# Hierarchical Organization and Description of Music Collections at the Artist Level

Elias Pampalk[1], Arthur Flexer[1,2], and Gerhard Widmer[1,3]

[1] Austrian Research Institute for Artificial Intelligence, Vienna, Austria
[2] Department of Medical Cybernetics and Artificial Intelligence, Center for Brain Research, Vienna University of Medicine, Austria
[3] Department of Computational Perception, Johannes Kepler University Linz, Austria
{elias,arthur,gerhard}@ofai.at

**Abstract.** As digital music collections grow, so does the need to organizing them automatically. In this paper we present an approach to hierarchically organize music collections at the artist level. Artists are grouped according to similarity which is computed using a web search engine and standard text retrieval techniques. The groups are described by words found on the webpages using term selection techniques and domain knowledge. We compare different term selection techniques, present a simple demonstration, and discuss our findings.

## 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 online 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 subgenres for each genre. An example is Ishkur's guide to electronic music with 180 subgenres within electronic music alone.[4] However, 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 boundaries between genres. An alternative is to use some form of similarity defined directly between artists. This allows a collection to be browsed starting with an familiar artist and exploring related ones. Successful examples include Amazon's recommendations.

In this paper we analyze the content of webpages ranked by Google to compute the similarity of artists. We investigate how this similarity can be used to automatically organize artists into overlapping hierarchical clusters. Furthermore, we investigate the advantages and disadvantages of different strategies to

[4] http://www.di.fm/edmguide

automatically describe these clusters with words. An HTML-based demonstration is available.[5]

The following sections are: (2) related work, (3) similarity computations, (4) hierarchical clustering, (5) term selection for clusters description, (6) results and discussion (including the user interface), (7) conclusions.

## 2 Related Work

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

One of the first approaches using web-based data to compute artist similarities was presented in [2]. 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 focus on 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 resort to a genre classification scenario [8]. Other web-based sources include expert opinions (such as those available from the All Music Guide[6]), album reviews [9], or song lyrics [10].

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, for example, in [11, 9, 12].

Related work in terms of structuring and describing music collections includes the Islands of Music approach which is based on audio signal analysis [13, 14]. 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. Hierarchical extensions were presented in [15, 16].

The same hierarchical approach we use for this work we previously applied to organize large collections of drum sample libraries [17]. The basic idea is similar to the approach used by the search engine vivisimo.[7]

[5] http://www.oefai.at/~elias/wa
[6] http://www.allmusic.com
[7] http://www.vivisimo.com

## 3 Similarity Computations

In this section we describe how we compute the similarity between 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.[8] For each artist the 50 top ranked pages are retrieved. 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.[9]

Let the term frequency $tf_{ta}$ be the number of occurrences (frequency) of word (term) $t$ in the pages retrieved for artist $a$. Let the document frequency $df_{ta}$ be the number of webpages (documents) for $a$ in which $t$ occurs at least once, and let $df_t = \sum_a df_{ta}$.

First, for computational reasons, for each artist we remove all terms for which $df_{ta} < 3$. (The highest possible value for $df_{ta}$ is 50). Then we merge all individual term lists into one global list. From this we remove all terms for which there is no $df_{ta} \geq 10$. (That is, for at least one artist the term must occur in at least 10 pages.)

In our experiment with 224 artists 4139 terms remained.[10] 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.

The frequency lists are combined using the term frequency $\times$ inverse document frequency ($tf \times idf$) function (we use the *ltc* variant [18]). The term weight per artist is computed as,

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

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) to remove the influence the document length 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 (manually assigned to 14 genres) which we use in our experiments we get accuracies of 85% for leave-one-out evaluation using a nearest neighbor classifier.

---

[8] This service is offered free of charge but is limited to 1000 queries a day per registered user. Each query returns 10 pages. http://www.google.com/apis

[9] http://www.oefai.at/~elias/wa/stopwords.txt

[10] A list of the artists is available from: http://www.cp.jku.at/people/knees/artistlist224.html

## 4 Hierarchical Clustering

There is a close link between clustering and the user interface. For the user interface we use simple lists of items instead of complex graphical visualizations to convey the information. We assume that there is a certain number of items on the list which can be displayed best. We arbitrarily choose 5, which is significantly less than the 14 genres present in the collection we use for our experiments.

A clustering technique suitable to our requirements is the one-dimensional self organizing map (SOM) [19] which can be structured hierarchically [20, 21]. More flexible approaches include the growing hierarchical self-organizing map [22]. 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. After training the SOM we increase the cluster size by 20% adding the artists closest to the border. This overlap makes it easier to find artists which are on the border of two or more clusters. Recursively, for each cluster another one-dimensional SOM is trained (for all artists assigned to the cluster) until the cluster size falls below a certain limit (e.g. only 7 artists remaining). In previous work we used the same one-dimensional overlapping clustering approach and a similar user interface to organize large drum sample libraries. The user feedback we got was very positive [17].

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 is shown 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. The nodes n1 and n5 are the only ones at the first level containing jazz music. Electronic and rap/hip-hop is only contained in n2 and n3, and blues only in n4 and n5.

