Using Fuzzy logic method which has been extensively used in the design of a recommender system to handle the uncertainty, impreciseness and vagueness in item features and users' behaviour especially with the user current mood.
- Increasing the flexibility of setting up evaluations by adding more metrics to the result overview and developing further connectors for social networks and other web services to enrich the user's context.

References

Adomavicius, G. & Tuzhilin, A., 2005. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6), pp.734–749.

Agarwal, D. & Chen, B.-C., 2009. Regression-based latent factor models. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 09*.

Aggarwal, C.C., 2016. Content-Based Recommender Systems. *Recommender Systems*, pp.139–166.

Aggarwal, C.C., 2016. Ensemble-Based and Hybrid Recommender Systems. *Recommender Systems*, pp.199–224.

Anderson, C.A. et al., 2010. Violent video game effects on aggression, empathy, and prosocial behavior in Eastern and Western countries: A meta-analytic review. *Psychological Bulletin*, 136(2), pp.151–173.

Auer, S.C.B. et al., 2007. DBpedia: A Nucleus for a Web of Open Data. *The Semantic Web Lecture Notes in Computer Science*, pp.722–735.

Baeza-Yates, R., & Ribeiro-Neto, B. (2011). *Modern information retrieval: the concepts and technology behind search*. Harlow, England: Pearson.

Bedekar, M.V., Deshpande, B. & Joshi, R., 2008. Web Search Personalization by User Profiling. *2008 First International Conference on Emerging Trends in Engineering and Technology*.

Bhat, A.S. et al., 2014. An Efficient Classification Algorithm for Music Mood Detection in Western and Hindi Music Using Audio Feature Extraction. *2014 Fifth International Conference on Signal and Image Processing*.

Boucher-Ryan, P.D. & Bridge, D., 2006. Collaborative Recommending using Formal Concept Analysis. *Knowledge-Based Systems*, 19(5), pp.309–315.

Bouras, C. & Tsogkas, V., 2012. Clustering to Deal with the New User Problem. *2012 IEEE 15th International Conference on Computational Science and Engineering*, pp.58–65.

Bradley, K., Rafter, R. & Smyth, B., 2000. Case-Based User Profiling for Content Personalisation. *Lecture Notes in Computer Science Adaptive Hypermedia and Adaptive Web-Based Systems*, pp.62–72.

Braynov, S., 2004. Personalization and Customization Technologies. *The Internet Encyclopedia*.

Burke, R., 2002. Hybrid Web Recommender Systems. *The Adaptive Web Lecture Notes in Computer Science*, pp.377–408.

Burke, R., 2007. Hybrid Web Recommender Systems. *Springer Berlin Heidelberg, Berlin, Heidelberg*, 377–408. DOI:h p://dx.doi.org/10.1007/978-3-540-72079-912

Cantador, I.C.A. et al., 2015. Cross-Domain Recommender Systems. *Recommender Systems Handbook*, pp.919–959.

Castellano, G. et al., 2007. Similarity-Based Fuzzy Clustering for User Profiling. *2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops*.

Celma, O., 2010. The Long Tail in Recommender Systems. *Music Recommendation and Discovery*, pp.87–107.

Chen, T. & He, L., 2009. Collaborative Filtering Based on Demographic Attribute Vector. *2009 ETP International Conference on Future Computer and Communication*.

Chowdhury, G. G. (2017). *Introduction to modern information retrieval*. S.l.: facet publishing.

Crane, M., 2011. The new user problem in collaborative filtering. *Thesis for the degree of Master of Science, Department of Computer Science. University of Otago, Dunedin, New Zealand.*

Cremonesi, P. & Quadrana, M., 2014. Cross-domain recommendations without overlapping data. *Proceedings of the 8th ACM Conference on Recommender systems - RecSys 14*.

Cremonesi, P., Tripodi, A. & Turrin, R., 2011. Cross-Domain Recommender Systems. *2011 IEEE 11th International Conference on Data Mining Workshops*.

