

“Improvise - Creating playlists according the user activity”

Ricardo Antunes Lopes da Cunha
ricardo.d.cunha@ist.utl.pt

Instituto Superior Técnico, Lisboa, Portugal

May 2014

Abstract

The number of songs available in Internet has grown steadily since its appearance. As a result of this growth, it is extremely difficult for users to find the appropriate music that suit their needs. To allow users to listen to the music they like, two techniques are quite used, searching and recommendation. While recommender systems rely more actively on the users needs, the search engines receive their needs more passively. In the present work we developed a solution for music recommendation, whose objective is to search and recommend songs in tune with the activity that the user is performing. With this objective in mind, we developed a hybrid approach reflecting the integration of a recommendation system based on context and content. Regarding context, we use information about the user’s activity. In what concerns the content, we took into account the information about the acoustic features of each song. The recommendation algorithm starts by recommending the same songs for all users, we call this the generic model. The user’s interaction with the system leads to the adjustment of the generic model, to its own musical preferences changing it into a personalized model. A personalized recommendation model for each user will then be obtained after some interactions with the system, which will better reflect their tastes.

Keywords: Music Recommendation System, Hybrid recommendation, Recommendation based on context, User activity, Association between music and activities, Generic and personalized model

1. Introduction

Nowadays there are numerous resources at our disposal for the consumption of digital music, whether through online stores or streaming services. Thus we have at our disposal thousands of songs that can be listen anywhere, anytime, and on multiple devices like MP3 players, smartphones and tablets.

However, with the access to such a vast collection of songs, the main problem arises when the user wants to choose songs he wants to listen. This problem is even worse when trying to create a playlist to listen to while performing a certain task, such as practicing sports, driving or studying. So creating playlists becomes a tiresome and frustrating process to users.

To solve this problem in the past years several proposals have been developed for the generation and creation of playlists. In this work we describe and analyze the main techniques used in these approaches, which include the use of recommendation methods like collaborative filtering, content-based, context-based and hybrid approaches.

Despite the good results that such techniques can achieve, they cannot be adapted to the specific activity the user is performing. In fact, this topic has not yet been well explored, therefore an effective so-

lution for this type of recommendation is missing.

With this problem in mind we aim to design a solution for playlist recommendation based on the activity the user is performing (context). In addition to the activity performed by the user our goal is to use information about the songs (content) to help in the recommendation process, to detect a possible parallelism between music features and activities.

Our solution is based on a hybrid recommendation system that considers the context in which the music is being listen to and the musical features of each song.

Initially the same set of songs is recommended to all users. This we call the generic model where each song is recommended taking into account some of the tests performed with real users in order to realize the songs predominately preferred for each activity they perform. The songs are recommended taking into account a set of intervals that define how music should be recommended for each activity. These intervals represent the musical features of each song. With the interaction of the user’s with the generic model, this is adapted to a personalized model that takes in account the previously choices made by the user’s.

In the next section we discuss the related work,

presenting research made in music recommender systems. Section 3 describes our solution for recommendation. Section 4 describes the tests made with users and in section 5 we present the results and findings of our work. Finally in section 6 we conclude our work and provide future direction.

2. Related Work

Given the enormous amount of songs that exist nowadays and the continuous growth of internet and the power of mobile devices, managing and searching songs has become more difficult. In this section we present the state-of-art approaches in music recommendations.

2.1. Music Recommendation Methods

2.1.1. Collaborative filtering

Collaborative filters are the most common approaches in recommender systems. This technique is based on user-generated content (eg: ratings or implicit feedback) where items are recommended to a user if they are appreciated by similar users.

In the music domain, collaborative filters works by constructing a matrix M with n songs and m users, which contains the user interaction with the items (eg ratings, number of views, etc.). Each line represents a profile of a user and the columns represent songs. The value M_{u_a, i_j} is the rating that the user u_a assigned to the item i_j .

Although this method is one of the most used for music recommendation, it presents some problems like the early-rater and cold-start problems[1].

2.1.2. Content-Based

Recommendations based on music content have been used considerably less than the collaborative filtering method. This is probably due to the fact that the techniques based on content requiring knowledge about the data, and music is notoriously difficult to describe and classify.

In a content-based recommendation system, information describing songs is gathered; this information relies on music features such as genre[2], loudness, energy, rhythm or melody. This approach is not based on assessments made by users, but on the features of the songs.

Because this type of recommenders is focused on finding similarities between the various characteristics of the songs using only own music features like rhythm, energy and genre, the user's personal opinion is not been taken into account when the recommendation is being made.

2.1.3. Context-Based

In the field of recommender systems, the context refers to the user situation such as time, mood or activity that he is performing while he is searching for a recommendation[3].

