A Music Recommendation System with a Dynamic K-means Clustering Algorithm

Dong-Moon Kim[1], Kun-su Kim[1], Kyo-Hyun Park[1] , Jee-Hyong Lee[1] and Keon Myung Lee[2]
[1]*Department of Electrical and Electronic Engineering, SungkyunKwan University, Korea*
[2]*School of Electrical and Computer Engineering, Chungbuk National University, Korea*
[1]*{skyscrape, kkundi, megagame}@skku.edu ,* [1]*jhlee@ece.skku.ac.kr,* [2]*kmlee@cbnu.ac.kr*

Abstract

A large number of people download music files easily from web sites. But rare music sites provide personalized services. So, we suggest a method for personalized services. We extract the properties of music from music's sound wave. We use STFT (Shortest Time Fourier Form) to analyze music's property. And we infer users' preferences from users' music list. To analyze users' preferences we propose a dynamic K-means clustering algorithm. The dynamic K-means clustering algorithm clusters the pieces in the music list dynamically adapting the number of clusters. We recommend pieces of music based on the clusters. The previous recommendation systems analyze a user's preference by simply averaging the properties of music in the user's list. So those cannot recommend correctly if a user prefers several genres of music. By using our K-means clustering algorithm, we can recommend pieces of music which are close to user's preference even though he likes several genres. We perform experiments with one hundred pieces of music. In this paper we present and evaluate algorithms to recommend music.

1. Introduction

A large number of people can download music files easily from web site. But only some sites provide personalized recommendation services. Generally, there are two approaches for the recommendation system: content-based and collaborative recommendation. The former analyzes the content of objects that a user has preferred in the past and recommends the ones with relevant content. The latter recommends objects that the user group of similar preference has liked. But the latter recommendation systems need much time to be stable and may recommend unsuitable music to a user because it does not consider each user's preference. Also the latter may not recommend pieces until quite a few users have listened. For that reason, we use the former approach.

But, there are three questions to be solved for personalized services: The first is how to analyze the property, the second is how to analyze a user's preference and the third is how to choose the pieces of music close to the user's preference. So, we suggest a solution about these three questions. The solution for the first is analyzing the sound wave of music. To analyze music's property, we use STFT (Shortest Time Fourier Form). From the STFT result, we can extract music's property in terms of numerical values.

To analyze a user's preference, we save the list of music the user downloaded or listened, and infer the user's preference from the list. We propose a dynamic k-means clustering algorithm for analyze users' preference. The number of clusters, k, should be given in advance for k-means to work properly. So if k is properly given, the data in the list maybe properly grouped and the grouped music can be used as the user's preference. However, choosing the proper value of k is not easy. We also restrict the range of clusters. The range of a cluster should be small enough so that it contains pieces of music which are similar. If the range of a cluster is large, the pieces of music which are not much similar to each other can be grouped together. So, the quality of recommendation possibly degrades. We control the number of clusters based on the range of clusters. If the range of some clusters exceeds the limit, k increases. This algorithm helps to prevent an unexpected result. The centers of clusters represent a user's preference. We can compare a user's preference with the properties of other pieces of music by Euclidean distance. If the property locates closer to the user's preference, we may conclude that the user will prefer the music also.

Usually, a web site has a very large number of music, so it may take a long time to find pieces closer to a user's preference. So we divide the pieces of music into small cells by their properties. If the system manager adjusts the size of cells, the system operates quickly. We compare a user's preference with the music which belongs to the cells similar to the user's preference.

0-7695-3069-9/07 $25.00 © 2007 IEEE
DOI 10.1109/ICMLA.2007.97

2. Related Work

P. Cano presented "content-based system." They extracted descriptions related to instrumentation, rhythm and harmony from music signal using similarity metrics [1]. G. Tzanetakis presented a way to extract feature from music, and divide musical genre automatically [5]. In this paper, we use G. Tzanetakis' method to analysis music. B. Logan presented that recommendation based solely on acoustics from groups of similar song of 'song sets' [2]. They extract song's property by MFCC (Mel Frequency Cepstral Coefficients). This approach is similar to ours. But MFCC makes 13 dimensional vectors.

H. Chen presents a collaborative recommendation system [3]. He groups pieces of music which are similar to each other and users who have similar preference. And it analyzes user's download list and recommends the music. B. Yapriady makes the demographic data by collaborative filtering [4].

There are many approaches to recommend system. But they did not concern about users' various kind of preferences. We focus on this problem.

3. Recommendation System

3.1. Analyzing music's property

A piece of music is composed of sound wave. Sound wave has may features (e.g. pitch, loudness, duration, timbre, etc.). Among these, we choose the ZCR, the spectral roll-off and the spectral flux. Because these features' combination has been used in voice and music matching algorithms and it has shown good accuracy [5] [6]. If we represent the feature of music as in a tuple form: (ZCR, Rolloff, Flux), a piece of music can be mapped into a point in three dimensional spaces.

