

Hierarchical Organization and Description of Music Collections at the Artist Level

Elias Pampalk¹, Arthur Flexer¹, and Gerhard Widmer^{1,2}

¹ Austrian Research Institute for Artificial Intelligence, Vienna, Austria

² Johannes Kepler University Linz, Austria

{elias,arthur,gerhard}@oefai.at

Abstract. As digital music collections grow the necessity to organize them (preferably automatically) increases. In this paper we present an approach to hierarchically structure artists automatically according to their similarity and to describe these clusters with words. The approach is based on a web search engine (Google), standard text retrieval techniques, term selection (using domain knowledge), and hierarchical clustering with fuzzy boundaries. We present the results and discuss shortcomings of different variations.

1 Introduction

The value of a large music collection is limited by how efficiently a user can explore it. Portable audio players which can store over 20,000 songs and on-line music shops with more than 1 million tracks in their repositories are not uncommon. Thus, simple artist/album based organizations are insufficient.

The probably best known approach to organize music is to classify it into genres such as pop, rock, classical etc. Experts are able to name several sub-genres for each genre. An example is Ishkur's guide to electronic music with 180 subgenres within electronic music alone.³ However, there are several reasons why classifying music into genres is not the perfect solution (for a detailed discussion see e.g. [1]). One of the main issues is that it is a very cumbersome task and in many cases requires an expert. Furthermore, large genre taxonomies tend to be inconsistent and have rather fuzzy borders.

One interesting alternative is to use some concept of similarity. A possible approach to browse a collection with such information is to start with familiar artists and search for related artists. A well known example is Amazon's recommendations.

In this paper we rely on the content of webpages ranked by Google to compute the similarity of artists. We investigate how this similarity concept can be used to automatically organize artists into overlapping hierarchical clusters. Furthermore, we investigate the advantages and disadvantages of different strategies to automatically describe these clusters with words. To demonstrate our approach we implemented an HTML interface which the interested reader might want to explore.⁴

³ <http://www.di.fm/edmguide>

⁴ <http://www.oefai.at/~elias/wa>

The field of music information retrieval is relatively young. However, there are a number of approaches related to extracting information from the web and approaches to structure and describing music collections.

One of the first approaches using web-based data to compute artist similarities was presented by Pachet et al. [2]. In particular, co-occurrences on playlists from radio stations and compilation CD databases were used to cluster a set of 12 songs and a set of 100 artists hierarchically according to similarity. The approach was demonstrated using single-linkage agglomerative clustering. The main difference of the approach we present in this paper is that we aim at automatically describing the *contents* of these clusters. We also used a larger set of artists, a different data source and a different clustering technique.

Another source are common webpages [3, 4]. The main idea is to retrieve top ranked sites from Google queries and apply standard text-processing techniques. Using the obtained word lists, the artist similarities are computed. A drastically simplified approach is to use the number of pages found by Google for a query containing two artist names [5, 6]. As the evaluation of artist similarity is quite difficult [7] it is tempting to evaluate the similarity measures in a genre classification scenario [8]. Other web-based sources include expert opinions (such as those available from the All Music Guide⁵), album reviews [9], or song lyrics [10].

Related work in terms of structuring and describing music collections includes the Islands of Music approach which is based on audio signal analysis [11, 12]. The idea is to organize music collections on a map such that similar pieces are located close to each other. The clusters are visualized using a metaphor of geographic maps and are labeled with abstract terms describing low-level audio signal characteristics. Furthermore, “weather charts” are used to describe the value of one attribute (e.g. bass energy) across the different regions of the map. A hierarchical extension was presented in [13].

Obviously a combination of audio-based and web-based sources is very desirable; however, that is outside of the scope of this paper. First approaches demonstrating the advantages of the combination can be found in [14, 9, 15].

In the following sections we describe the three main steps of our approach. The first step is to compute the similarity between the artists. Subsequently, we cluster the artists hierarchically. Finally, we describe each cluster with words.

