MusicTagger: Exploiting User Generated Game Data for Music Recommendation

Hannes Olivier, Marc Waselewsky, and Niels Pinkwart

Clausthal University of Technology
Tannenhöhe 25, 38678 Clausthal-Zellerfeld, Germany
{hannes.olivier,marc.waselewsky,niels.pinkwart}@tu-clausthal.de

Abstract. The system “MusicTagger” is a game in which two players hear 30 seconds of a song, describe it independently and get points if they succeed in making the same descriptions. Additionally, it is a music recommendation system which compares songs with the help of the descriptions given in the game. MusicTagger is based on the principle of “human computation”, meaning that problems (in this case, music recommendation) are solved by computers via eliciting human knowledge and making intelligent use of the aggregated information. This paper presents the design and implementation of the “MusicTagger” system together with results of an empirical lab study which demonstrates the potential of the recommendation engine.

1 Introduction

Deciding what music people would (potentially) like or not is a challenging but potentially highly valuable task for a computer system. There is quite a variety of attempts for creating good music recommendation systems. A prominent example is Apple Genius which is able to generate playlists of music similar to a selected song, relying essentially on collaborative filtering. Other approaches for music recommendation include the process proposed by Eck et al. [1], which relies on four steps for calculating song similarity: first, an acoustic feature extraction is calculated. After that, tags to the songs will be categorized. Next, AdaBoost, a meta-learning algorithm, is used for tag prediction. While not outperforming social tagging, first tests indicate a decent success of this approach [1]. Logan [2] used acoustic information for calculating a distance between songs. When new songs are integrated into a song collection, they will first be grouped to a song set which has the minimal distance. Yoshii, Goto, Komatani, Ogata and Okuno [3] used a hybrid music recommendation method. They tried to reduce the weaknesses of content-based recommendation and collaborative filtering by combining both approaches. Kuo, Chiang, Shan and Lee [4] designed an emotion-based music recommendation system. Based on their analysis of film music and the emotions this music conveys, they analyzed different music to assign emotional classifications and make recommendations based on this information. The system of presented in [5] is separated into seven segments: a track selector, a feature extractor, a classifier, a profile manager, a recommendation module, an interface and a database. New songs will first be integrated to the first two segments. The track selector categorizes the

J.A. Jacko (Ed.): Human-Computer Interaction, Part IV, HCII 2011, LNCS 6764, pp. 678–687, 2011.

songs as monophonic music objects or polyphonic music objects. The feature extractor then detects technical data of the songs (e.g., pitch density, pitch entropy, tempo degree and loudness). This data is needed for classifying the songs into music groups. The profile manager then saves information like the last date a user has looked for a song.

The "Friend of a Friend" (FOAF) and "Rich Site Summary" (RSS) vocabularies are used for the system presented in [6]. Here, music recommendations are generated in four steps: (1) get interests from user's FOAF profile, (2) detect artists and bands, (3) select related artists from artists encountered in the user's FOAF profile, and (4) rate results by relevance. Finally, TagATune [7] is a game where teams of two players play 3:30 minutes for a round. In this round, they hear seven songs for 30 seconds each. In these 30 seconds, they describe the song they hear and see the descriptions their partner gave. After each round, they have to evaluate if they have heard the same song or not. For a right evaluation, they get points. If they earn enough points, they can play a bonus round. In this, they hear three songs and they have to say which one has the biggest difference to the other two. If both players say the same, they get points. The descriptions in the first rounds and the evaluation of the (in)different songs are saved for a music recommendation system.

The approach for music recommendation presented in this paper is similar to the ESP Game [8] and to TagATune: it is based on the design principle of human computation where essentially humans solve tasks that are hard to do for computers (but where computers come into play in terms of aggregating and intelligently processing the human-generated data). In contrast to TagATune however, our approach aims at producing *categorized* key words as results (for example, artist or instrument), so that the underlying recommendation engine can rely on pre-classified information and can assign different weights to different types of information about songs.

In the next sections of this paper, the system MusicTagger and a pilot lab study conducted with the system to test the game and the recommendation engine are presented.

