

Similarity Clustering of Music Files According to User Preference

Bastian Tenbergen

Human-Computer Interaction M.A. Program
State University of New York at Oswego
Oswego, NY, 13126
Formerly:
Cognitive Science Bachelor Program
School of Human Sciences
University of Osnabrück
Germany

Abstract. A plug-in for the Machine Learning Environment Yale has been developed that automatically structures digital music corpora into similarity clusters using a SOM on the basis of features that are extracted from files in a test corpus. Perceptionally similar music files are represented in the same cluster. A human user was asked to rate music files according to their subjective similarity. Compared to the user's judgment, the system had a mean accuracy of 65.7%. The accuracy of the framework increases with the size of the music corpus to a maximum of 75%. The study at hand shows that it is possible to categorize music files into similarity clusters by taking solely mathematical features into account that have been extracted from the files themselves. This allows for a variety of different applications like lowering the search space in manual music comparison, or content-based music recommendation.

1 Introduction

During the last few years, the channels of acquiring musical data have increased rapidly. Due to the growing importance of broad-band network connections, users are no longer limited to radio or television broadcasting services or other non-computer aided methods to acquire copies of musical pieces. More or less legal peer-to-peer networks, online music stores and personalized Internet radio stations provide the user with a constant and never-ending flow of musical information. On the one hand this large supply makes it possible to get any piece of music any time with a minimal amount of costs. On the other hand, this gives rise to a new, challenging problem: Structuring the music collection of an individual user or a user community to allow for tasks like content-based music recommendation. Keeping the overview over a large collection becomes more difficult and costly with the size of the collection. Unnamed files or files with an unknown content are very difficult to identify and classify manually, which makes a system desirable that is able to accomplish the task

of disambiguating the information about a collection automatically, so two or more users can share these disambiguated information.

Many different research teams have addressed the problem of music information retrieval in different ways. Research is being conducted on the basis of non-real-world audio data, [13], [16], [17], i.e. data of musical information that need to be interpreted in one form or another. Among this kind of musical information belong both score representations of pieces as well as music stored as MIDI files. Although research in musical information retrieval based on non-real-world data sets reveals very promising results, the problem is that those kinds of data sets are usually not comparable to the collections that are stored on a user's hard disc or on broadcasting station servers. It is more likely that a form of compressed or uncompressed real-world audio data (like compact discs or MP3 files) can be found in a collector's database (throughout this work, the term "user" will be used to refer to any kind of individuals that possess and/or consume audio data – no matter if it is a home user, a radio or television station, a web-radio service or a music store, either online or not). Because of this, a system that can deal with real-world audio data is of a much higher value for single users or user groups than a system that can deal with pre-interpreted score- or MIDI-like representations instead. Processing real-world audio data for music information retrieval purposes can generally be done using two prominent approaches: a statistical or mathematical approach [10] and a more neurophysiological approach [15], both possibly involving neural networks [14]. Since the ability to retrieve musical information from real-world audio data is a very impressive capability of the human brain, especially the application of artificial neural networks provides distinguished results. Both approaches in dealing with real-world data make it necessary to extract certain audio features from the data before hand.

Pardo, Shlfrin and Birmingham [13] have conducted a pilot study in matching a sung query to a database of stored MIDI files. They especially address the problem of erroneous queries due to a low humming performance of the user. The outcomes of the experiments were that specifically designed user-based error models provide much better results in handling a mismatch between query and target songs than the complete absence of any error model. Yet, synthetic error models provide equally good results than data-driven, singer-based error models. However, in order to provide a framework for music recommendation, relying on the users to sing or hum a query seems not very applicable. Hence, it might be more useful to have a system that collects and processes data from the collection itself.

Tzanetakis et al. [15] have contributed an important milestone in musical genre classification. The authors argue that human listeners can classify musical content by referring only to small excerpts of a piece, e.g. a few fractions of a second. Hence, human listeners must rely on musical surface features instead of complex (mathematical) interaction between waveforms of the piece. So, Tzanetakis et al. focus on the extraction of surface and rhythm properties and propose a set of own features. The authors proposed two different graphical user interfaces (GUIs) that visualize a song's membership to a certain genre. Since the borderlines of musical genres are rather fuzzy, a nice side effect of the GUIs is that not only the genre of a song is displayed but rather a number of other genres the song shows characteristics of are visualized.