2 Similarity Computations

In this section we describe how we compute the similarity between two artists. This approach is a simplified version of the approach originally presented in [3] and is based on standard text information retrieval techniques. In previous work [8] we applied this approach successfully to classify artists into genres.

For each artist a query string consisting of the artist’s name as an exact phrase extended by the keywords *+music +review* is sent to Google using Google’s SOAP interface.⁶ Google offers this service free of charge but limits

⁵ <http://www.allmusic.com>

⁶ <http://www.google.com/apis>

the number of queries to 1000 a day per registered user. Each query returns 10 pages. To retrieve 50 pages for each of the 224 artists⁷ (as we used for our experiments) thus requires two days.

We remove all HTML markup tags, taking only the plain text content into account. Using a stop word list we remove very frequent and unwanted terms.⁸ For each artist we create a list of the words that occur in the 50 pages. For computational reasons for each artist we remove those words which only occur in less than 3 (6%) of the pages. This removes a large number of typos and drastically improves the speed of the following steps.

In addition we remove each term for which no artist exists for whom the term occurs in at least 10 (20%) of the webpages. In our experiment with 224 artists 4139 terms remained. The data is inherently extremely high-dimensional as the first 200 eigenvalues (using an eigenvalue decomposition, also known as PCA) are needed to describe 95% of the variance.

For each artist a and each remaining term t appearing in the retrieved pages, we count the number of occurrences tf_{ta} (term frequency) of term t in documents relating to a . Furthermore, we count df_t the number of pages the term occurred in (document frequency). These are combined using the term frequency \times inverse document frequency ($tf \times idf$) function (we use the *ltc* variant [16]). The term weight per artist is computed as,

$$w_{ta} = \begin{cases} (1 + \log_2 tf_{ta}) \log_2 \frac{N}{df_t}, & \text{if } tf_{ta} > 0, \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where N is the total number of pages retrieved.

This gives us a vector of term weights for each artist. We normalize the weights such that the length of the vector equals 1 (Cosine normalization). This removes the influence that the length of the retrieved webpages would otherwise have. Finally, the distance between two artists is computed as the Euclidean distance of the normalized term weight vectors.

The evaluation of this approach within a genre classification context can be found in [8]. For the set of 224 artists (from 14 genres) which we use in our experiments we get accuracies of 85% for leave-one-out evaluation using k -nearest neighbor (with $k=1$). This is significantly higher than the accuracies which can be achieved with state of the art audio-based similarity measures.

3 Hierarchical Clustering

The clustering is closely linked to the user interface. We attempt to demonstrate that our approach can work on very limited displays such as mobile phones or portable audio players. In order not to exclude such displays from the start we limit ourselves to simple one-dimensional lists of items as the main information

⁷ A complete list of the used artists can be found at:
<http://www.cp.jku.at/people/knees/artistlist224.html>

⁸ <http://www.oefai.at/~elias/wa/stopwords.txt>

carrier. Furthermore, we assume that there is a certain number of items which can be best displayed (we arbitrarily choose 5, which is significantly less than the 14 genres present in the collection we use for our experiments). As a consequence we put aside techniques which try to find the optimum number of clusters as well as approaches which rely on a two-dimensional display. In previous work, where we used the same one-dimensional clustering approach (and a similar user interface) to organize large drum sample libraries, we got very positive user feedback [17].

Basically, we use a one-dimensional self organizing map (SOM) [18] which is structured hierarchically [19, 20]. More flexible approaches include the growing hierarchical self-organizing map [21]. Other alternatives include, for example, hierarchical agglomerative clustering (as used in [2]).

The SOM groups similar items into clusters and places similar clusters close to each other. In our implementation, after we train the SOM, we overlap the clusters. In particular, we increase the cluster size by 20% adding the artists closest to the border. Recursively, for each cluster found we train another one-dimensional SOM (for all artists assigned to the cluster) until the cluster size falls below a certain limit.