Damljanovic, D., Stankovic, M. & Laublet, P., 2012. Linked Data-Based Concept Recommendation: Comparison of Different Methods in Open Innovation Scenario. *Lecture Notes in Computer Science The Semantic Web: Research and Applications*, pp.24–38.

Delannay, N. & Verleysen, M., 2008. Collaborative filtering with interlaced generalized linear models. *Neurocomputing*, 71(7-9), pp.1300–1310.

Duan, M., 2015. Collaborative Filtering Recommendation Algorithm.

Duzen, Z. & Aktas, M.S., 2016. An approach to hybrid personalized recommender systems. *2016 International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*.

Eerola, T., 2012. Modeling Listeners' Emotional Response to Music. *Topics in Cognitive Science*, 4(4), pp.607–624.

Elahi, M., 2011. Adaptive Active Learning in Recommender Systems. *User Modeling, Adaption and Personalization Lecture Notes in Computer Science*, pp.414–417.

Elahi, M., Ricci, F. & Rubens, N., 2013. Active learning strategies for rating elicitation in collaborative filtering. *ACM Transactions on Intelligent Systems and Technology*, 5(1), pp.1–33.

Elahi, M., Ricci, F. & Rubens, N., 2016. A survey of active learning in collaborative filtering recommender systems. *Computer Science Review*, 20, pp.29–50.

Elbadrawy, A. & Karypis, G., 2015. User-Specific Feature-Based Similarity Models for Top-n Recommendation of New Items. *ACM Transactions on Intelligent Systems and Technology*, 6(3), pp.1–20.

Festinger, L., 1957. *A theory of cognitive dissonance*, Stanford University.

Figueroa, C. et al., 2015. A systematic literature review of Linked Data-based recommender systems. *Concurrency and Computation: Practice and Experience*, 27(17), pp.4659–4684.

Fortes, R.S., Freitas, A.R.R.D. & Gonçalves, M.A., 2017. A Multicriteria Evaluation of Hybrid Recommender Systems: On the Usefulness of Input Data Characteristics. *Proceedings of the 19th International Conference on Enterprise Information Systems*.

Goldberg, D. (1992). *Using collaborative filtering to weave an information tapestry*. Palo Alto, CA: Xerox Corp., Palo Alto Research Center.

Gunawardana, A. & Shani, G., 2015. Evaluating Recommender Systems. *Recommender Systems Handbook*, pp.265–308.

Guy, I., 2015. Social Recommender Systems. *Recommender Systems Handbook*, pp.511–543.

Heitmann, B., Kinsella, S., Hayes, C., & Decker, S., 2008. Implementing Semantic Web applications: reference architecture and challenges. *In Proceedings of the 5th Workshop on Semantic Web Enabled Software Engineering, volume 524 of CEUR Workshop Proceedings. CEUR-ws.org.*

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems, 22*(1), 5-53. doi:10.1145/963770.963772

Hernández-Rubio, M., Cantador, I., Bellogín, A.: A comparative analysis of recommender systems based on item aspect opinions extracted from user reviews. User Model. User-Adapt. Interact. (2018)

Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. In: 4th International Conference on Learning Representations (ICLR) (2016)

Hijikata, Y., 2004. Implicit user profiling for on demand relevance feedback. *Proceedings of the 9th international conference on intelligent user interface - IUI 04*.

Holbrook, M.B. & Gardner, M.P., 2000. Illustrating a dynamic model of the mood-updating process in consumer behavior. *Psychology and Marketing*, 17(3), p.165.

Holte, R. C., & Yan, J. N. (1996). Inferring what a user is not interested in. *Lecture Notes in Computer Science Advances in Artificial Intelligence,* 159-171 doi: 10.1007/3-540-61291-2_49

Hu, X., Downie, S. J. & Ehmann, A. F. (2009). Lyric text mining in music mood classification. *10th International Society for Music Information Retrieval Conference, ISMIR,* 411-16

Jannach, D., Adomavicius, G.: Recommendations with a purpose. In: Proceedings of the 10th ACM Conference on Recommender Systems (RecSys 2016), pp. 7–10. Boston, Massachusetts, USA (2016)

Kaminskas, M. & Ricci, F., 2012. Contextual music information retrieval and recommendation: State of the art and challenges. *Computer Science Review*, 6(2-3), pp.89–119.