Up to this point, all researchers who extract features from digital audio data mostly presented their own features. This leads to a large variety of features that can be extracted from audio data. However, a unifying framework that measures the features' applicability was missing. Mierswa and Morik [10] were the first researchers to address this problem. In order to accomplish this task, Mierswa and Morik made use of a Genetic Programming approach to explore the space of all possible feature extraction methods in order to learn the feature extraction method that has an ideal performance to classify audio data into genres. Three test corpora were compiled and consisted of music files that belong to either the Pop or Techno genre, the Pop or Hip hop genre or to Pop or Classic music. The files in every corpus needed to be classified in either of the two genres in the corpus.

The article at hand aims at a more general approach to the problem of music collection structuring, as most users probably do not have a pre-structured database. A system is suggested that assists users in the task of finding music pieces by presenting clusters of similar songs according to an input query. This way, a possibly very large collection of song files can be narrowed down to a choice of much fewer songs. This allows for lowering the complexity in comparing songs manually, since a global comparison has already been performed by the system. To accomplish this task, a plug-in for the Machine Learning Environment Yale has been developed. The plug-in makes use of a Self-Organizing Map [4], [5] to cluster audio files and to create similarity clusters. The files need to be processed in a feature extraction algorithm that extracts certain features [10] from the underlying corpus.

Dividing songs into similarity clusters allows for a variety of different tasks. As an example, meta information for unnamed song files can be retrieved. Unidentified files can easily be compared with songs in a cluster containing much fewer elements than the original corpus (which might be excessively large). This helps user groups as well as single users to keep an overview of large music databases. In addition, the suggested system can help automating user preference analysis to individualize the offer of online music stores and other personalized music services.

2 Method

2.1 Experimental Set-Up

In order to classify musical data into perceptually similar clusters, a plug-in for the Machine Learning Environment Yale has been developed. The plug-in contains a simple Self-Organizing Map operator. This Self-Organizing Map basically lays a network of artificial neurons over the feature vectors. Each feature vector represents a music file in the corpus that needs to be clustered and consists of a number of generic features (like mean loudness, maximum pitch, average beats per minute, etc. [10]). Essentially, the SOM performs a mapping of the high dimensionality of the feature vectors onto a two dimensional representation. The result is that the SOM's layout changes according to the distribution of the feature vectors so that each pattern that naturally exists in the vector space is represented by one or more neurons.

The features are extracted from the music files by making use of a feature extraction method that has provided good results on a variety of different tasks in other research [10].

The underlying requirement of all experiments is that the audio corpus on which the experiments are performed on consists of generic MPEG-1 Layer III files with a sampling rate of 44.1KHz [3]. The proposed clustering framework is robust against corpora consisting of files encoded with different bitrates.

The primary test corpus consisted of 360 audio files, all differing in length, bitrate and genre. These files were subdivided into six sub-corpora: three small ones, each containing approximately 20 files, and three medium ones, each containing 100 files. The files in the small sub-corpora were taken from a private collection and vary in style and genre. The medium corpora contained the (unofficial) top 100 chart songs from the years 1990, 2000 and 2005, respectively. To test the proposed system for applicability in very large collections, a secondary corpus consisting of 1,554 files, i.e. the (unofficial) charts from the years 1990 to 2005 has been compiled and processed. Although an effort has been made to take songs from as many different genres as possible, this research focuses on modern popular music, ranging from early pop stages to fairly recent modern publications. Recapitulating, the overall music corpus consisted of approximately 8.07 gigabytes in 1,914 files summing up to over 160 hours of total playtime. The files in the source directory are generally not structured although recursive processing of subdirectories is supported to take databases into account that have been pre-structured by a user. Similarity clusters are found on the bases of a query file. This file is located within the source directory and is hence processed in terms of feature extraction too. The query file represents a file, the user is interested in finding similar songs to.

2.2 Procedure

The basic experimental procedure has been divided into two steps. In the first step, the raw data are transformed into a feature vector representation by simply extracting the features from the music files in the test corpus. The output of the feature extraction step is a file containing the feature vectors. In the second step, this file is used by the clustering SOM operator to train the Self-Organizing Map. In addition, the SOM operator retrieves the file name of the query file from the raw audio data. After the clustering process is done, this file name is retrieved from the similarity clusters that have emerged from the process. The system's output is a string representation of all file names of the songs that are in the same cluster as the query file.