Advanced technologies such as wearable computers and smart devices, have created the opportunity for researchers to collect and use data about the context to enrich the interaction between users and computers. Situational information as time, emotional status, presence of others, past and future events can help the system to better understand the current needs of the user.

The positive aspect of these systems is that the users needs may vary depending on the context, and these recommendation systems can be adapted to provide the most relevant services.

2.1.4. Demographic filtering

Demographic filtering is one of the most basic methods for identifying the type of users that like a certain item[4].

This technique classifies users profiles into groups according to some personal data (age, marital status, gender, etc.), geographic data (city, country) and psychographics (interests, lifestyle, etc.). The main drawback of this method resides in the fact that the recommendation is too generic since all users with the same demographic profile will receive the same recommendations, which is incorrect since people of same gender or age can have very different tastes.

2.1.5. Hybrid approaches

The major problems of collaborative filters and content based approaches rely on new users or items and in modeling the user preferences, respectively. The main purpose of a hybrid method is to achieve better recommendations combining some of the techniques described earlier.

According to Burke[5] there are several methods to integrate different approaches into a hybrid recommender where we emphasize the weighted, switching, mixed and cascade methods.

Hybrid systems perform better than other approaches presented, since they can alleviate some of the drawbacks that suffer a single technique integrating multiple approaches.

2.2. Music recommendation works

2.2.1. Context-Aware Mobile Music Recommendation for Daily Activities

This work presents a novel approach to employ contextual information collected with mobile devices for satisfying user's short-term music playing needs[6].

This work presents a novel approach to employ contextual information collected with mobile devices for satisfying users short-term music playing needs. The presented idea is to create a music system that is able to detect automatically and in real time, which activities the user is performing and playing music appropriate to these activities. The activities that are automatically detected by this

work are walking, working, running, relaxing, sleeping and shopping.

For each one, the sensor data includes time of day, accelerometer data, and audio from a microphone. The recommendation problem is formulated as a two-step process: (1) infer the users current context category using the features collected by the mobile device, and (2) find a song matching the context that was inferred. The first step is called context inference and the second step music content analysis.

2.2.2. Lifetrak: Music In Tune With Your Life

The essential idea behind Lifetrak[7] is that several sensing modalities influence the music that is played. Essentially, Lifetrak enables a context-aware playlist that automatically chooses music in real-time based upon the location, the pace of movement, the current time, and various other phenomena in the users environment. Furthermore, Lifetrak uses a simple learning mechanism to adjust the ratings of songs for a particular context based on users feedback when a song is being played.

The system does not autonomously analyze songs and connect them to corresponding contexts. Instead, Lifetrak relies on the user to tag the song database with the basic context constructs. Lifetrak then figures out the users current context and ranks the song database according to this information.

There are basically four main components involved in the software architecture for Lifetrak. The first component is the user space document which contains a list of all songs and the contexts that apply for each song. Then there is the context engine that actually gets information from various sensing modalities and categorizes them into specific tags. There is also the rating generator which takes the current context and the user defined song database, which contains the individual context tags for each song, and then generates a ranked playlist. Finally, the music player provides an interface for the end user.

2.2.3. Nextone Player: A Music Recommendation System Based On User Behavior

In this paper the authors designed a system that allows recommending songs that are favored by the user, new to his ear and that correspond to his preferences[8]. To this end the authors use user's behavior as feedback to decide what song should be played at that particular moment.

The authors determine whether a song is to be recommended as the next one in the playlist from five perspectives: genre, year, favor, freshness and time pattern. Time series analysis of genre and year are used to predict these attributes to the next song rather than to select the song with similar genre

and year to the current song. According to the authors a similar song relatively to a current one cannot be assumed as a reasonable good choice of recommendation. Prediction using time series analysis caters better to a users likes. The number of times a song has been actively played and how many times the same has been completely played are used to infer the strength of songs favor. Other feature introduced by the authors was to consider the musics freshness in relation to a user. The freshness of a song may be considered as the strength of strangeness or the amount of experience forgotten. In a different period of a day or a week, users tend to select different styles of songs. For example, in the afternoon, a user may like a soothing kind of music for relaxation and may switch to energetic songs in the evening. This paper uses a Gaussian Mixture Model to represent the time pattern of listening and compute the probability of playing a song at that time.

At the time of the recommendation, the obtained results from the different perspectives presented above are integrated into a final result. Given that different users have different tastes, different weights are attributed to the five factors at the time of integration of results. This system has interesting techniques such as the concept of freshness and the use of users interaction with the system to find out the year and gender of the next song to play. However this approach does not take into account the users context which we consider relevant to a more effective recommendation.