Kim et al., 2010. Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation. *Electronic Commerce Research and Applications*, 9(1), pp.73–83.

Kim et al., 2010. Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation. *Electronic Commerce Research and Applications*, 9(1), pp.73–83.

Klyne, G., Carroll, J., 2004. Resource Description Framework (RDF): Concepts and Abstract Syntax - W3C Recommendation. Retrieved Feb 14, 2018, http://www.w3.org/TR/rdfconcepts/

Knees, P. & Schedl, M., 2013. Music similarity and retrieval. *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval - SIGIR 13*.

Knees, P. & Schedl, M., 2016. Introduction to Music Similarity and Retrieval. *Music Similarity and Retrieval The Information Retrieval Series*, pp.1–30.

Knobloch-Westerwick, S., 2006. Mood Management: Theory, Evidence, and Advancements. *In: Bryant, J., Vorderer, P. (eds.) Psychology of Entertainment,* pp. 239–254

Kobilarov, G., et al., 2009. Media Meets Semantic Web - How the BBC Uses DBpedia and Linked Data to Make Conections. *Proceedings of the 6th European Semantic Web Conference (ESWC2009).*

Kohrs, A. & Merialdo, B., 2001. Improving collaborative filtering for new users by smart object selection. *In Proceedings of the International Conference on Media Features (ICMF)*.

Langseth, H. & Nielsen, T.D., 2015. Scalable learning of probabilistic latent models for collaborative filtering. *Decision Support Systems*, 74, pp.1–11.

Leung, C.W.-K., Chan, S.C.-F. & Chung, F.-L., 2008. An empirical study of a cross-level association rule mining approach to cold-start recommendations. *Knowledge-Based Systems*, 21(7), pp.515–529.

Levene, M., 2010. *An introduction to search engines and web navigation*, Hoboken, NJ: John Wiley.

Li, J., Ritter, A. & Hovy, E., 2014. Weakly Supervised User Profile Extraction from Twitter. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Li, Q. & Kim, B.M., 2003. Clustering approach for hybrid recommender system. *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*.

Liang, D., 2016. Collaborative Filtering Recommendation Algorithm.

Loh, S. et al., 2009. Identifying Similar Users By Their Scientific Publications To Reduce Cold Start In Recommender Systems. *Proceedings of the Fifth International Conference on Web Information Systems and Technologies*.

Mahata, A., Saini, N., Saharawat, S., & Tiwari, R. (2017). Intelligent Movie Recommender System Using Machine Learning. *Intelligent Human Computer Interaction Lecture Notes in Computer Science,* 94-110. doi:10.1007/978-3-319-52503-7_8

Maneeroj, S. & Takasu, A., 2009. Hybrid Recommender System Using Latent Features. *2009 International Conference on Advanced Information Networking and Applications Workshops*.

Martinez, L., Perez, L.G. & Barranco, M.J., 2009. Incomplete preference relations to smooth out the cold-start in collaborative Recommender Systems. *NAFIPS 2009 - 2009 Annual Meeting of the North American Fuzzy Information Processing Society*.

Masthoff, J., 2010. Group Recommender Systems: Combining Individual Models. *Recommender Systems Handbook*, pp.677–702.

McAuley, J., Targett, C., Shi, Q., Van Den Hengel, A.: Image-based recommendations on styles and substitutes. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 43–52 (2015)

McAuley, J., Targett, C., Shi, Q., Van Den Hengel, A.: Image-based recommendations on styles and substitutes. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 43–52 (2015)

Messina, P., Dominguez, V., Parra, D., Trattner, C., Soto, A.: Content-based artwork recommendation: integrating painting metadata with neural and manually-engineered visual features. User Model. User-Adapt. Interact. (2018)

Middleton, S.E., Shadbolt, N.R. & Roure, D.C.D., 2004. Ontological user profiling in recommender systems. *ACM Transactions on Information Systems*, 22(1), pp.54–88.