In order to test the clustering performance, the Self-Organizing Map was applied on the extracted features with a variety of different parameter settings for the overall number of neurons. Since the number of clusters cannot exceed the number of neurons in the SOM, but increasing the number of neurons negatively influences the time needed to complete the clustering, an ideal number of neurons that allows for a maximum of clusters needs to be found – a trade-off. All test runs of the SOM had the following parameters in common. The network topology was always set to an “open“

$$n \times m \tag{1}$$

matrix, with

$$n, m > 1, \quad (2)$$

i.e. the neurons in the SOM were aligned in a rectangular structure – the first and last neuron in every row and column were not considered to be neighbors. The number of training runs was set to 1,000 and the learning rates for the winner neuron and its neighboring neurons was set to 1 and 0.25 respectively. According to [14], no superiority of any distance metric over another metric exists. Hence, all test runs have been performed using the Euclidean Distance as the metric to measure the distance of every neuron on the SOM to every input feature vector (that represents a certain song).

To test, which features are most important for the classification task, test runs were performed with normalized and filtered patterns. The patterns have been normalized by dividing each pattern $x_{i,j}$ – with i being the i -th pattern (i.e. i -th song from the source directory) and j being the j -th feature of the i -th song – by the root mean square deviation of the respective column in the pattern matrix. A filtering of the patterns was done by setting every pattern with a very small absolute value, i.e.

$$x_{i,j} \leq 1, \quad (3)$$

to zero.

From each corpus, one file was chosen to be the query file. Since the query file is the file that users will enter as a sample of their musical taste (for instance), the evaluation focuses on the cluster containing this file to ensure that the suggested similar songs are indeed similar.

To evaluate the similarity of the songs in the cluster containing the query file, the cluster was transposed into a play list and evaluated by a human user. The user was asked to rate each song on a 6-point scale (0: no similarity; 1: little similarity; 2: medium to little similarity; 3: medium to high similarity; 4: high similarity; 5: very similar/close to equal) using the query file as the prototype. The user was asked to consider two songs a and b similar, if the

musical style of both songs is subjectively similar,
musical themes can be described as subjectively similar, and
pace and/or rhythms of the songs are subjectively similar.

Two songs can also be considered similar, if they are not members of the same (subjective) genre. The user was a twenty-two year old male German student, with no background in music theory and music science. The participation in this study was on a voluntary basis and the user did not receive a reward. Prior the ranking of the songs, the user was naive to the purpose of this study to avoid a user bias. The songs were presented to the user in a random order. Before the user evaluated each song in the similarity cluster, the prototype song was presented to him. He could listen to the song as often as he wanted throughout the evaluation phase.

3 Results

While conducting the experiments, special attention has been paid to the performance of the clustering process of the Self-Organizing Map. Generally speaking, the

Self-Organizing Map was able to find similarity clusters among the music files and hence provided good results in structuring the source directory.

In the small test corpora, only a very limited number of clusters could be found. Increasing the number of neurons did not influence the number of clusters. Accordingly, the optimal number of neurons for the small and medium test corpora does not depend on the number of found similarity clusters as well, as long as the SOM contained more neurons than the maximum number of clusters that can be found with respect to a test corpus (that is in worst case: one cluster per song in the corpus).

For every test run, an average of 12 clusters could be found. Optimal clustering performance was achieved by setting the number of neurons to approximately 30% of the number of files in the test corpus. As it was expectable, a very large number of clusters was found in the large test corpus. However, the number of found clusters did not exceed 25% of the number of files in the corpus. For no test run, a very large number of neurons (i.e. in cases in which the number of neurons exceeded the number of files in the corpus by a multiple) influenced the clustering performance negatively.

Normalizing the feature patterns negatively influenced the clustering performance. In all test runs, only two clusters could be found – independent from the chosen target file. Optimal results were provided in test runs in which normalizing of the patterns did not take place. It is to note that the extracted feature vectors have a very wide scope, i.e. some features have a very small absolute value, while others have a very large absolute value. Filtering the feature patterns in every test run did not influence the clustering performance negatively. In most cases, the clusters found with the filtering option activated were the same as the similarity clusters without filtering having taken place.