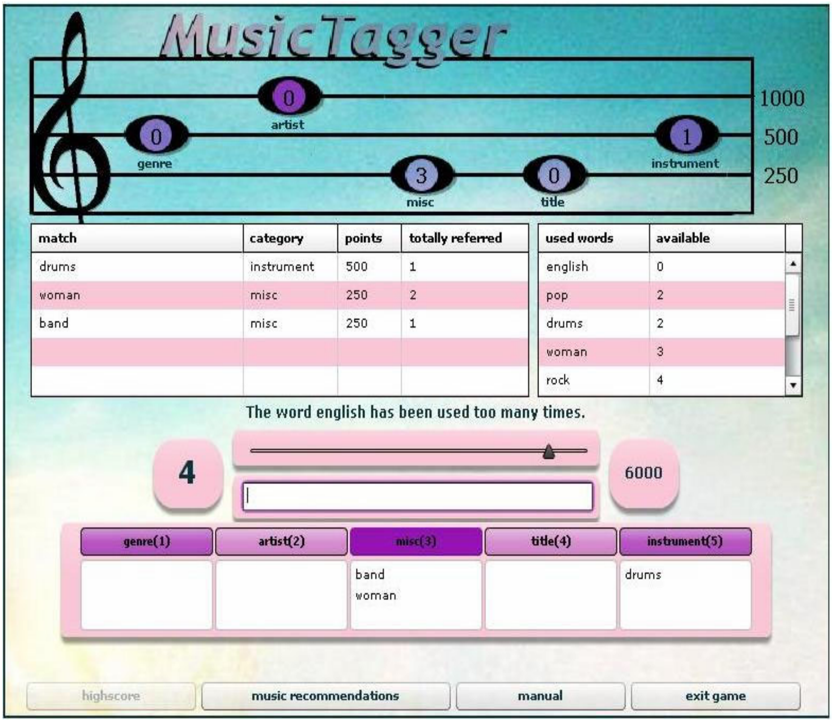
2 The MusicTagger System

The system is separated into two parts, the game to collect data and the recommender system using the generated data.

2.1 The Game

In the game, teams of two players hear the same part of a randomly chosen song. This part is 30 seconds long and chosen randomly from the length of the song. This approach was taken in order to account for changes in the song over time for obtaining key words (e.g., an intro might have a different style than a refrain).

While hearing the song snippet, players have to describe it to earn points. Figure 1 shows the main game screen. The players have five categories where they can put in their descriptions: genre, instrument, artist, title and miscellaneous. This is different from other music recommendation systems like TagATune that provide only one category. If both players add the same description word to a category, they get points



match	category	points	totally referred
drums	instrument	500	1
woman	misc	250	2
band	misc	250	1

used words	available
english	0
pop	2
drums	2
woman	3
rock	4

The word english has been used too many times.

Fig. 1. User interface of MusicTagger game

and can play one more round with each other (and this song description is added to the database for the respective category). Points are accumulated over the different rounds to encourage the players to continue with the game. For each matching word, 1000, 500 or 250 points can be earned. The point assignment is variable and depends on the current state of the database. The category with the fewest data contained for the played song yields the most points (in Figure 1, the category “artist”). This design encourages players to enter data that the system really needs in order to make good music recommendations.

In the user interface, the circles in the top of the screen also contain the number of words the other player already provided for each category (e.g., in Figure 1, one instrument keyword was typed in by the partner). This design was chosen to improve the player’s chances to find a match. A slider represents the remaining time, and the number to the right shows all points earned by the team. Below, the players can provide their descriptions category-wise and they can see which descriptions they have already given in this round (e.g, “drums” for the instrument category).

In a pilot study conducted with a preliminary version of the system, it was noticed that participants gamed the system by describing every song with the same words to earn points, regardless of what the songs were about. This resulted in wrong descriptions for the songs. To discourage this kind of behavior, a “black list” was integrated (table on the right side). The black list counts every word a team has already earned points with. Once a word has been used 5 times by the team, the word is forbidden from then on.

2.2 The Recommendation System

With the data collected by the game, the recommender system compares the songs to each other using the algorithm shown in Figure 2.