Mizgajski, J., Morzy, M.: Affective recommender systems in online news industry: how emotions influence reading choices. User Model. User-Adapt. Interact. (2018)

Nasraoui, O. et al., 2008. 10 years of knowledge discovery on the web post-workshop report. *ACM SIGKDD Explorations Newsletter*, 10(2), pp.78–83.

Negre, E., 2015. Recommender Systems. *Information and Recommender Systems*, pp.7–27.

Noia, T.D. et al., 2012. Linked open data to support content-based recommender systems. *Proceedings of the 8th International Conference on Semantic Systems - I-SEMANTICS 12*.

Oliver, M.B., 2008. Tender Affective States as Predictors of Entertainment Preference. *Journal of Communication*, 58(1), pp.40–61.

Oramas, S., Ostuni, V.C., Noia, T.D., Serra, X., Sciascio, E.D.: Sound and music recommendation with knowledge graphs. ACM Trans. Intell. Syst. Technol. (TIST) **8**(2), 21 (2017)

Ostuni, V.C. et al., 2013. Top-N recommendations from implicit feedback leveraging linked open data. *Proceedings of the 7th ACM conference on Recommender systems - RecSys 13*.

Ouanaim, M. et al., 2010. Dynamic user profiling approach for services discovery in mobile environments. *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference on ZZZ - IWCMC 10*.

Park, S.-T. & Chu, W., 2009. Pairwise preference regression for cold-start recommendation. *Proceedings of the third ACM conference on Recommender systems - RecSys 09*.

Park, S.-T. et al., 2006. Naïve filterbots for robust cold-start recommendations. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 06*.

Park, Y.-J. & Tuzhilin, A., 2008. The long tail of recommender systems and how to leverage it. *Proceedings of the 2008 ACM conference on Recommender systems - RecSys 08*.

Passant, A., 2010. Measuring Semantic Distance on Linking Data and Using it for Resources Recommendations. *AAAI Spring Symposium: Linked Data Meets Artificial Intelligence.*

Pazos-Arias, J. J., Vilas, A. F., & Rebeca, D. R. (2012). *Recommender systems for the social web*. Heidelberg: Springer.

Piao, G. & Breslin, J.G., 2016. Measuring semantic distance for linked open data-enabled recommender systems. *Proceedings of the 31st Annual ACM Symposium on Applied Computing - SAC 16*.

Poo, D., Chng, B. & Goh, J.-M., 2003. A hybrid approach for user profiling. *36th Annual Hawaii International Conference on System Sciences, 2003. Proceedings of the*.

Rao, K.N. & Talwar, V., 2011. Content-Based Document Recommender System for Aerospace Grey Literature: System Design. *DESIDOC Journal of Library & Information Technology*, 31(3), pp.189–201.

Rashid, A.M. et al., 2002. Getting to know you: learning new user preferences in recommender systems. *In Proceedings of the 7th International Conference on Intelligent User Interfaces, (2002) San Francisco, California, USA.*, pp.127–134.

Rashid, A.M., Karypis, G. & Riedl, J., 2008. Learning preferences of new users in recommender systems: an information theoretic approach. *ACM SIGKDD Explorations Newsletter*, 10(2), p.90.

Rashid, A.M., Karypis, G. & Riedl, J., 2008. Learning preferences of new users in recommender systems. *ACM SIGKDD Explorations Newsletter*, 10(2), p.90.

Rubens, N. et al., 2015. Active Learning in Recommender Systems. *Recommender Systems Handbook*, pp.809–846.

Ryan, R.M., Rigby, C.S. & Przybylski, A., 2006. The Motivational Pull of Video Games: A Self-Determination Theory Approach. *Motivation and Emotion*, 30(4), pp.344–360.

Said, A., 2010. Identifying and utilizing contextual data in hybrid recommender systems. *Proceedings of the fourth ACM conference on Recommender systems - RecSys 10*.