2.2.4. Foxtrot: A Soundtrack for Where You Are

In this article[9], the authors propose a mobile application named Foxtrot, that allows people to share music they associate to a given location. To provide an engaging experience, but at the same discrete time, the Foxtrot mixes tags assigned by users, the geographical location and music, to create a kind of radio channel where people can hear music associated with a given location.

The Foxtrot revolves around two primary objects, users, and songs. Users have their current location, speed, hearing profile, history, and friends list. Each song focuses on a geographical location, distance visibility, a type and a set of tags specified by users. The presented approach to create a playlist consists of three steps. In a first step, the application collects songs that are close to the user's location, listed in ascending order of distance. To every song is associated a score that indicates their relevance to the user. This score is a combination of three factors, namely: preference, geographical relevance and freshness. In the last step, the tracks are arranged to achieve a pleasant-sounding mix for a good user experience.

One drawback in this solution is that anyone can

assign a song to a given location, however this assignment may have been made in a completely different context from that in which the person (user) hears the music which might lead one to think that the allocation made is meaningless in his context.

3. Developed Solution

3.1. Association between songs and activities

To develop an effective music recommendation system according to the activity that each user is performing it is necessary to understand users preferences in accordance with the task being performed. This section will therefore describe the method used to capture these tastes as well as the implementation decisions made during the process. We will also present the application used to collect the data to be used in the generic model, explaining how it works.

In the first phase of the recommendation process our solution recommends an equal set of songs for all users and each activity. This procedure corresponds to the interaction of the users with the generic model. This recommendation model is to be used by all users and his aim is to be framed taking into account the users interaction with the system. With the first interaction results the personalized model that is shaped according to the choices made by the user when using the generic model. Since it is from this model that we will generate the multiple custom models, it is important therefore that it captures general likings for all potential users.

In order to gather songs that all users might prefer, for each activity we developed a web application that will be explained in detail in the next section.

Due to the paradox of choice of Schwartz [10] which indicates that the difficulty of a choice is greater, the greater the range of options we have at our disposal, it was important to find a way to filter the vast amount of musics and artists available.

There are currently dozens of genres and their taxonomy is not well defined. Because of this difficulty of identifying music genres, we decided to use the one accepted by the majority of digital music services. Therefore we chose to represent 15 different genres. The number and genres were chosen based on the division of genres made by some services known in the field of digital music like last.fm, musicbox and musicoverly.com. Further the identification of the main musical genres there was the need of filtering by artist. This filtering considers three factors: the number of available music genres, the genres chosen for each activity during the association process and the current popularity of each existing artist. The number of artists displayed is 30, this is twice the number of genres. We consider this an ideal number of artists since it's not a very large number and ensures that in the worst case ce-

nario the user has at least two possible choices for each genre, which happens when the user chooses all the 15 genres for a given activity. Since there are millions of songs available, it was necessary to filter the number of songs to be presented. To achieve this objective filterings by genre and artists were important. Filterings by genre, allows us to choose the music genre that best fits the user needs depending on current activity. Filtering by artists in turn allows us to choose our favorite artists, reducing the number of possible songs shown. Yet for each activity we set the limit of 100 songs to be presented. These 100 songs are then divided by the total number of artists previously selected.

Having taken the previous decisions, we started to develop the web application whose purpose was to collect information on the preferred songs for each different activity supported by the system.

3.2. Data collection application

To gather information about the songs preferred for each activity, we developed a web-application that considers the decisions we presented in the previous section. The collected data will then be used in the creation of the generic model.

The usage of this application, for the creation of the generic model, can be divided into three major phases. First an activity is chosen randomly from those for which the user has not yet selected any music. Selected the current activity, a set of genres is shown to the user and he will have to choose at least one that conforms with his interests to the assigned activity (figure 1).

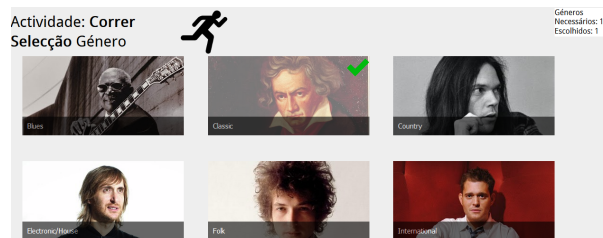


Figure 1: Genre selection.

In the second phase are presented artists that reflect the choices made in the previous step. These artists are obtained through the API of the echonest service ¹, being these artists the most relevant for each of the chosen genres. Here the user must select at least one to proceed to the last stage (Figure 2).