Figure 1 shows the distribution of the genres within the nodes (i.e. clusters) in the hierarchical structure. (A screenshot of how the hierarchy is displayed in the user interface can be found in Figure 3.) At the first level classical music (node n1) is well separated from all other music. The effects of the overlap are immediately visible as the sum of artists mapped to all units in the first layer is beyond 224. One example for a direct effect of the overlap is that there is jazz music in node n1, which would not be there otherwise. We can also see that n1 and n5 are the only nodes at the first level that contain jazz music. The nodes n2 and n3 are the only ones which contain electronic and rap/hip-hop. Blues can only be found in n4 and n5.

At the second level we can see that most nodes have specialized. For example, n5.1 contains 34 artists mainly from jazz and blues and few from rock & roll. Another nice example is n3.2 which contains mostly punk but also some alternative. Self-critically, we must admit that the one-dimensional ordering of the SOM (i.e. similar units should be close to each other) is not apparent. One reason for this might be the extremely high-dimensional data (as mentioned previously, 200 eigenvectors are necessary to preserve 95% of the variance in the data). Another not so welcome observation is that there are some nodes which seem to contain a bit of almost every genre.

Figure 2 shows what happens at the third level (in the subbranch of the node n2). For example, while the node n2.3 contains artists from punk and soul/R&B none of its children mix the two. Another positive example is that node n2.5.1 captures most of the electronic music in n2.5. However, as can be seen the clustering is far from perfect and leaves a lot of room for improvement.

To truly evaluate different clustering techniques would require a series of expensive user studies. In this paper we will use illustrative examples instead. Furthermore, the main limitation of any clustering approach is always the quality of the similarity measure. While any clustering approach would produce very

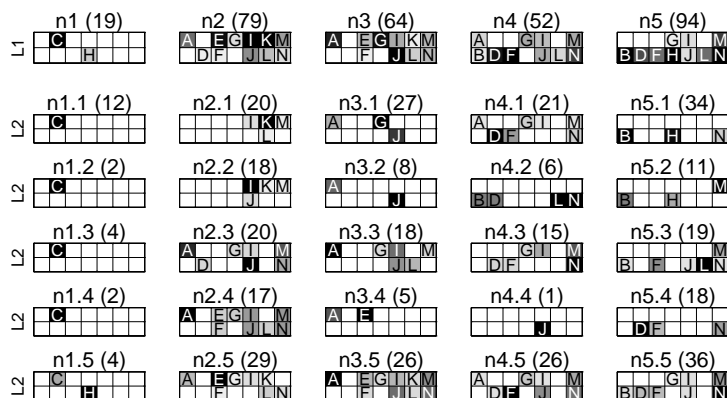


Fig. 1. Distribution of the 14 genres in the nodes of the first level (L1, first row) and second level (L2, all columns starting with the second row). For example, n2.4 (L2 node) is the fourth child of parent n2 (L1 node). Each subplot represents the distribution of genres in a node (visualized as histogram and displayed in two lines to save space). Black corresponds to high values. The boxes in the histogram corresponds to the following genres: A alternative/indie, B blues, C classic, D country, E electronic, F folk, G heavy, H jazz, I pop, J punk, K rap/hip-hop, L reggae, M R&B/soul, N rock & roll. The numbers in brackets are the number of artists mapped to the node.

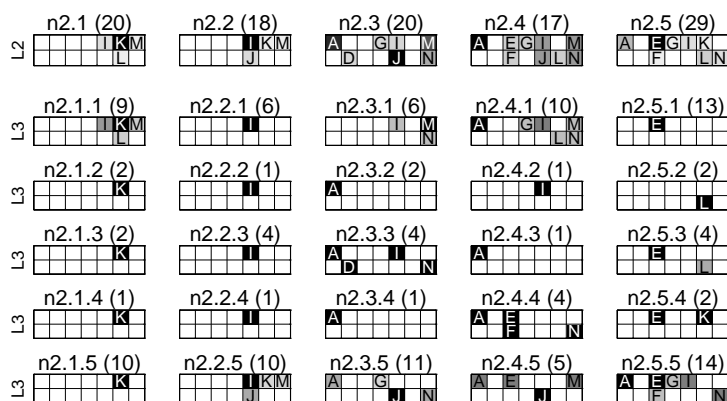


Fig. 2. Distribution of the 14 genres in the nodes of the second (L2) and third (L3) level in the subbranch of node n2.

interesting results given a perfect artist similarity measure, no approach can produce interesting results if the artist similarity measure fails.