Generally speaking, applying the SOM on the features extracted from the small corpora resulted in two or three clusters that are found amongst the input files. Although filtering did not influence any of the other test corpora, in the first small corpus, finding similar files was highly dependent on whether the feature patterns were filtered or not. Without filtering, the determined clusters only contained two files, both of which were rated to be very not similar to the target song (target song: “Paddy’s Sicknote“ by “The Dubliners“, Songs in cluster: Marilyn Manson – “Tainted Love” and Frank Boeijen Groep – “Kronenburg Park”). Filtering the patterns caused 14 songs to be in the similarity cluster of the target song, none of which were rated to be similar. The user judged the similarity of “Paddy’s Sicknote” and it became obvious that only “Yellow Submarine” by “The Beatles” was considered to be slightly similar to the target song (it received a ranking of one).

Repeating the test runs on this small corpus with another query file did not reproduce these results. Instead, the results were comparable to the results of test runs on the other corpora.

The clustering process provided good results in structuring the input space into similarity clusters. In all test corpora, similarity clusters could be determined with regard to a specific target file. With regard to the medium corpora and the large corpus, 23.7 similar files could be found in each cluster in average. These files were

rated by a user to judge the subjective similarity between the files. The majority of the files in the similarity clusters were considered to be similar to the target file. In each similarity cluster, there was an average of 14.7 similar files. These files achieved at least a ranking of three on the 6-point scale. Hence, in average, 65.7% of the files in the retrieved similarity clusters are considered to be similar to the query file. Only a few files in the found similarity clusters were not considered similar by the user (average: 9 files, that is 34.3%); these files received a ranking of two or less.

More interestingly, most of the songs that were not enclosed in the similarity cluster of the query file (i.e. those songs that the system considers not similar to the query file) were not considered similar by the human user, too. Only 17% received a ranking of three or above and can be hence thought of as “forgotten” by the system.

Another important finding is the fact that the accuracy of the clustering process was very high in the large corpus. For instance, there were 24 songs in the similarity cluster for the song “Nothing else Matters” by Metallica, 75% of which received a ranking of at least three (18 files), and 54% received a ranking of four or higher (13 files). However, it is to note that a lot of songs that were present in this song's similarity cluster during other test runs (on one of the medium test corpora) were not included in the similarity cluster in the test run on the large corpus.

4 Discussion

Although the clustering accuracy of the suggested system is not as high as initially desired, clustering music files into similarity clusters is a good way to structure the search space (i.e. a very large music corpus). The suggested system presents itself as a capable tool to help structuring large musical databases to find music pieces according to the taste of a user or customer, or just structure the database. Nevertheless, there are a few things to note about the results from the previous section.

Filtering the feature patterns in every test run did generally not influence the clustering performance, which proves that features with a very small absolute value do not influence a classification and only play a minor role in the clustering process. These features can hence be omitted and do not need to be extracted from the source files. Therefore, the generic feature set that has been used to extract features from the songs can be designed more stream-lined, i.e. specifically for the task of feature extraction for similarity clustering using a Self-Organizing Map.

The clustering performance is directly connected to the amount of files in the underlying directory. The more files are in the source directory, the higher the clustering performance, regarding the number of retrieved clusters. This, of course, is not surprising since common sense suggests that the higher the number of songs that are included, the higher the diversity among those songs will be. This explains the poor clustering performance when using “Paddy's Sicknote” as the prototype song, as explained in the previous section. Since this song is a rather unusual one (Irish Gaelic Folk), it is unlikely that a similar song is found in a small cluster. In addition, it is to

note that the number of files falsely included into the similarity cluster of a target song is much lower in larger corpora than in smaller corpora. This can be explained with the fact that it is more likely to find truly similar files with regard to a certain target song in a larger corpus than in a smaller one. In the smaller corpus, the matches that are just “relatively” similar are included because these files are more similar to the target song than to any other song in the corpus – there are just no pieces that are more similar to the target song. However, much more similar songs might be present in the larger corpus. This also explains, why music pieces that were considered similar in a smaller cluster and hence included in a similarity cluster of a certain song are not included in the similarity cluster of the same song in a larger corpus (see the example of the song “Nothing else Matters” in the previous section). Songs, previously considered similar are simply more similar to another song than to the target song in the environment of a larger corpus.