The last phase presents a set of 100 songs related to the artists previously chosen. For all songs shown the user must select at least 20 that he considers appropriate for the current activity (figure 3).

The web application was based on HTML, CSS and JavaScript technologies. For the collection of

¹echonest.com

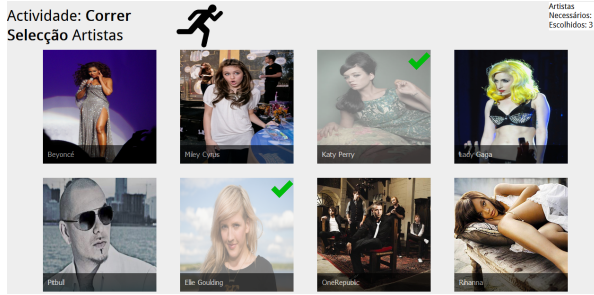


Figure 2: Artist selection.

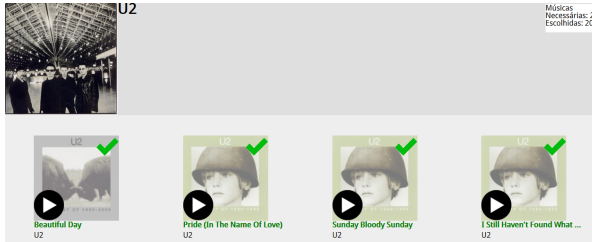


Figure 3: Music selection.

artists and songs we used the API provided by echonest service, as well as the service provided by deezer² that lets us play the songs previously chosen by echonest service. To store the information provided by the users we used PHP which save, the songs selected for each activity.

3.3. Recommendation Model

The generic model is fairly comprehensive and initially we intended it to be the same for all users. With this intuition we needed the input from a large range of listeners in order to attain a wide range of musical interests for the different activities supported by the system. This enabled us to determine the existence of dominant characteristics when choosing specific musics for specific activities. The choices collected from several users led to the identification of the top 100 songs for the various activities. Using then the echonest API we extracted the songs content information.

After collecting the music features it was necessary to evaluate any possible relationship between those and each users activity.

We used the Weka[11] software in order to better understand the influence of each feature, during the selection of a song in a given activity. Weka³ allows us to use various learning algorithms such as C4.5 decision trees and clustering algorithms such as the Expectation Maximization algorithm. In addition to these capabilities Weka can be used to evaluate the attributes present in a dataset in order to understand the most discriminatives.

²deezer.com

³<http://www.cs.waikato.ac.nz/ml/weka/>

This last feature was used in our dataset to identify the most important musical features. For this we used the CfsSubsetEval attribute evaluator along with the bestfit search method. After this evaluation we were able to conclude that acousticness, energy, loudness and tempo were the most important features capable of discriminating the relationship between users chosen music and activities. These are the features that we are going to use in the recommendation process of our solution.

Having chosen the most important features we used Weka to test different types of classifiers in order to understand if it would be easy distinguish songs using these features. We concluded that the Random Tree and Random Forest[12] classifiers produced the best results when verifying the quality of the features, and from now on all the tests made with classifiers will use these two.

Our results showed that the chosen features are good, in general they are sufficiently discriminative to predict what music should belong to each activity.

	a	b	c	d	e	f
a = Walking	70	5	0	0	5	20
b = Working	20	40	5	20	0	15
c = Relaxing	5	10	65	0	0	20
d = Running	15	15	0	60	0	10
e = Sleeping	5	0	0	0	95	0
f = Shopping	20	10	0	5	0	65

Table 1: Confusion matrix.

However through the confusion matrix shown in table 1, we were able to see that the working activity is the only one where not at least 50% of the songs belonging to that activity were detected. One of the reasons may be related to the fact of this activity being the most general among the six activities given in our solution, since the work that each person is undergoing might require both different physical and mental effort.

3.3.1. Activities Refinement

With data collected through the classifier, we noticed that there were some difficulties in distinguishing what songs should belong to the working activity, since these were often confused with running and walking activities (see table 1).

Due to the results obtained and to the fact that the working activity was the most subjective, we chose to exclude this activity from our solution. Thus we recreated the classifier in the same manner as explained above but with only 100 songs because the 20 songs belonging to the working activity were removed. The confusion matrix resulting from the removal of working activity can be found in table 2.

The removal of this activity improved the number

	a	b	c	d	e
a = Walking	60	5	5	10	20
b = Relaxing	10	80	0	0	10
c = Running	10	0	75	0	15
d = Sleeping	0	0	0	100	0
e = Shopping	15	10	5	0	70

Table 2: Confusion matrix with 5 activities.

of correctly classified songs in all activities, with the exception of walking activity. However overall the total number of songs correctly classified is higher than in the previous case. The results obtained so far lead us to exclude the working activity from our solution as it was the hardest to distinguish from all the others. We believe that this option will certainly improve the recommendation received by the user.