Schaefer, A., Nils, F., Sanchez, X., & Philippot, P., 2005: A Multi-criteria Assessment of Emotional Films (*unpublished manuscript*)

Schafer, J.B., Frankowski, D., Herlocker, J. & Sen, S., 2007. Collaborative filltering recommender systems, in: P. Brusilovsky, A. Kobsa, W. Nejdl (Eds.), *The Adaptive Web*, pp. 291–324

Schedl, M. et al., 2015. Music Recommender Systems. *Recommender Systems Handbook*, pp.453–492.

Schedl, M., Flexer, A. & Urbano, J., 2013. The neglected user in music information retrieval research. *Journal of Intelligent Information Systems*, 41(3), pp.523–539.

Schein, A. I., Popescul, A., Ungar, L. H., & Pennock, D. M. (2002). Methods and metrics for cold-start recommendations. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR 02*. doi:10.1145/564376.564421

Schein, A.I. et al., 2002. Methods and metrics for cold-start recommendations. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR 02*.

Schein, A.I. et al., 2002. Methods and metrics for cold-start recommendations. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR 02*.

Schmachtenberg, M., Bizer, C. & Paulheim, H., 2014. Adoption of the Linked Data Best Practices in Different Topical Domains. *The Semantic Web – ISWC 2014 Lecture Notes in Computer Science*, pp.245–260.

Stern, D.H., Herbrich, R. & Graepel, T., 2009. Matchbox. *Proceedings of the 18th international conference on World wide web - WWW 09*.

Tang, J. et al., 2010. A Combination Approach to Web User Profiling. *ACM Transactions on Knowledge Discovery from Data*, 5(1), pp.1–44.

Tesser, A., Millar, K., Wu, C.H., 1998. On the Perceived Functions of Movies. *J. Psychology* 122, 441–449

Tkalcic, M. & Chen, L., 2015. Personality and Recommender Systems. *Recommender Systems Handbook*, pp.715–739.

Vasile, F., Smirnova, E., Conneau, A.: Meta-prod2vec: Product embeddings using side-information for recommendation. In: Proceedings of the 10th ACM Conference on Recommender Systems, pp. 225–232 (2016)

Vekariya, V., & Kulkarni, G. R. (2012). Hybrid recommender systems: Survey and experiments. *2012 Second International Conference on Digital Information and Communication Technology and its Applications (DICTAP)*. doi:10.1109/dictap.2012.6215409

Wang, C. & Blei, D.M., 2011. Collaborative topic modeling for recommending scientific articles. *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 11*.

Waterman, A.S., 1993. Two Conceptions of Happiness: Contrasts of Personal Expressiveness (eudaimonia) and Hedonic Enjoyment. *J. Personality and Social Psychology* 64, 678–691

Weng, L.-T. et al., 2008. Exploiting Item Taxonomy for Solving Cold-Start Problem in Recommendation Making. *2008 20th IEEE International Conference on Tools with Artificial Intelligence*.

Zhao, H. et al., 2017. Collaborative Filtering with Social Local Models. *2017 IEEE International Conference on Data Mining (ICDM)*.

Zhong, J. et al., 2002. Conceptual Graph Matching for Semantic Search. *Conceptual Structures: Integration and Interfaces Lecture Notes in Computer Science*, pp.92–106.

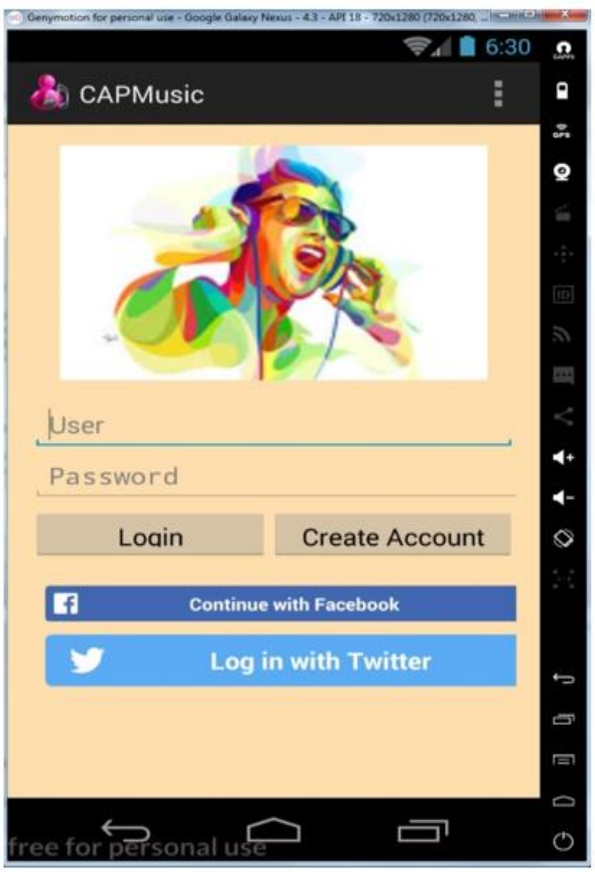
Zhou, T. & Li, H., 2010. User context based collaborative filtering recommendation. *Journal of Computer Applications*, 30(4), pp.1076–1078.