4 Term Selection for Cluster Description

The term selection is a core part of the user interface. The goal is to select those words which best summarize a group of artists. There are 3 main assumptions

which we build upon. First, the artists are mostly unknown to the user (otherwise we could just label the nodes with the artists’ names). Second, we also do not know which artists the user knows (otherwise we could use those to describe the nodes). Third, we assume that space is limited and thus we want to describe each node with as few words as possible. Obviously dropping assumption 2 could lead to very interesting interactive user interfaces. However, this is beyond the scope of this paper.

In our experiments we compare 5 different term selection approaches, and two different concepts regarding the set of terms to start with in the first place. In particular, we suggest using a domain-specific dictionary instead of the terms used to compute the similarity.

4.1 Techniques

First we describe the term selection techniques. Given are the term frequency tf_{ta} , document frequency df_{ta} , and the Cosine normalized $tf \times idf$ weight w_{ta} for each term t and artist a . Furthermore, we know the artists assigned to it and its children nodes.

A straightforward approach is to use the $tf \times idf$ computations, i.e. w_{ta} . For each node we compute the average over the cluster c of the assigned artists w_{tc} and select the terms with the highest values.

The second approach is called “LabelSOM” [22] and has successfully been applied to label large document collections organized by SOMs. LabelSOM is built on the observation that terms with a very high w_{tc} and a high variance (i.e., they occur very rare in some of the documents in the cluster) are usually poor descriptors. The suggested solution is to compute the variance of w_{ta} for each term of the mapped artists and use it to rank the terms (better terms have lower variances). Since terms which do not occur in c ($w_{tc} = 0$) have variance 0, terms with w_{tc} below a manually defined threshold are removed from the list of possible candidates. This threshold depends on the number of input dimensions and how the vectors are normalized. For the 4139 dimensions we used a threshold of 0.045. For the approach with the dictionary (see below) where we have 1269 dimensions we used a threshold of 0.1. Note that the variance in a cluster consisting only of one artist is meaningless. In such cases we use $tf \times idf$ ranking instead.

Neither $tf \times idf$ ranking nor the LabelSOM technique try to find terms which discriminate two nodes. However, emphasizing differences between nodes of the same parent helps reduce redundancies in the descriptions. Furthermore, we can assume that the user already know what the children have in common after reading the description of the parent. A standard technique to select discriminative terms is the χ^2 (chi-square) test (e.g. [23]). The χ^2 -value measures the independence of t from group c and is computed as,

$$\chi_{tc}^2 = \frac{N(AD - BC)^2}{(A + B)(A + C)(B + D)(C + D)} \quad (2)$$

where A is the number of documents in c which contain t , B the number of documents not in c which contain t , C the number of documents in c without

t , D the number of documents not in c without t , and N is the total number of retrieved documents. As N is equal for all terms, it can be ignored. The terms with highest χ_{tc}^2 values are selected because they are least independent from c . Note that the document frequency is very informative in our case because df_{ta} describes the percentage of times the terms occur in the 50 retrieved documents per artist.

The fourth approach was proposed by Lagus and Kaski (LK) [24]. Like the LabelSOM it has been developed to label large document collections organized by SOMs. While χ^2 only uses df , LK only use tf . The heuristically motivated ranking formula (higher values are better) is,

$$f_{tc} = (tf_{tc} / \sum_{t'} tf_{t'c}) \cdot \frac{(tf_{tc} / \sum_{t'} tf_{t'c})}{\sum_{c'} (tf_{tc'} / \sum_{t'} tf_{t'c'})}, \quad (3)$$

where tf_{tc} is the average term frequency in cluster c . The left side of the product is the importance of t in c defined through the frequency of t relative to the frequency of other terms in c . The right side is the importance of t in c relative to the importance of t in all other clusters.