Recapitulating it can be said that the very challenging task to structure large music databases in perceptually similar clusters and hence helping the user to find similar sounding songs more easily can be accomplished by the system at hand. The Self-Organizing Map plug-in for the Machine Learning Environment Yale provided good results in finding similarity clusters on different test corpora and is a competent framework to structure digital music databases automatically so user interaction on the same database is simplified as a database structuring disambiguates the data. Hence, the system can not only be used to help a user finding more music according to his/her personal taste, but it can be used by owners of very large music libraries to manage their databases. This might even be possible “on-the-fly” by extracting features of songs that are added to the library and feeding them into the clustering operator that has already been trained on the rest of the database. The suggested framework can also help in designing offers for customers of music online stores or Internet radio services. It is also imaginable that music pieces can be compared on a topological level, for example by music historians, to visualize musical correlation and similarity.

This can help to answer questions of plagiarism with regard to melody similarity.

Since the random sample of the user judgment was very small in this study, more thorough user surveys need to be conducted with regard to the similarity judgment in successive research to answer the question if the presented system is applicable in wider user communities.

Future work will also address the design of a more streamlined feature extraction method to provide a feature set that allows for more accurate clustering tasks. In addition, the presented system can be tested on a larger variety of music files. The system has mainly been tested on fairly modern and recent music but should be tested on older music and/or classic music. Future research should answer the question of the importance of corpus pre-structuring as well as genre distribution among the test files.

Another interesting future task is to develop a standalone system out of the present plug-in. A piece of software is desirable that works independently from Yale and is

able to provide home users or work groups with an intelligent structuring of their music archives. It is imaginable to include the possibility to share extracted features and store them in a centralized server, which will provide the opportunity to search remote databases for similar songs without violating copyright laws. Connecting online music stores with that database can make it possible for the users to quickly obtain legal copies of music files that are similar to one's private collection for a minimum of financial costs.

References

1. Allamanche, E., Herre, J., Hellmuth, O., Fröba, B., Kastner, T., Cremer, M.: Content-based identification of audio material using MPEG-7 low level description. In: ISMIR. Proceedings of the Second Annual International Symposium on Music Information Retrieval, pp. 197–204 (2001)
2. Fischer, S., Klinkenberg, R., Mierswa, I., Ritthoff, O.: YALE: Yet Another Learning Environment – Tutorial. No. CI-136/02, Collaborative Research Center 531, University of Dortmund, Dortmund, Germany (2002)
3. Hacker, S.: MP3, the definitive guide. O'Reilly & Associates, Inc. (2000)
4. Kohonen, T.: Self-Organized Formation of Topologically Correct Feature Maps. *Biological Cybernetics* 43, 59–69 (1982)
5. Kohonen, T.: *Self-Organizing Maps*. Springer, New York (2001)
6. Kurth, F., Clausen, M.: Full-Text Indexing of Very Large Audio Data Bases. 110th Audio Engineering Society Convention (2001)
7. Koza, J.R.: Genetic Programming. *Encyclopedia for Computer Science and Technology* (1997)
8. Liu, Z., Wang, Y., Chen, T.: Audio Feature Extraction and Analysis for Scene Segmentation and Classification. *Journal of VLSI Signal Processing* 20, 61–79 (1998)
9. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: YALE: Rapid Prototyping for Complex Data Mining Tasks. In: KDD 2006. Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, New York (2006)
10. Mierswa, I., Morik, K.: Automatic Feature Extraction for Classifying Audio Data. *Machine Learning Journal* 58, 127–149 (2005)
11. Mierswa, I.: Value Series Processing with Yale. Version 3.3
12. Pachet, F., Laigre, D.: A Naturalist Approach to Music File Name Analysis. In: Proceedings of 2nd International Symposium on Music Information Retrieval (2001)
13. Pardo, B., Shlfrin, J., Birmingham, W.: Name That Tune: A Pilot Study in Finding a Melody From a Sung Query. *Journal of the American Society for Information Science and Technology* 55(4) (2004)
14. Schedl, M., Pampalk, E., Widmer, G.: Intelligent Structuring and Exploration of Digital Music Collections. Austrian Research Institute for Artificial Intelligence (ÖFAI), Vienna, Austria and Department of Computational Perception Johannes Kepler Universität (JKU) Linz, Austria (2004)
15. Tzanetakis, G., Essl, G., Cook, P.: Automatic Musical Genre Classification Of Audio Signals. In: ISMIR. Proceedings of the Int. Symposium on Music Information Retrieval, pp. 205–210 (2001)

16. Uitdenbogerd, A.L., Zobel, J.: An Architecture for Effective Music Information Retrieval. *Journal of the American Society for Information Science and Technology* 55(12), 1053–1057 (2004)
17. Unal, E., Narayanan, S.S., Chew, E.: A Statistical Approach to Retrieval under User-dependent Uncertainty in Query-by-Humming Systems. In: *International Multimedia Conference, Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*, pp. 113–118. ACM Press, New York (2004)