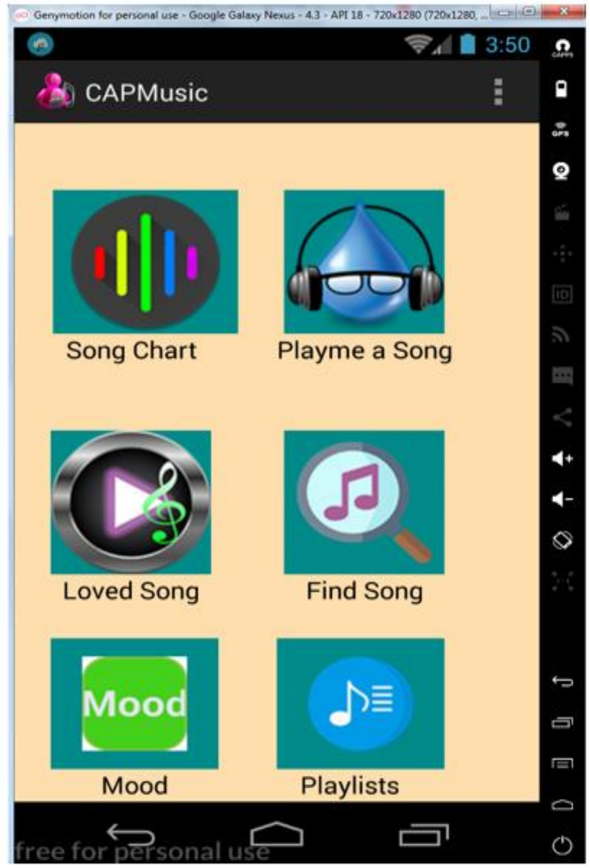
Zillmann, D., 1988. Mood Management: Using Entertainment to Full Advantage. *In: Donohew, L., Sypher, H.E., Higgins, E.T. (eds.) Communication, Social Cognition, and Affect*, pp. 147–171

Zillmann, D., 2000. Mood Management in the Context of Selective Exposure Theory. *Annals of the International Communication Association*, 23(1), pp.103–123.