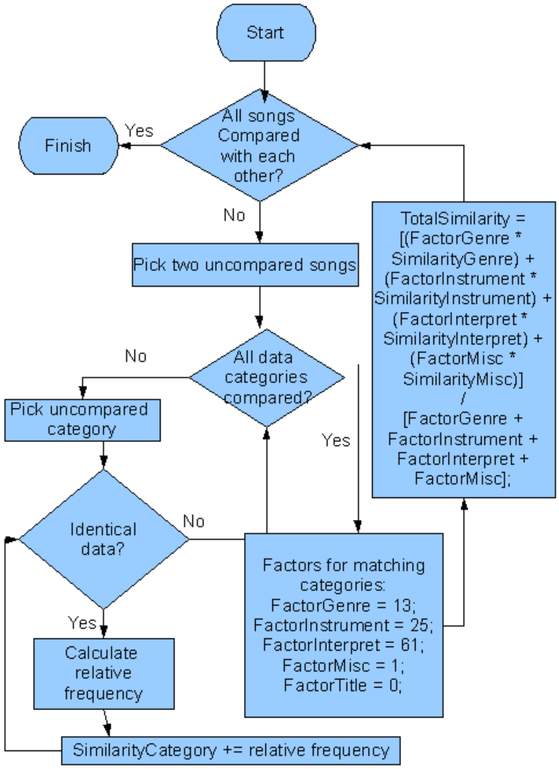

Start
All songs Compared with each other?
Yes
Finish
No
Pick two uncompared songs
All data categories compared?
No
Pick uncompared category
Yes
Identical data?
No
Yes
Calculate relative frequency
SimilarityCategory += relative frequency
Factors for matching categories:
FactorGenre = 13;
FactorInstrument = 25;
FactorInterpret = 61;
FactorMisc = 1;
FactorTitle = 0;
TotalSimilarity = [(FactorGenre * SimilarityGenre) + (FactorInstrument * SimilarityInstrument) + (FactorInterpret * SimilarityInterpret) + (FactorMisc * SimilarityMisc)] / [FactorGenre + FactorInstrument + FactorInterpret + FactorMisc];

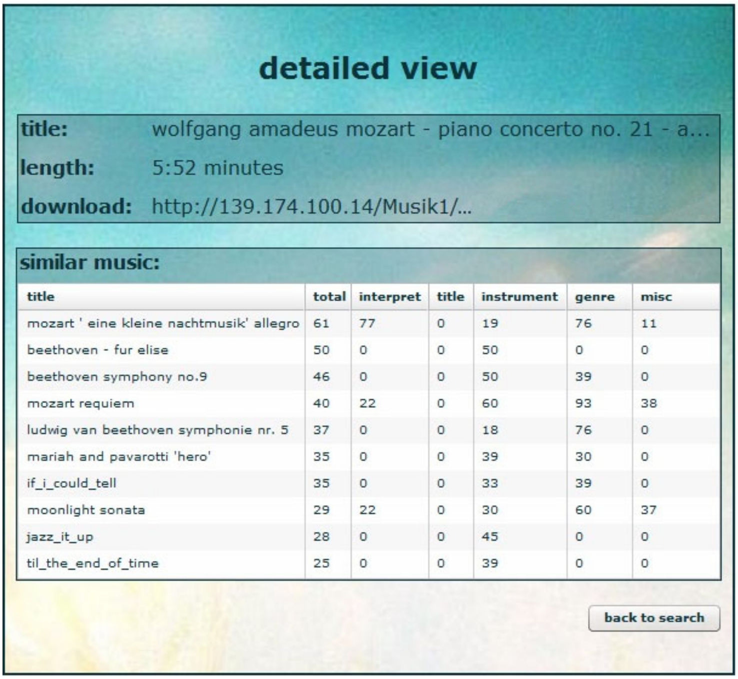
Fig. 2. Similarity calculation algorithm

In the algorithm, songs will be compared if there is at least one category (e.g., “genre”) where both songs have at least one data entry. For each such category, a similarity score will be calculated (which is 0 if the data entries for that category are different, 1 if they are identical, and in between if there are partial matches such as one song having 2 data entries and another having 3, with 1 of them being shared).

An example calculation: Let us assume we have two songs with one having (0, 3, 1, 1, 2) entries by category and the other having (0, 0, 2, 1, 3) entries respectively. Let us further assume that (0, 0, 0, 1, 1) of these entries are identical. In category one, both songs have no data: this has no influence to the total similarity. The second category will not influence the total similarity either because the second song has no entries in this category. Category three has entries for both songs, but no identical description, so its similarity in this category is 0%. The fourth category has only one entry for each song and the entries are identical, so this category has a similarity of 100%. In the fifth category, we first calculate the relative frequency for the identical word: It is 1/2 for the first song and 1/3 for the second song. Next, the smaller frequency is divided by the bigger one, and the result is multiplied with the average of the relative frequencies. The similarity for this category is thus min(1/2 / 1/3, 1/3 / 1/2) * (1/2 + 1/3) / 2 * 100 = 27.78%.