The fifth approach is a variation of LK. We implemented it to demonstrate the effects of extreme discrimination. In particular, in this variation tf_{tc} are normalized over the whole collection such that a word which occurs 100 times in cluster c and never in any cluster is equally important to a word that occurs once in c and never otherwise. Obviously this approach can only produce meaningful results when using a specialized dictionary. We ignore all terms which do not occur in at least 10% of the documents per cluster. The ranking function (higher values are better) is,

$$f_{tc} = (tf_{tc} / \sum_{c'} tf_{tc'}) \cdot \frac{(tf_{tc} / \sum_{c'} tf_{tc'})}{\sum_{c''} (tf_{tc''} / \sum_{c'} tf_{tc'})}. \quad (4)$$

In addition we implemented 2 combinations. In particular combining LK with χ^2 , and the LK variant with χ^2 . In both cases the values were combined by multiplication.

4.2 Domain-Specific Dictionary

One of the main pillars of our approach is the use of a dictionary to avoid describing clusters with artist, album, and other specialized words likely to be unknown to the user. This dictionary contains general words used to describe music, such as genre names. The dictionary contains 1398 entries,⁹ 1269 of these actually occur in the retrieved documents. The dictionary was manually compiled by the authors in a sloppy manner by copying lists from various sources such as Wikipedia, the Yahoo directory, allmusic.com, and other sources which contained music genres (and subgenres), instruments, or adjectives. The dictionary is far from complete. On the other hand, it would not hurt to clean it up a bit. For

⁹ <http://www.oefai.at/~elias/wa/dict.txt>

example, (as we will see later) the terms: *world*, *uk*, *band*, and *song* should be removed. However, to get a better understanding of possible drawbacks we did not modify the dictionary.

We parse each retrieved webpage and compute the term frequencies, document frequencies, and $tf \times idf$. So why did we not use the dictionary to compute the similarities? There are 2 reasons.

First, the classification performance using k -nearest neighbors with leave-one-out validation is only about 79% compared to the 85% of the standard approach. The explanation for this is that the standard approach captures a lot of the very specific words such as the artists names, names of their albums and many other terms which co-occur on related artist pages.

Second, while the dictionary is an important pillar of our approach we try not to rely too much upon it. By manipulating the dictionary it is very likely that we could achieve 100% classification accuracies on our set of 224 artists. However, it would be impossible to generalize our findings to other music collections. Furthermore, in our current approach the specialized dictionary can be replaced at any time without impact on the hierarchical structure.

5 Results and Discussion

In this section we first describe the user interface we implemented to demonstrate the approach. Second, we compare differences between different term selection approaches. Due to space limitations we will use simple lists of words to do so (see Table 1). Finally, we discuss our approach in general.

5.1 User Interface

To demonstrate our approach we implemented a very simple HTML interface.¹⁰ There are two main parts of the interface: the hierarchy of clusters visualized as a grid of boxed texts and, just to the right of it, a display of a list of artists mapped to the currently selected cluster. The clusters of the first level in the hierarchy are visualized using the five boxes in the first (top) row. After the user selects a cluster, a second row appears which displays the children of the selected cluster. The selected clusters are highlighted in a different color. The hierarchy is displayed in such a way that the user can always see every previously made decision on a higher level. The number of artists mapped to a cluster is visualized by a bar next to the cluster. Inside a text box, at most the top 10 terms are displayed. However, if a terms value is below 10% of the highest value then it is not displayed. The value of the ranking function for each term is coded through the color in which the term is displayed. The best term is always black and as the values decrease the color fades out. For debugging purposes it is also possible to display the list of all ranked LK words for a cluster. Figure 3 shows what the user interface looks like (using LK labeling) after node n2.5.1 was selected (thus 4 levels are visible).