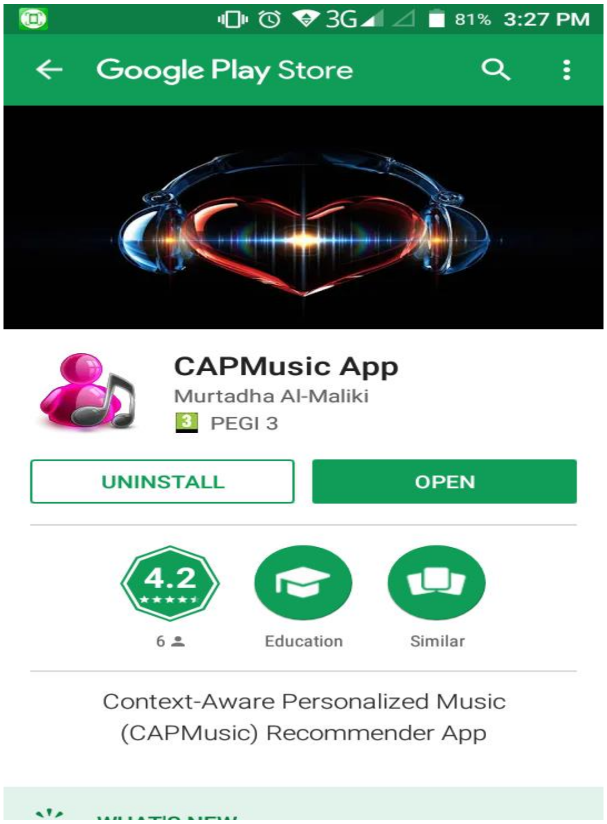
Appendix A

CAPMusic user interface





CAPMusic as look in google play



Appendix B – Questionnaire A

Thank you for taking part in this activity. We would be very grateful if you would fill in this questionnaire to evaluate a music recommender Apps.

CAPM App.

1- No. of attempts of recommendations (without mood):

1	2	3	4	5

2- Please give a rate to the songs for each attempt and arrange the songs according to your preferences and select the new song with letter "N"

Attempt 1									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 2									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4

	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 3									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 3									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 4									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3

	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 5									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1

3- No. of attempts of recommendations (with mood):

1	2	3	4	5

4- Please give a rate to the songs for each attempt and select the new song with letter “N”

Attempt 1									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**

Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 2									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 3									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 3									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5

	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 4									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 5									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2

	1		1		1		1		1

5- Please give a rate the App interface

1	2	3	4	5

6- Please feel free to write a comment about the music App

Last.fm App.

1- No. of attempts of recommendations

1	2	3	4	5

2- Please give a rate to the songs for each attempt and arrange the songs according to your preferences and select the new song with letter "N"

Attempt 1									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 2									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**

Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 3									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 3									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3

	2		2		2		2		2
	1		1		1		1		1
Attempt 4									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 5									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1

3- Please give a rate the App interface

1	2	3	4	5

4- Please feel free to write a comment about the music App

Spotify App.

1- No. of attempts of recommendations

1	2	3	4	5

2- Please give a rate to the songs for each attempt and arrange the songs according to your preferences and select the new song with letter "N"

Attempt 1									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 2									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 3									

Song No.	Rate	Song No.	Rate	Song No.	Rate	Song No.	Rate	Song No.	Rate
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 3									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 4									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4

	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Attempt 5									
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 1	5	Song 2	5	Song 3	5	Song 4	5	Song 5	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1
Song No.	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**	**Song No.**	**Rate**
Song 6	5	Song 7	5	Song 8	5	Song 9	5	Song 10	5
	4		4		4		4		4
	3		3		3		3		3
	2		2		2		2		2
	1		1		1		1		1

3- Please give a rate the App interface

1	2	3	4	5

4- Please feel free to write a comment about the music App

Appendix C – Questionnaire B: Music/Songs Mood Preferences

I am a PhD student at the University of Portsmouth, researching on music recommender system and part of my research is combining the user moods in my music recommender system (mobile app) to give users suitable music/songs according to their preferences in that specific mood.

There are 18 mood categories (Hu, 2009): calm, sad, happy, romantic, etc. For each mood category, there are a number of songs suitable for it (e.g. calm songs for mood calm, happy songs for mood happy and so on).

Zillmann (2002) proposed a mood management theory which said that a person in any negative mood can enhance his/her mood and be in a positive mood by listening to joyful music/songs, and based on this theory, I hereby design this questionnaire to know the your preferences in each mood category, as a response.

This is an anonymous questionnaire you will not be able to be identified from the information you provided, hence it will be confidentially and officially treated, without public consumption.

Please tick as appropriate.

Thanks for your time.