For these (max. 5) category similarities, a weighted average is calculated next. These weight factors are: genre 13, instrument 25, artist 61, miscellaneous 1, title 0 (i.e., the title will not be used to recommend songs, while same artist has a high weight). In our example, the third category (0%) could be the category “artist”, the fourth category (27.78% similarity) could be the category “instrument” and the fifth category (100% similarity) could be the category genre. Accordingly, the total song similarity is ((61 * 0) + (13 * 100) + (25 * 27.78)) / (61 + 13 + 25) = 20.15%

Note that the weight of the “song title” information may be subject to debate. Having different versions of the same song in a database might result in the title information being an important category (e.g., in the classical music section, different orchestras may play the same Mozart piece). For calculating the weight factors, 18 song pairs of a small pilot study were used. Participants of this study reported on the similarity of these 18 song pairs manually. The similarities (for each category) for the 18 song pairs and the weight factors for the algorithm were then calculated based on this data.



detailed view

title: wolfgang amadeus mozart - piano concerto no. 21 - a...

length: 5:52 minutes

download: http://139.174.100.14/Musik1/...

similar music:

title	total	interpret	title	instrument	genre	misc
mozart ' eine kleine nachtmusik' allegro	61	77	0	19	76	11
beethoven - fur elise	50	0	0	50	0	0
beethoven symphony no.9	46	0	0	50	39	0
mozart requiem	40	22	0	60	93	38
ludwig van beethoven symphonie nr. 5	37	0	0	18	76	0
mariah and pavarotti 'hero'	35	0	0	39	30	0
if_i_could_tell	35	0	0	33	39	0
moonlight sonata	29	22	0	30	60	37
jazz_it_up	28	0	0	45	0	0
til_the_end_of_time	25	0	0	39	0	0

Fig. 3. User interface of music recommendation engine

The user interface of the recommendation system allows users to search for songs in the database. They can perform a free-text search and the system then searches for songs which have the entered name or which have been described in the game with the entered key word. Then, the system shows a result list with relevant songs. By clicking on a result, the system shows the title of the selected song, its length, (optional) a link for downloading the song and a ranking with ten songs which have the highest similarity with the selected song. The table also includes the total similarity and the similarities for each category for all ten results. An example for a song of Mozart is shown in Figure 3. It shows the top-ten-ranking with "Mozart 'eine kleine nachtmusik' allegro" having the biggest similarity (61%) to the chosen song and "til_the_end_of_time" on rank ten with a similarity of 25%.

3 Research Questions

The MusicTagger system was evaluated in an empirical lab study in order to answer several research questions. One of the main problems of the previous pilot study with the preliminary version of the system was people cheating by using certain (fixed) words all the time to continue playing. The new version of the game now included a black list. We were interested whether the black list changed the players' behavior and whether its use resulted in different answers. Also, we were generally interested if the players enjoyed the game.

Concerning the recommendation system, we were interested in the following research questions:

- Do the similarities calculated by the recommendation system correlate with user generated similarities?
- Do the users accept the systems recommendations?
- Does more data generate better recommendations?

4 Study Description

For testing our system, a two-week study with 44 participants was conducted. In the first week, the participants played the game; in the second one they tested the recommendation engine. Not all participants showed up in week two, resulting in only 41 people evaluating the recommendation system. The participants were not informed that the two study sessions they participated in used the same underlying system. Most of them did not see the relation of the two sessions. Some even asked if these were two independent studies or if there was some small connection. The participants were paid 25 Euros. In both weeks, five sessions with 8 to 10 participants were conducted. The sessions in the first week needed an even number of participants, so that every participant had a partner to play with every time. In the second week, the participants worked alone, so the number of participants per session did not matter. All participants were in the same room, but they were sitting separated from each other and were not allowed to communicate with each other. This was enforced through the experimenter who was sitting in the room.