Appendix: Similarity Cluster Examples

In the following, please find a number of tables with a choice of similarity clusters that have evolved from the clustering process. The artist names are printed in regular font, the song titles are bold. The value in parentheses denotes the similarity to the target song ranked on a 6-point scale. On the left hand side of the table, all songs in the cluster are shown. On the right hand side, all similar songs (at least a ranking of 3 or above) that have been “forgotten” by the framework are displayed. For reference, the tables also include the basic settings of the individual test run and an enumeration of the files not included in the similarity cluster.

Table 1. Similarity Cluster for “Nothing else Matters” by Metallica

<i>Similar Songs in Cluster (ranking):</i>	<i>Similar songs not in cluster (ranking):</i>
Band Ohne Namen Take my Heart (5)	Highland Bella Stella (5)
Die 3. Generation Ich will, dass Du mic liebst (5)	Echt Weinst Du (5)
Laura Immer wieder (5)	Reamonn Supergirl (4)
Die Ärzte Wie Es Geht (4)	Orange Blue She's Got That Light (4)
Sisqo Thong Song (4)	R Kelly If I could Turn back Hands (4)
Madonna Music (3)	Rednex Spirit of The Hawk (3)
Anastasia I'm Outa Love (3)	Santana Maria Maria (3)
Manu Chao Bongo Bong (3)	Madonna American Pie (3)
Music Instructor feat. Dean Super Fly (3)	Sting Desert Rose (3)
Gigi D'Agostino The Riddle (0)	Gabrielle Rise (3)
Limp Bizkit Take a Look Around (0)	
ATC My Heart Beats like a Drum (0)	Corpus Size : 100
Mauro Picotto Komodo (0)	Neurons : 15x15
Vanessa Amorosi Absolutely Everybody (0)	Filtering : No
Alex Ich will nur Dich (0)	Normalization : No
	Total # of Clusters : 7
	Similar files : 15
	Target Song:
	Metallica Nothing else Matters

Table 2. Similarity Cluster for “Nothing else Matters” by Metallica

<i>Similar Songs in Cluster (ranking):</i>	<i>Similar songs not in cluster (ranking):</i>
Elton John & George Michael Don't Let The Sun Go Down on Me (5)	Band Ohne Namen Take my Heart (5)
Garland Jeffreys Hail Hail Rock 'n Roll (5)	Highland Bella Stella (5)
Michael Jackson Heal the World (5)	Echt Weinst Du (5)
Youssou Ndour & Neneh Cherry 7 Seconds (5)	Die Ärzte Wie Es Geht (5)
Meat Loaf I'd Do Anything for Love (5)	Die 3. Generation Ich will, dass Du mich liebst (5)
Young Deenay Walk On By (5)	Laura Immer wieder (5)
Thomas D Liebesbrief (5)	Reamonn Supergirl (4)
Christina Aguilera Beautiful (5)	Orange Blue She's Got That Light (4)
Coldplay Speed Of Sound (5)	R Kelly If I could Turn back Hands (4)
Hypertraxx The Darkside (4)	Sisqo Thong Song (3)
Nelly Furtado I'm Like a Bird (4)	Rednex Spirit of The Hawk (3)
Atomic Kitten It's OK (4)	Santana Maria Maria (3)
Daniel Bedingfield If You're Not The One (4)	Madonna Music (3)
Nomad I wanna give you devotion (4)	Madonna American Pie (3)
Salt 'n Pepa Lets Talk about Sex (3)	Sting Desert Rose (3)
Ace of Base Don't Turn Around (3)	Gabrielle Rise (3)
Dune Hardcore Vibes (3)	Anastasia I'm Outa Love (3)
Chris Brown Run It (3)	Manu Chao Bongo Bong (3)
Dru Hill How Deep is Your Love (2)	Music Instructor feat. Dean Super Fly (3)
J-Kwon Tipsy (2)	Corpus Size : 1,554
Color Me Badd I Wanna Sex You Up (1)	Neurons : 15x15
Cher Believe (1)	Filtering : No
Interactive Living Without Your Love (0)	Normalization : No
Wolfgang Petry Die Längste Single (0)	Total # of Clusters : 143
	Similar files : 24
	Target Song: Metallica Nothing else Matters