3.3.2. Generic Model

After having at our disposal the most important song features, as well the most chosen songs for each activity, it was necessary to understand how to use this information in order to achieve an effective recommendation.

A recommendation system is often seen as a suggestion of items similar to a feature vector that represents the users tastes. That is, the recommendation is based at a point in n-dimensional space where n represents the number of dimensions/features. However, the use of the echonest API doesn't allow us to search a set of songs given a single point in space. To overcome this problem, instead of a single point it was used a set of intervals delimiting the value that each feature could take, in space these group of intervals defines a hyper rectangle.

Taking into account the decisions taken earlier these hyper rectangles are made of 4 dimensions which are defined by the intervals calculated with maximum and minimum values that each feature accouticness, energy, loudness and tempo can take within each activity. The size of these rectangles differ between activities and users. In our solution since we had five activities, we will have not one but five hyper rectangles, one for each activity.

To find the best way to generate intervals that represent each song, was the next step in the model development.

We started by testing two different methods in order to calculate the intervals of each feature. The first proposed method used the mean minus the standard deviation for finding the minimum of the interval and the mean plus the standard deviation to find its maximum. The second method relied on the use of the 10% percentile as the minimum value of the interval and the 90% percentile for its maximum.

The intervals created using the two methods described above, were used to search songs through a website developed using the API echonest. The minimum and maximum values of the intervals created for each musical feature were used for this purpose.

Each query performed using those intervals returned 100 songs that were used with the Random Tree and Random Forest classification algorithms. With these tests we concluded that the methods used to generate the intervals were not adequate since the intervals generated were too large and there was a considerable overlap between them. The resulting recommendation was poor being quite similar for all activities.

Considering that the two tested methods had given poor results it was decided to use two new methods based on the first one, however we decided to use only a percentage of the standard deviation, namely 15, 20, 25 and 30%. In addition the resulting variant of the first method, another method was tested similar to the previous one but with the difference of using the median instead of the average.

The new methods were once again tested, using the 100 songs selected by the intervals of each feature generated for each activity as the test sets of the classification algorithms.

This time limits were generated in two different ways, one using the top 100 songs and the other using the top 20 selected by the users when using the data collection application presented on section 3.2. With this new variant a set of 16 different data sets were created (figure 4).

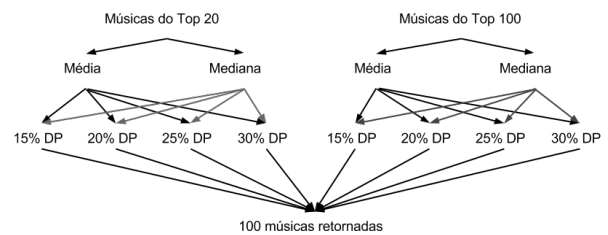


Figure 4: Generated data sets.

The top 20 and 100 songs selected previously by the users were used as training data sets of the classification algorithm. Taking this into account and since we had two different classification algorithms to test our intervals, we made a total of 64 test in order to determine the best method to calculate the intervals to be used in the recommendation process. The use of the top 20 and 100 songs for the interval creation was related to the need to understand if it would be beneficial or not to have a wide historical of songs (20 or 100) for the generation of the intervals, in order to obtain the highest percentage of songs considered correct for each activity.

For the 64 tests made the best result for the number of songs that deemed correct occurred for the method consisting on the average $\pm 15\%$ of the standard deviation. However it should be noticed that in this case the number of songs used to test the classifier (i.e. recommended) were only 331 instead of the 500 used in most of the other tests.

Given the low number of songs collected by the method mentioned above, we decided to opt for the second best method corresponding to the usage of the median $\pm 20\%$ of the standard deviation. In this case the median and standard deviation were calculated from the top 100 songs chosen by the users. The training instances used by the classifier were the top 20 songs chosen by the users for each activity. Tables 3 and 4 correspond to the confusion matrices for the two tests mentioned above (mean $\pm 15\%$ of the standard deviation and median $\pm 20\%$ of standard deviation).

	a	b	c	d	e
a = Walking	88.5	11.5	0	0	0
b = Relaxing	0	100	0	0	0
c = Running	40	0	57	0	3
d = Sleeping	0	5	0	95	0
e = Shopping	19	81	0	0	0

Table 3: Confusion matrix of mean $\pm 15\%$ standard deviation using Random Forest algorithm.

	a	b	c	d	e
a = Walking	53	11	30	0	6
b = Relaxing	0	85	0	0	15
c = Running	2	0	95	0	3
d = Sleeping	0	0	0	100	0
e = Shopping	52	30	3	0	15

Table 4: Confusion matrix of median $\pm 20\%$ standard deviation using Random Forest algorithm.