The database was filled with 102 songs. This number was chosen on purpose to be able to collect enough data to evaluate the recommendation system (too few songs will lead to repetitions, too many songs will lead to too few keywords for each song). We used music out of eleven genres (black, jazz, classic, dance, hip hop, metal, pop, rap, rock, alternative and folk music) with 9 songs for each genre. Three songs were duplicated (to check for the system-generated similarity for identical songs).

In the first week, the participants played the game and, as a side-effect, filled the database with information. Two versions of the game were played (one with blacklist, the dynamic category-dependent point system and some UI improvements, one without) to check whether the black list would improve the results of the answers. Everybody played two sessions (one with each version of the software) for 30 minutes. After the two sessions, the participants had to fill out a survey to evaluate both versions.

After this, the database was filled with 1100 descriptions (excluding entries from the “control version” that did not have the black list) for the 102 songs. Only 91 songs got descriptions from the participants, the other 11 did not get any (some of them were not played, some did not lead to scores).

To answer the question if the black list produced better results, the generated data from both versions was compared. The data of the non-blacklist version was also merged with the data from a previous study data to see if more data generates clearer results in the recommender system.

In the second week, the recommendation system was reviewed. 32 new pairs of songs (i.e., not containing the songs that belonged to the initial 18 pairs that were used to inform the algorithm) were prepared, with every song having at least one entry in the categories “genre”, “instrument” and “miscellaneous”. These 32 pairs belonged to four sets: eight pairs with the same artist, eight pairs with different artists and a system-calculated similarity >70%, eight pairs with different artists and a similarity < 30%, and eight pairs with different artists and a similarity between 30% and 70%.

Every participant was given one pair out of every set. The participants first listened to both songs of all pairs and then had to evaluate the similarity of their four pairs (in the categories genre, instrument and total similarity) on a scale from 0% to 100% in steps of 10%. After this, they were presented with the similarity results of the system for their four pairs and had to write down whether they thought the results of the system were satisfying or not. They described this by their own words in a free text. Each of the 32 test pairs was reviewed by at least 5 participants.

5 Results

Concerning the impact of black list, some results were found. 75% of the participants stated that the list influenced their gaming. They said that as they were looking at the “black list”, they tried to generate less common entries and not always standard words like rock, pop, guitar. These general terms were “reserved” for game situations where no other similarity was found or when the category offered a lot of points. The people that stated that they were not influenced by the black list said that they either never had the problem of blocked words or that they didn’t know a lot of words and accepted the risks of an early game end.

The database entries did not show a significant change in the overall use of "standard" keywords. The only observable tendency is that in the "black list" version, these words were used for fewer songs, which is in line with the statements of the participants.

The questionnaire at the end of the first study week contained some questions concerning the differences of the systems. The participants preferred the new, flexible point system. In the control version, the point table was not as flexible as it was in the experimental version and contained static points by category. The participants said that they were curios in each round which category would bring the most points and tried to reach at least one match in this category. This, of course, supports the design choice of making the points dependent on the state of the database (assigning many points to data entries that the system needs to make good recommendations).

Overall, the players enjoyed the game and gave average scores of 2.0 for motivation, 2.1 for usability and 2.1 for fun (on a scale of 1(good) to 5(bad)). These scores were assigned for the features of control system version. Additionally, the participants were asked about their opinion regarding the differences between the control and the experimental version. Here, they stated that they preferred the experimental version.

The similarities reported by the participants in the second week of the study were used to evaluate the recommender system. Comparing the system-generated and the user-generated similarities resulted in interesting insights.

The recommendation algorithm could only compare two of the three songs doubled in the database. The third one didn't receive descriptions. When the two songs were compared the system generated a similarity of 91 and 92%.

An example for a pair with a good result in the total similarity was a pair with two songs of the artist "Scooter". The system calculated a total similarity of 65% (100% artist, 0% genre, 37% instrument), the participants evaluated this song pair with 70% (on average) for all categories. An example for a poor result is one pair which consists of two songs from a German artist called Heintje. The system calculated a total similarity from 24%, but the participants stated a total similarity of 76%. In this case, the category artist contained no data, so the categories genre and instrument became very important for calculating the total similarity. But for both categories, one song had only one description entry and the other song had a few different descriptions, so that the category similarities were also low and finally, the total similarity became very low. This shows the problem of small data sets – with more data in the database (particularly with artist information), the calculation would have been more accurate. The current algorithm was able to produce an average difference to user generated evaluations of 15.4 percent (with sd=12.1).