ZCR (Zero Crossing Rate): The zero crossings provide a measure of the noisiness of the signal. For example heavy metal music due to guitar distortion and lots of drums will tend to have much higher zero crossing values than classical music [5]. The ZCR, Z_t, is defined as follows:

$$Z_t = \sum_{n=1}^{N} \frac{\left| sign\left(x[n]\right) - sign\left(x[n-1]\right) \right|}{2} \qquad (1)$$

where the sign function is 1 for positive arguments and 0 for negative arguments and x[n] is the time domain signal at frame t.

Spectral Rolloff: The spectral rolloff is defined as the frequency R_t below which 85% of the magnitude distribution is concentrated [5]:

$$\sum_{n=1}^{R_t} M_t[n] = 0.85 \times \sum_{n=1}^{N} M_t[n] \qquad (2)$$

where Mt is the magnitude of the Fourier transform at frame t and frequency bin n. The rolloff is another measure of the spectral shape and shows how much of the signal's energy is concentrated in the lower frequencies.

Spectral Flux: The spectral flux is defined as the squared difference between the normalized magnitudes of successive spectral distributions [5]:

$$F_t = \sum_{n=1}^{N} \left(N_t[n] - N_{t-1}[n] \right)^2 \qquad (3)$$

where $N_t[n]$, $N_{t-1}[n]$ are the normalized magnitude of the Fourier transform at the frame t, and frame t-1, respectively. The spectral flux is a measure of the amount of the local spectral change. The spectral flux has also been shown by user experiments to be an important perceptual attribute in the characterization of musical instrument timbre.

3.2. Analyzing user's property by a dynamic K-means algorithm

A user's music list is growing as the user downloads or listen pieces of the music. The list can be regarded as the accumulation of the user's preference. So we have to extract a representative value from the list. One of candidates is the average of the list. But it is may not a good representative value. If a user prefers the music in two different genres, it cannot reach a correct result. For example, a user prefers classical and rock music, the average of list will not be classical nor rock music. There have been many approaches for recommend systems. But most of them assume that user will prefer only one kind genre of music, not various kinds. Thus, a simple average is used for a representative of user's preference. We assume that a user may prefer many different genres of music. So, instead of simple averaging, we first cluster the pieces of music in the list according to feature values. Then we use the average values of clusters as a user's preference. If a user likes two genres of music, it is highly probable that his music list is clustered into two clusters.

K-means is one of the simplest unsupervised learning algorithms that solves the clustering problem. However, K-means algorithm needs to be given k, which is the number of clusters in advance. But we do not know what kinds of music a user likes. A user can prefer many genres. So we need to change k dynamically. Even though, the music list is properly clustered, there is one more question. Let us suppose that a user's data is clustered into two groups: one is small and the other is large as shown in Fig. 1. Then, each center of the clusters will be a good representative value for the user's preference. So, we may recommend the pieces close to either center.

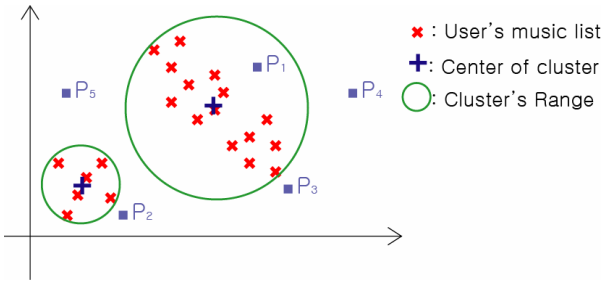

Fig. 1. Example of a large cluster: improper to represent a user's preference

P_1 and P_2 may be recommended but P_3 may not. Because P_3 is not close to one of the centers even though it is close to the user's preference. So, we limit the radius of a cluster to Rmax, Rmax is also used for dynamic change of k. If the radiuses of some clusters are larger than Rmax, k is increased. We repeat this until each cluster's radius become shorter than Rmax.