1- Gender: Male □ Female □
2- Age: 20-34 □ 35-49 □ 50-64 □ 65-74 □
3- What kind of music/songs do you prefer listening to when your mood is calm (please tick all that apply):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

4- What kind of music/songs do you prefer listening to when your mood is sad (please tick all that apply):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

5- What kind of music/songs do you prefer listening to when your mood is happy (please tick all that apply):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

6- What kind of music/songs do you prefer listening to when your mood is romantic (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

7- What kind of music/songs do you prefer listening to when your mood is gleeful (please tick all that apply and number them by priority):

calm songs □	depressed songs □	brooding songs □

sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

8- What kind of music/songs do you prefer listening to when your mood is earnest (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

9- What kind of music/songs do you prefer listening to when your mood is depressed (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

10- What kind of music/songs do you prefer listening to when your mood is angry (please tick all that apply):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

11- What kind of music/songs do you prefer listening to when your mood is grief (please tick all that apply):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

12- What kind of music/songs do you prefer listening to when your mood is dreamy (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

13- What kind of music/songs do you prefer listening to when your mood is cheerful (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

14- What kind of music/songs do you prefer listening to when your mood is pessimism (please tick all that apply and number them by priority):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □

earnest songs □ pessimism songs □ excitement songs □

15- What kind of music/songs do you prefer listening to when your mood is brooding (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

16- What kind of music/songs do you prefer listening to when your mood is aggressive (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

17- What kind of music/songs do you prefer listening to when your mood is anxious (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

18- What kind of music/songs do you prefer listening to when your mood is confident (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □

gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

19- What kind of music/songs do you prefer listening to when your mood is hopeful (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

20- What kind of music/songs do you prefer listening to when your mood is excitement (please tick all that):

calm songs □	depressed songs □	brooding songs □
sad songs □	angry songs □	aggressive songs □
happy songs □	grief songs □	anxious songs □
romantic songs □	dreamy songs □	confident songs □
gleeful songs □	cheerful songs □	hopeful songs □
earnest songs □	pessimism songs □	excitement songs □

Appendix D – Form UPR16

FORM UPR16

Research Ethics Review Checklist

University of Portsmouth

Please include this completed form as an appendix to your thesis (see the Postgraduate Research Student Handbook for more information

Postgraduate Research Student (PGRS) Information		**Student ID:**	411895
PGRS Name:	Murtadha Sami Luaibi Al-Maliki		
Department:	School of Engineering	**First Supervisor:**	Dr. Linda Yang
Start Date: (or progression date for Prof Doc students)	01/10/2014		
Study Mode and Route:	Part-time ☐ Full-time ☒	MPhil ☐ PhD ☒	MD ☐ Professional Doctorate ☐

Title of Thesis:	User based Hybrid Algorithms for Music Recommendation Systems
Thesis Word Count: (excluding ancillary data)	39046

If you are unsure about any of the following, please contact the local representative on your Faculty Ethics Committee for advice. Please note that it is your responsibility to follow the University's Ethics Policy and any relevant University, academic or professional guidelines in the conduct of your study

Although the Ethics Committee may have given your study a favourable opinion, the final responsibility for the ethical conduct of this work lies with the researcher(s).

UKRIO Finished Research Checklist:
(If you would like to know more about the checklist, please see your Faculty or Departmental Ethics Committee rep or see the online version of the full checklist at: http://www.ukrio.org/what-we-do/code-of-practice-for-research/)

a) Have all of your research and findings been reported accurately, honestly and within a reasonable time frame?	YES ☒ NO ☐
b) Have all contributions to knowledge been acknowledged?	YES ☒ NO ☐
c) Have you complied with all agreements relating to intellectual property, publication and authorship?	YES ☒ NO ☐
d) Has your research data been retained in a secure and accessible form and will it remain so for the required duration?	YES ☒ NO ☐
e) Does your research comply with all legal, ethical, and contractual requirements?	YES ☒ NO ☐

Candidate Statement:

I have considered the ethical dimensions of the above named research project, and have successfully obtained the necessary ethical approval(s)

Ethical review number(s) from Faculty Ethics Committee (or from NRES/SCREC):	800D-760C-C5CC-E0CD-BC93-E138-F1B4-3338

If you have *not* submitted your work for ethical review, and/or you have answered 'No' to one or more of questions a) to e), please explain below why this is so:

Signed (PGRS): Murtadha Al-Maliki **Date:** 02/10/2018

UPR16 – August 2015