In addition to the problem mentioned earlier about the number of songs returned using the method of the average $\pm 15\%$ of the standard deviation, when analyzing the confusion matrix shown in table 3 we can verify that no songs of the shopping activity were assigned as belonging to that same activity.

In table 4 we can check an higher mess than the one presented in table 3, however in this case a total of 493 songs were returned using this method. A number of songs that were quite close to the desired value of 500 songs. Excepting for the shopping activity, in this matrix we can see that all activities had a success rate of over 50% of songs considered as belonging to each activity. Since the number of songs considered correct for the shopping activity were not significantly lower when compared with

those obtained in other tests, we decided to use the method that uses the top 100 songs for the calculation of the median $\pm 20\%$ of the standard deviation as the method to determine the intervals of the hyper rectangle (figure 5).

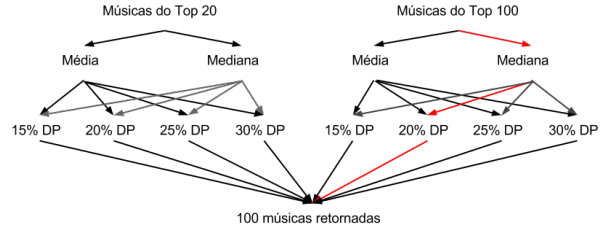


Figure 5: Method used to generate the recommendation intervals.

4. User Evaluation

After the best calculation method for the intervals was found, that information was used to create the generic model that will do the same recommendations for all users. In order to validate this model, we developed a web page where users could choose the songs they consider most appropriate for each activity. Some of the results obtained during the tests made with the generic model were then used to train the personalized model using the selected songs in the previous test. The web page used to test the personalized model was the same that was used to test the generic model.

The application for the evaluation of the generic model was built in with the aim to recommend a set of songs according to the activity that each user is performing. Users begin by answering a small questionnaire with their demographic information. Afterwards, a set of 5 activities supported by the system are displayed to the user from which he must choose the one that he is performing (Figure 6).



Figure 6: Activity selection.

To test the quality of the generic model we show 50 songs to the users that are within the intervals calculated according to the options taken in the previous section. Although a total of 50 songs are given to the users, in each moment only one song is present to the user.

In order that users have an idea of the music suggested, we play a sample of 30 seconds for each song, in detriment of the total length of each song to decrease the time spending during the process of choice.

In the developed application the user should select only the songs that he considers appropriate for each activity. Whenever a song does not suit his tastes, he should press the next button to move on to the next song (see figure 7).



Figure 7: Music selection.

For each suggested music, only a 30 seconds sample is played in order to decrease the time spending during the process of choice. An example of an interaction with the application used in the generic model can be seen in figure 7.

Our solution is based on the construction of a generic model that is initially equal for all users. The generic model will evolved to a personalized model as a result of the interaction with the user, adapting constantly to the choices made by him.

The evaluation was divided into two steps. The first step was to evaluate the generic model in order to understand if this is flexible enough to recommend music that comply with some of the preferences of potential users.

In the second step, some users who had previously interacted with the generic model, were asked to interact again with the recommender application. However this time the new songs are recommended taking into account the choices made by the users during their interaction with the generic model.

The new songs presented were chosen using the same method of calculation used in the creation of the intervals for the generic model. The aim of this procedure was to evaluate whether the model is adjusted according to user preferences. In this case an increase in the number of songs that each user would consider appropriate for each activity should be expected.

5. Results and Findings

The generic model evaluation was performed in two different ways. An online evaluation where we spread the web page described in section 5.1 via some social networks such as facebook and google+, and presential tests, where we asked users to test the generic model.

The online evaluation yielded a total of 21 responses and presential tests were carried out with the help of 10 users. The generic model was tested mainly by male users. In fact 15 of the online tests were made by male users, and 8 of the presential tests were also made with male users. The majority of the user's representing 81% of the total aged between 22 and 29 years old,. Five users corresponding to a total of 16% had ages between 16-21 years, and the remaining 3% were represented by just 1 user for the class age between 30 and 39 years old.

When comparing online tests with presential ones, there are significant differences in the number of songs chosen by each user. Online users only consider in average 4 to 5 songs as correct for each activity. This value increases for the presential users to an average between 11 and 12 songs per activity.