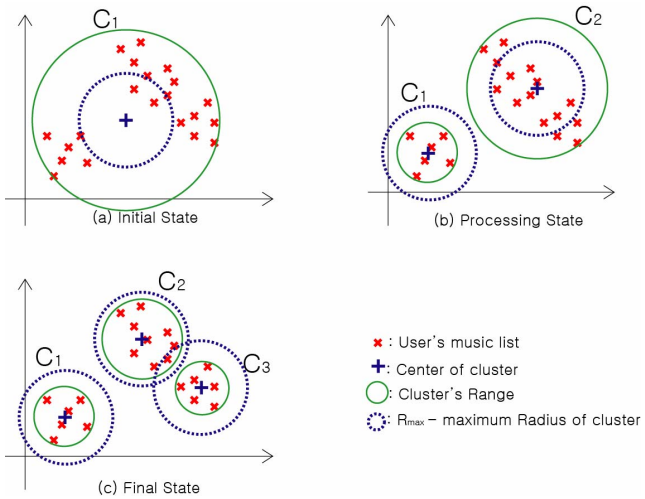

Fig. 2. Clustering steps

Fig. 2 shows the steps of clustering. Initially, k is 1, so all data is grouped into one cluster as shown in Fig. 2-(a). The radius is larger than R_{max}. So, k is increased by one and the data is clustered as Fig. 2-(b), still the radius of C2 is larger than R_{max}. Thus, k is increased. In Fig. 2-(c), all clusters are smaller than R_{max}. This is the final clustering result.

3.3. Recommendation list

We score each candidate piece of music based on the clustering result. The score of music m is the summation of the scores from each cluster. The score from a cluster is defined as:

$$\text{Score}_C(m) = \frac{1}{Dist(v_C, v_m)} \times \frac{ClusterData_C}{AllData} \quad (4)$$

where AllData is the number of pieces in the user's list, and ClusterData_C is the number of pieces inside cluster C. And $\text{Dist}(v_C, v_m)$ is the Euclidian distance between the center of cluster C, v_C, and a candidate music, m; In the feature space, m is mapped into a point, v_m. The score will be higher if the music is closer to a cluster and the number of pieces in the cluster is larger. The recommendation list is created according to the scores.

3.4. Implementation

The number of pieces of music is increasing day by day. So it needs to reduce the search time for real-time recommendation. To do this, we divide the space into small cells. And the system searches the cells close to the user's preference. The size of cells is defined by the system manager. If the size is small, searching speed will improve, but the number of searched data may be not enough. In this paper, we search into the neighboring cells as well as the closest one.

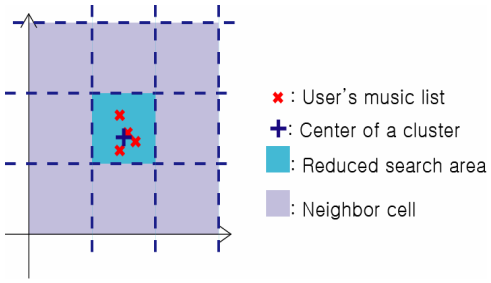

Fig. 3. Space partition method

4. Experiment

The second and following pages should begin 1.0 inch (2.54 cm) from the top edge. On all pages, the bottom margin should be 1-1/8 inches (2.86 cm) from the bottom edge of the page for 8.5 x 11-inch paper; for A4 paper, approximately 1-5/8 inches (4.13 cm) from the bottom edge of the page.

For experiment, we build music database. The music database is composed of 100 pieces of music from ballad, rock, jazz and classical genre: 25 pieces per genre. A music file is composed of 32bit 2channel and 44100Hz form. The feature of a piece of music is normalized by the length of the playtime. It makes all pieces of music virtually to be the equal length. Each feature of a piece is divided by the average (for example, the ZCR is divided by the ZCR's average). By this way, three types of data are normalized. And we define the maximum radius R_{max} as 0.5 and cell's size as 0.2.

We explain the recommendation process step by step. Let us suppose that a user downloads and listens following three pieces:

- Mozart – Piano sonata no. 21
- Bizet – Carmen suite no. 2
- Vivaldi – Four season: autumn the 1st movement

The property of the pieces, (ZCR, Roll-off, Flux) / playtime are:

- (179895, 6997, 115000) / 101
- (216980, 4593, 321000) / 100
- (474532, 24881, 1306000) / 197

The normalized property are:

- (0.523217, 0.009610, 0.158465)
- (0.637388, 0.027014, 0.105065)
- (0.707593, 0.055783, 0.288911)

If we apply our clustering algorithm, only one cluster is found and its center is:

- (0.622733, 0.0308023, 0.184147)

Based in this center, we score the candidate pieces and top 5 pieces are recommended as shown in Table 1.

TABLE Ⅰ

MUSIC RECOMMENDATION IN CLASSICAL

Rank	Title	Property	Genre
1	Bizet – Minuet	(0.514, 0.115, 0.011)	Classical
2	Saint Saens – "The Swan"	(0.523, 0.158, 0.010)	Classical
3	Gounod – Ave Maria	(0.433, 0.102, 0.005)	Classical
4	"March" in Nutcracker	(0.712, 0.319, 0.558)	Classical
5	Kreisler - Leibeseleid	(0.604, 0.358, 0.070)	Classical

We evaluate our system with the cases where users prefer only one genre of music. We assume that users listen five pieces in a genre. The genres are ballad, rock, jazz and classical. We repeat the test 10 times per genre. We get the result as Table 2. In the cases where users listen only the pieces in ballad, 56% of the recommended pieces in ballad, 12% rock, 26% jazz and 6% classical. In the cases of rock, 22%, 70%, 8% and 0%. From the table, it is noted that our system mostly recommend the pieces in the genre users like.