¹⁰ <http://www.oefai.at/~elias/wa>

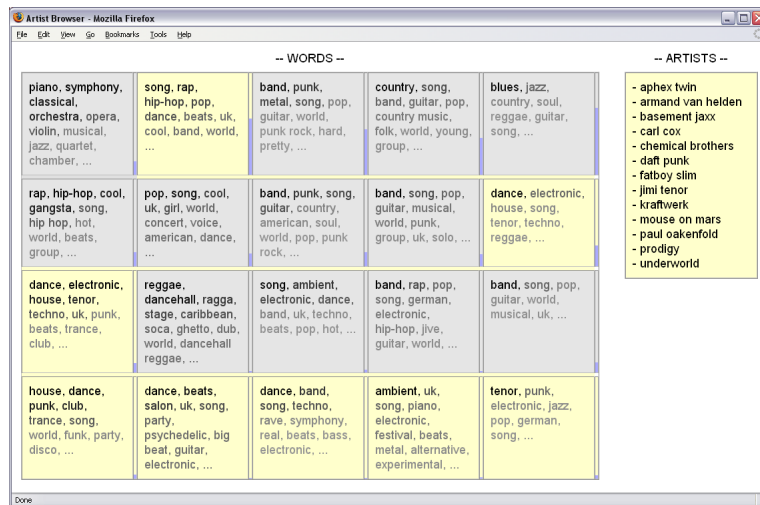


Fig. 3. Screenshot of the HTML user interface.

5.2 Comparison of Term Selection

Table 1 lists all top-ranked words for the different approaches at the first level and some examples of the second level (for the children of node n5.1). Comparing this table with Figures 1 and 2 shows how different types of genres are described. For example, in most cases describing the classical cluster (node n1) works very well. In the next paragraphs we describe some of the observations we made.

First, the results using the dictionary are better in most cases at the first level. The main difference becomes clearer at the second level (for the children of node n5). In particular, using the dictionary avoids the frequent appearance of artist names.

Second, not all words in the domain specific dictionary make sense. Although not directly noticeable at the first level there are some words which appear frequently in the top-ranked words but do not convey much information: *world*, *uk*, *band*, *song*, *musical*. On the other hand, from studying the lists we noticed that words such as *love* and *hate* are missing in our dictionary. Having a few meaningless words is not such a big problem. In an interactive interface the user could just click on the words to remove and the interface could be updated immediately. However, adding missing words to the dictionary is a bit more complex and requires scanning all the retrieved documents for occurrences.

Third, the performance of the non discriminating approaches ($tf \times idf$, Label-SOM) is very poor. On the other hand, all discriminative approaches (χ^2 , LK, and combinations) yield interesting results with the dictionary. However, the LK variant by itself focuses too much on the differences. Obviously, to truly judge the quality of the different variations would require user studies. Subjectively our impression was that the approach from Lagus & Kaski performed slightly better than the others.