This difference may be related to the fact that when facing the users during the presential tests, they might probably be intimidated and as a result tend to be more concentrated on the tests responding therefore more attentively. The results for the average number of songs selected by activity are represented in figure 8.

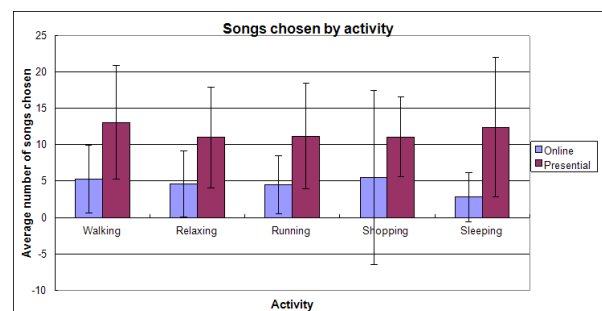


Figure 8: Average selected songs by activity.

Once selected the songs for each activity, users had to say whether or not they agreed with the recommendation made. The possible answers were: strongly disagree, disagree, neither agree nor dis-

agree, agree and strongly agree. This is a subjective evaluation, however the result allowed us to understand, approximately, the satisfaction level of the recommendation made for each user.

Using the scale to rate the degree of users satisfaction for each of the recommendations made, we can classify the first and second options as a negative feedback, the third as neutral and last two as a positive feedback. Using this division, we found an increase of 13% in the positive feedback by the users who made the presential tests, the percentages relating to the feedback provided by users are shown in figure 9.

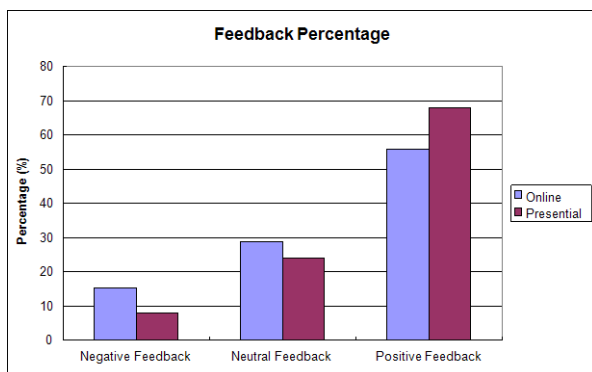


Figure 9: User feedback about the recommendation.

Although the users who had completed the online test in average selected a small number of songs, we could verify that the overall satisfaction with the recommended songs is positive. The same can be seen with the users who performed the tests in person, these results demonstrate that in general the recommendation meets the interests of users however for some reason in average the online users choose a much narrower number of songs.

In order to check if the generic model can be transformed into a personalized model, new tests were conducted with the 10 users who responded to the previous tests in person with the generic model. To test the personalized model, the intervals of the different music features were modified taking into account the choices made by each user in their interaction with the generic model.

The results obtained during the tests used with the personalized model can be seen in figure 10.

In the graph shown in figure 10 we can see that there was on average an improvement in the recommendation when using the personalized model, which can recommend on average more than 15 songs per activity.

After the evaluation made to all users using the personalized model, we made another test using the chosen songs during the first interaction with the personalized model. This test was intended to check if there was an improvement in the number of songs

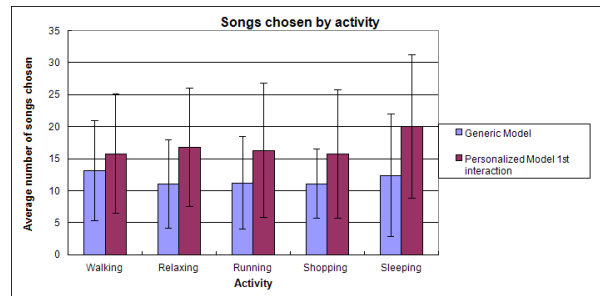


Figure 10: Personalized model results.

deemed appropriate for each user using the personalized model. Unfortunately it was not possible to perform these tests with all the 10 users who had experienced their personalized model, having been performed only by 5 of them.

figure 11 shows the evolution of the number of songs chosen by each of the users who used the personalized model for the second time.

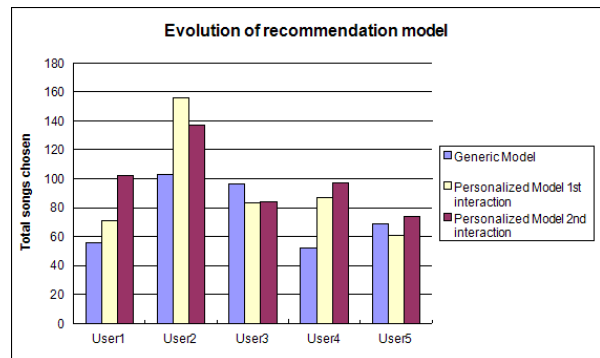


Figure 11: 2nd iteration using the personalized model.