TABLE Ⅱ

A USER LISTENS TO MUSIC OF A SINGLE GENRE

User's list / Recommendation List	Ballad	Rock	Jazz	Classical
Ballad	56%	22%	28%	8%
Rock	12%	70%	10%	6%
Jazz	26%	8%	62%	12%
Classical	6%	0%	0%	74%

Next, we assume that users prefer only one of male music, female music or instrument only. There are no singer in classical and some jazz music, so it classified as 'instrument only'. We experiment in a similar manner of the first. We get Table 3 and it shows that the system properly recommend to the users.

TABLE Ⅲ

A USER LISTENS TO MUSIC OF SINGER TYPE: MALE, FEMALE, INSTRUMENT ONLY

User's list / Recommendation list	Male	Female	Instrument only
Male	72%	16%	8%
Female	16%	76%	12%
Instrument only	8%	8%	80%

Finally, we assume that users prefer rock and classical. Rock and classical are very different from each other. A user listens five pieces of each genre. We compare the previous method with our algorithm. Table 4 shows results that compares the previous K-means algorithm and ours. In the case of the previous K-means, the experiment was performed four times with different numbers of clusters: K = 1, 2, 3 and 4. In case of the simple average method, that is K is 1, it recommends more ballad music than any other genres. Sixty percent of the recommended pieces are ballad which is not the user's preference. 25% are rock, 16% are jazz and 4% are classical. Only 29% are close to the user's preference. As K is increased, it can recommend more correctly. Generally, if K is large enough, the recommendation result has a tendency to be close to the user's preference. But the recommendation time will increase proportionally to the value of K. So if K is large, it may take a long time to recommend. Thus, the value of K should carefully be chosen so that the recommendation result is closer to user's preference. However, it is not easy task because the proper value of K maybe different case by case. When K is 3or 4, the method shows quite good recommendation results. In the

case of K = 4, rock is 36% and classical is 44%. Thus 80% of the recommended pieces are close to user's preference. Our method overcomes this problem. Our method automatically regulates the value of K. And it mainly recommends rock and classical music. Seventy-four percent (34% rock + 40% classical) belongs to the user's preference genres. So we may say that our method can reflect users' various interest in music.

TABLE Ⅳ

A USER LISTENS TO MUSIC OF TWO GENRE OF MUSIC: ROCK AND CLASSICAL

User's list / Recommendation List	K-means				Dynamic K-means
	K=1	K=2	K=3	K=4	
Ballad	60%	32%	16%	8%	10%
Rock	25%	16%	40%	36%	34%
Jazz	16%	4%	8%	4%	6%
Classical	4%	48%	36%	44%	40%

5. Conclusion

A large number of people download music files easily from web site. But rare music sites provide personalized services. In this paper, we analyze music's property from sound wave, and inference a user's preference from the user's list. Also, we propose a dynamic K-means algorithm which is usefully applicable to recommendation. And we use a space partition method to speed up the recommendation process. In this method, we can effectively handle large music data. We performed various experiments on genre and gender, multiple genres. Each experiment shows that our system operates well.

6. References

[1] A.B. Smith, C.D. Jones, and E.F. Roberts, "Article Title", *Journal*, Publisher, Location, Date, pp. 1-10.

[2] Jones, C.D., A.B. Smith, and E.F. Roberts, *Book Title*, Publisher, Location, Date.

[1] P. Cano, M. Koppenberger, N. Wack, "Content-based music audio recommendation," *Proc. of ACM international conference on Multimedia*, pp. 211-212, 2005.

[2] B. Logan. "Music recommendation form song sets," *Proc. of ISMIR*, pp. 425-428, 2004

[3] H. C. Chen, A. L. P. Chen. "A music recommendation system based on music data grouping and user interests," *Proc. of CIKM*, pp. 231-238, 2001.

[4] B. Yapriady, A. L. Uitdenbogerd, "Combining demographic data with collaborative filtering for automatic music recommendation," *Lecture notes in computer science*, Springer, pp. 201-207, 2005

[5] G. Tzanetakis, P. Cook, "Musical genre classification of audio signals," IEEE *Trans. Speech and Audio Processing*, pp. 293-302, 2002.

[6] J. H. Ban, K. M. Kim, K. S. Park, "Quick audio retrieval using multiple feature vector." *Proc. of the acoustic society of Korea spring conference*, pp. 351-354, 2004(in Korean).