	with dictionary	without dictionary
$tf \times idf$		
n1	classical, piano, orchestra, symphony, musical	classical, piano, orchestra, works, composer
n2	song, pop, world, uk, band	listen, color, news, pop, size
n3	song, band, pop, world, guitar	band, listen, great, pop, live
n4	song, band, guitar, pop, world	color, listen, live, pop, size
n5	song, blues, band, guitar, world	color, size, family, listen, blues
LabelSOM		
n1	world, musical, concert, song, uk	two, information, musical, recordings, great
n2	world, musical, classical, song, real	content, know, people, listen, sound
n3	musical, world, song, group, pop	great, sound, news, listen, live
n4	musical, world, classical, song, pop	content, great, information, pop, listen
n5	musical, classical, group, song, world	information, great, content, listen, pop
χ^2		
n1	piano, orchestra, symphony, concert, opera	classical, composer, musical, great, piano
n2	dance, rap, hip-hop, beats, group	news, pop, sound, track, release
n3	guitar, musical, group, punk, metal	band, sound, live, great, pop
n4	musical, guitar, country, group, blues	live, band, pop, news, policy
n5	blues, band, country, pop, jazz	blues, jazz, country, hits, policy
Lagus & Kaski		
n1	piano, symphony, classical, orchestra, opera	op, bach, piano, symphony, classical
n2	song, rap, hip-hop, pop, dance	hop, hip, rap, listen, pop
n3	band, punk, metal, song, pop	band, punk, metal, bands, great
n4	country, song, band, guitar, pop	country, alice, elvis, brooks, rate
n5	blues, jazz, country, soul, reggae	blues, jazz, color, john, size
$\chi^2 \cdot LK$		
n1	piano, orchestra, symphony, opera, violin	classical, piano, composer, orchestra, symphony
n2	rap, dance, hip-hop, beats, uk	news, pop, hop, hip, track
n3	punk, guitar, metal, musical, group	band, punk, live, sound, great
n4	country, guitar, musical, group, blues	country, live, band, pop, hits
n5	blues, jazz, country, band, soul	blues, jazz, country, john, hits
Lagus & Kaski variant		
n1	rondo, fortepiano, contralto, fugue, mezzo	nabucco, leopold, cycles, figaro, sonatas
n2	hardcore techno, latin rap,	pies, grandmaster, hash, tricky, pimp
n3	southern rap, east coast rap	
n4	pop-metal, melodic metal, detroit rock,	roisin, pies, hash, dez, voulez
n5	flamenco guitar, math rock	
n4	new traditionalist, british folk-rock,	csn, dez, voulez, shapes, daltrey
n5	progressive bluegrass, gabba, slowcore	
n5	rockabilly revival, new beat, progressive	hodges, precious, shanty, broonzy, dez
n5	country, vocalion, freakbeat	
$\chi^2 \cdot LK$ variant		
n1	piano, orchestra, symphony, opera, violin	classical, symphony, composer, piano, orchestra
n2	rap, hip-hop, beats, dance, cool	hop, hip, rap, eminem, dj
n3	punk, metal, guitar, punk rock, hard	band, punk, metal, bands, live
n4	country, guitar, country music, folk, group	country, brooks, elvis, dylan, hits
n5	blues, jazz, country, soul	blues, jazz, willie, otis, john
Lagus & Kaski		
n5.1	blues, jazz, guitar, band, orchestra	blues, jazz, john, coltrane, basie
n5.2	soul, blues, song, gospel, pop	aretha, soul, redding, king, franklin
n5.3	reggae, ska, song, world, dancehall	marley, reggae, tosh, cliff, baez
n5.4	country, country music, song, bluegrass, folk	country, hank, elvis, cash, kenny
n5.5	band, song, pop, guitar, blues	elvis, roll, rate, band, bo
LK variant (with dictionary)		
n5.1	hot jazz, post-bop, vocalion, rondo, soul-jazz, classic jazz, hard bop, superstitious, octet	
n5.2	british blues, pornographic, colored, classic soul, sensual, erotic, precious, rap rock, stylish	
n5.3	vocal house, soca, british punk, gong, ragga, ska, dancehall, dancehall reggae, hard house	
n5.4	new traditionalist, yodelling, middle aged, country boogie, outlaw country, rockabilly revival	
n5.5	experimental rock, boogie rock, castanets, psychedelic pop, pagan, dream pop, crunchy	

Table 1. List of top ranked terms for selected nodes.

5.3 Discussion

One of the main problems is that our approach relies on the artist names. In many cases this name might have several meanings making it difficult to retrieve relevant webpages. Another problem is that many new and not so well known artist do not appear on webpages. This limits our approach to yesterdays mainstream western culture. This limitation is also underlined by the dictionary we use which contains terms mainly used in our culture. However, the dictionary could easily be replaced. Another issue is the dynamics of web contents (e.g. [25]). We studied this in [8] and the study was continued in [26]. So far we observed significant changes in the Google ranks, but these did not have a significant impact on the similarity measure.

6 Conclusions

In this paper we demonstrate possibilities to hierarchically organize music at the artist level. In particular we suggest using fuzzy hierarchical structures and a domain-specific dictionary to describe the clusters. The results are very promising, however, we fail to present a thorough evaluation. In future work we plan to conduct small scale user studies and combine this approach with other approaches based on audio signal analysis.