The question if the participants were satisfied or dissatisfied with the similarity results delivered of the system was analyzed based on the free-text answers. In total, the participants answered they were satisfied with a result 106 times (65%), "still satisfied" 12 times (7%) and dissatisfied 46 times (28%). In interpreting this data, it has to be considered that sometimes humans do not agree at all: in one extreme case, one participant stated a pair similarity of 80% while another participant stated 20%. With similarity estimations diverging this far, an automated algorithm cannot satisfy both participants.

When the participants were asked about the top ten recommendations, they stated that the recommendations were overall very good. In most cases, only one or two songs were considered wrongly recommended. Here, one has to consider that in our study, only nine songs per category were available – so it is not surprising that a top ten list contains a few songs that not very similar.

By comparing the difference between the similarity score assigned by the users and the one calculated by the system, we additionally observed the following result:

- If the difference was 10 percentage points or less, the participants were always satisfied with result of the system.
- If the difference was between 11 and 15 percentage points, many, but not all participants were satisfied with the result of the system.
- If the difference was more than 15 percentage points, almost every participant was dissatisfied with the result of the system. Only in a few cases, when the total song similarity was less than 30%, a few participants said there were satisfied, because of the same tendency of low similarity. With higher song similarities, the participants looked more for the numerical value and were dissatisfied, because of the high difference.

According to this (relatively tough, but realistic) measure, we have three classes (<10%, 10-15%, >15%) in terms of difference between system and user-stated similarity. For our 32 song pairs, this classification leads to 14 satisfying results (44%), 8 mostly satisfying results (25%), and 10 bad results (31%), cf. Table 1 for details.

Table 1. Differences between system-calculated and user-stated similarity

	Satisfying (<10%)	Still satisfying (10-15%)	Dissatisfying (>15%)
Set 1: same artist	3	2	3
Set 2: different artist, >70% system similarity	3	2	3
Set 3: different artist, <30% system similarity	6	1	1
Set 4: different artist, 30%-70% similarity	2	3	3

6 Conclusion

In this paper, we report on a pilot study with the system “MusicTagger”, a two-part-system consisting of (1) a game which is based on the design principle of human computation, and (2) a music recommendation system which uses the descriptions delivered by game players for calculating music similarities. Users can search for a song they like, and based on their choice the ten songs with the highest similarity to the selected song are recommended.

The study showed that, even after relatively few hours of generating data through playing (and, thus, based on relatively few data), the users were satisfied with the game and with the results of the recommendation engine. Specifically, the dynamic allocation of game points based on the data entry types that the recommendation engine could benefit from and the “black list” which prevents gaming the system were successful. In the recommendation system, the majority of users were satisfied with the music similarity results delivered by the system (yet, of course larger studies with more songs and users will be required to confirm this result).

References

1. Eck, D., Lamere, P., Bertin-Matiuex, T., Green, S.: Automatic Generation of Social Tags for Music Recommendation. University of Montreal
2. Logan, B.: Music Recommendation from Songs Sets. Cambridge Research Laboratory, HP Laboratories Cambridge, HPL-2004-18 (2004)
3. Yoshii, K., Goto, M., Komatani, K., Ogata, T., Okuno, H.: Hybrid Collaborative and Content-based Music Recommendation Using Probabilistic Model with Latent User Preferences. University of Victoria (2006)
4. Kuo, F.-F., Chiang, M.-F., Shan, M.-K., Lee, S.-Y.: Emotion-based Music Recommendation By Association Discovery from Film Music (2005)
5. Chen, H.-C., Chen, A.L.P.: A music recommendation system based on music data grouping and user interests. Department of Computer Science, National Tsing Hua University (2001)
6. Celma, O., Ramírez, M., Herrera, P.: Foafing the music: a music recommendation system based on RSS feeds and user preferences. University of London, Queen Mary (2005)
7. Law, E.L.M., von Ahn, L., Dannenberg, R.B., Crawford, M.: TagATune: A game for music and sound annotation. Austrian Computer Society (2007)
8. von Ahn, L., Dabbish, L.: Labeling Images with a computer game. Vienna, Austria (2004)