From the obtained data we found that in both cases where there were fewer songs chosen during the use of the generic model there was a continuous increase in the recommendation using the personalized model. In the other cases a continuous grow on the recommendation was not found, but in only one case the personalized model was not able to recommend more songs than the generic model.

6. Conclusions and Future Work

With the arrival of digital music and Internet technologies, a great ammount of musical content is available to millions of users around the world. With the millions of artists and songs on market, begins to be difficult for users to search songs that please them. Furthermore, the large amount of music available has opened opportunity for the emergence of new recovery and music recommendation techniques.

With this work it was possible to perceive the challenges and the inherent complexity of the rec-

ommendation systems. Knowledge related to this area, namely in what respects the types of recommendation techniques and the challenges presented by them were acquired. At the same time it was important to realize the state of the art in recommender systems focused on music.

In our solution, we verified that the use of a generic model for all users can be beneficial for an initial recommendation. The capacity of this model for an initial recommendation is important because it allows fulfilling one of the great problems found in recommendation known as the Cold Start Problem.

The use of intervals to distinguish songs to recommend in each activity revealed to be a capable method of molding to the likings of the different users. This was visible with the improvement verified in the number of songs recommended during the passages of the generic model to the personalized one and through the second iteration made using the songs recommended by the personalized model.

These intervals thus allow to recommend musics based on their musical features and overcome one of the problems of some recommendation techniques known as popularity bias. In such a way we attained the intended objectives, in presenting a solution that has no difficulty in recommending songs to new users and that is capable to recommend any type of music be it known or unknown to him. However, the concept of freshness is not included in our solution, since the number of times that a music was already recommended to each user, as well as the last time that the recommendation was made was not taken into account. Given that we are not using the concept of freshness in our solution, the inclusion of this concept would be an important aspect to consider in the future. With this concept would be possible to assign a freshness score to each song, that would serve to have a more diverse recommendation, and promote the discovery of new songs.

It could also be interesting to choose randomly two new points within the current interval to represent a new maximum and minimum. These new points would be always within the current interval, the restriction of some values inside of the interval, would allow to verify more rapidly in which direction are the musical tastes of each user.

Finally, information concerning the names of artists whose songs were commonly chosen by users could also be used in the recommendation process. This allows us to recommend new songs of the user's preferred artists which had not yet been recommended. With the artists identity it should also be possible to look for similar ones. This could be interesting as it gives the user the opportunity of finding new songs. Such as the artists name, an-

other possibility is information of each song genre for each activity.

References

- [1] Christopher Avery and Richard Zeckhauser. Recommender systems for evaluating computer messages. *Commun. ACM*, pages 88–89, 1997.
- [2] N. Scaringella, G. Zoia, and D. Mlynek. Automatic genre classification of music content: a survey. *Signal Processing Magazine, IEEE*, pages 133–141, 2006.
- [3] Anind Kumar Dey. Providing architectural support for building context-aware applications. 2000.
- [4] Elaine Rich. Readings in intelligent user interfaces. pages 329–342, 1998.
- [5] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, pages 331–370, 2002.
- [6] Xinxi Wang, David Rosenblum, and Ye Wang. Context-aware mobile music recommendation for daily activities. *Proceedings of the 20th ACM international conference on Multimedia - MM '12*, page 10, 2012.
- [7] Jeff Mascia. Lifetrak : Music In Tune With Your Life Categories and Subject Descriptors. *1st ACM International Workshop on Human-Centered Multimedia*, pages 25–34, 2006.
- [8] Yajie Hu and Mitsunori Ogihara. Nextone player: A music recommendation system based on user behavior. *Proceedings of the 12th International Society for Music Information Retrieval Conference*, pages 103–108, 2011.
- [9] Anupriya Ankolekar and Thomas Sandholm. Foxtrot: a soundtrack for where you are. *Proceedings of Interacting with Sound Workshop: Exploring Context-Aware, Local and Social Audio Applications*, pages 26–31, 2011.
- [10] B. Schwartz. The Paradox of Choice: Why More Is Less. New York. page 304, 2005.
- [11] Geoffrey Holmes Bernhard Pfahringer Peter Reutemann Ian H. Witten Mark Hall, Eibe Frank. The weka data mining software: An update; sigkdd explorations, volume 11, issue 1. 2009.
- [12] Leo Breiman. Random forests. *Machine learning*, pages 1–33, 2001.