7 Acknowledgments

This research was supported by the EU project SIMAC (FP6-507142). The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry for Education, Science, and Culture and by the Austrian Federal Ministry for Transport, Innovation, and Technology.

References

1. Pachet, F., Cazaly, D.: A taxonomy of musical genres. In: Proc of RIAO Content-Based Multimedia Information Access. (2000)
2. Pachet, F., Westerman, G., Laigre, D.: Musical data mining for electronic music distribution. In: Proc of WedelMusic Conf. (2001)
3. Whitman, B., Lawrence, S.: Inferring descriptions and similarity for music from community metadata. In: Proc of Intl Computer Music Conf. (2002)
4. Baumann, S., Hummel, O.: Using cultural metadata for artist recommendation. In: Proc of WedelMusic Conf. (2003)
5. Zadel, M., Fujinaga, I.: Web services for music information retrieval. In: Proc of Intl Conf on Music Information Retrieval. (2004)
6. Schedl, M., Knees, P., Widmer, G.: A web-based approach to assessing artist similarity using co-occurrences. In: Proc of Workshop on Content-Based Multimedia Indexing. (2005) submitted.
7. Ellis, D., Whitman, B., Berenzweig, A., Lawrence, S.: The quest for ground truth in musical artist similarity. In: Proc of Intl Conf on Music Information Retrieval (2002)

8. Knees, P., Pampalk, E., Widmer, G.: Artist classification with web-based data. In: Proc of Intl Conf on Music Information Retrieval (2004)
9. Whitman, B., Ellis, D.: Automatic record reviews. In: Proc of Intl Conf on Music Information Retrieval (2004)
10. Logan, B., Kositsky, A., Moreno, P.: Semantic analysis of song lyrics. In: Proc of IEEE Intl Conf on Multimedia and Expo (2004)
11. Pampalk, E.: Islands of Music: Analysis, Organization, and Visualization of Music Archives. Master's thesis, Vienna University of Technology (2001)
12. Pampalk, E., Rauber, A., Merkl, D.: Content-based organization and visualization of music archives. In: Proc of the ACM Multimedia (2002)
13. Rauber, A., Pampalk, E., Merkl, D.: Using psycho-acoustic models and self-organizing maps to create a hierarchical structuring of music by sound similarities. In: Proc of Intl Conf on Music Information Retrieval (2002)
14. Whitman, B., Smaragdis, P.: Combining musical and cultural features for intelligent style detection. In: Proc of Intl Conf on Music Information Retrieval (2002)
15. Baumann, S., Pohle, T., Shankar, V.: Towards a socio-cultural compatibility of MIR systems. In: Proc of Intl Conf on Music Information Retrieval (2004)
16. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. *Information Processing and Management* **24** (1988)
17. Pampalk, E., Hlavac, P., Herrera, P.: Hierarchical organization and visualization of drum sample libraries. In: Proc of Intl Conf on Digital Audio Effects (2004)
18. Kohonen, T.: *Self-Organizing Maps*. Springer (2001)
19. Miikkulainen, R.: Script recognition with hierarchical feature maps. *Connection Science* (1990)
20. Koikkalainen, P., Oja, E.: Self-organizing hierarchical feature maps. In: Proc of the Intl Joint Conf on Neural Networks (1990)
21. Dittenbach, M., Merkl, D., Rauber, A.: The Growing Hierarchical Self-Organizing Map. In Proc of Intl Joint Conf on Neural Networks (2000)
22. Rauber, A.: LabelSOM: On the Labeling of Self-Organizing Maps. In: Proc of the Intl Joint Conf on Neural Networks (1999)
23. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In Proc of Intl Conf on Machine Learning (1997)
24. Lagus, K., Kaski, S.: Keyword selection method for characterizing text document maps. In: Proc of Intl Conf on Artificial Neural Networks (1999)
25. Lawrence, S., Giles, C. L.: Accessibility of Information on the Web. In: *Nature* (1999)
26. Knees, P.: Automatic Classification of Musical Artists based on Web-Data (in German). Master's thesis, Vienna University of Technology (2004)