At the second level 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. An interesting observation is 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 observation indicating that the approach is not flawless 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 node n2). For example, while 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.

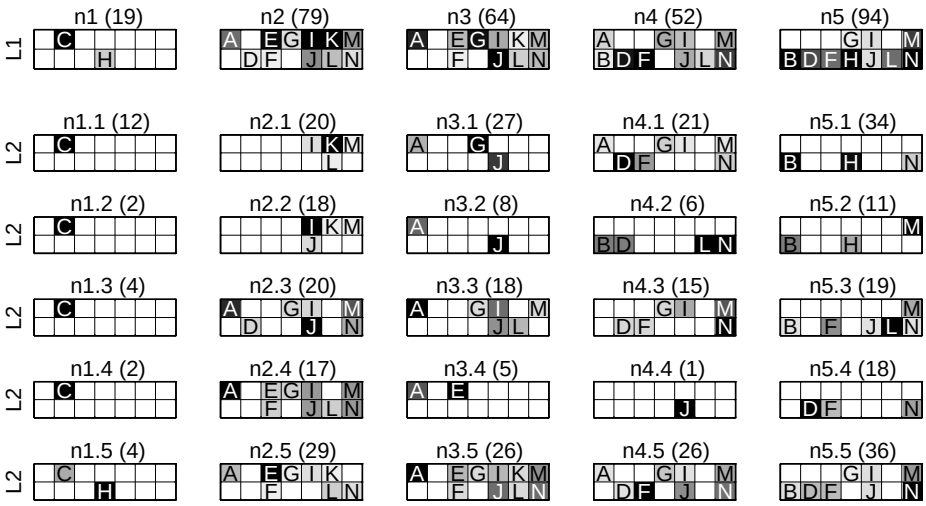


**Fig. 1.** Distribution of the 14 genres in the nodes of the first level (L1, first row) and second level (L2, all columns starting with the second row). For example, n2.4 (L2 node) is the fourth child of parent n2 (L1 node). Each subplot represents the distribution of genres in a node (visualized as 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.

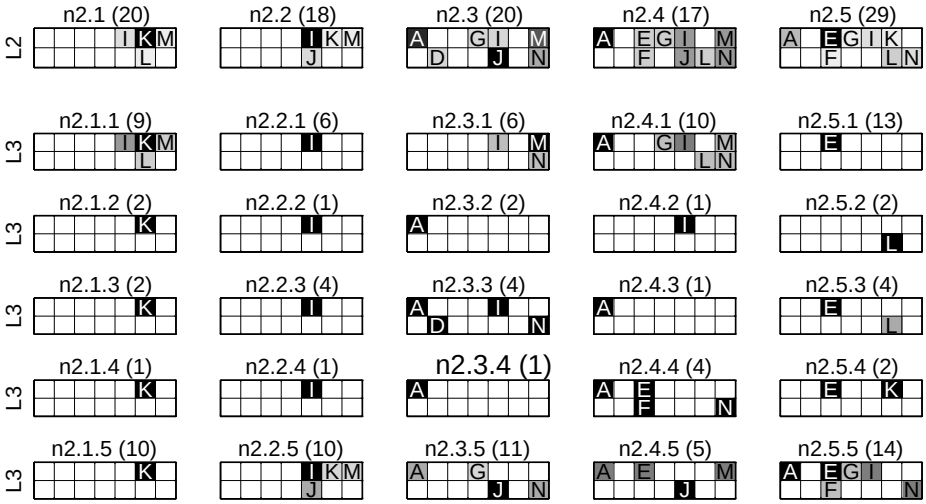


**Fig. 2.** Distribution of the 14 genres in the nodes of the second (L2) and third (L3) level in the subbranch of node n2.

## 5 Term Selection for Cluster Description

Term selection is a core component of the user interface. The goal is to select words which best summarize a group of artists. With respect to the user interaction we make three assumptions with impact on the cluster description. 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. Dropping assumption two could lead to very interesting interactive user interfaces. However, this is beyond the scope of this paper.

In our experiments we compare five term selection techniques, 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.

### 5.1 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 $a$. 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" [23] 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 are very rare in some of the documents in the cluster) are usually poor descriptors. Thus, instead of $w_{tc}$ the variance of $w_{ta}$ in $c$ is used to rank the terms (better descriptors 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 LabelSOM 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 knows what the children have in common after reading the description of the parent. A standard technique to select discriminative terms is the $\chi^2$ (chi-square) test (e.g. [24]). The $\chi^2$-value measures the independence of $t$ from group $c$ and is computed as,

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

where $A$ is the number of documents in $c$ containing $t$, $B$ the number of documents not in $c$ containing $t$, $C$ 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 $\chi^2_{tc}$ values are selected because they are least independent from $c$. Note that the document frequency is very informative because $df_{ta}$ describes the percentage of times the terms occur in the 50 retrieved documents per artist.

The fourth technique was proposed by Lagus and Kaski (LK) [25]. Like LabelSOM it was developed to label large document collections organized by SOMs. While $\chi^2$ only uses $df$, LK only use $tf$. The heuristically motivated ranking formula (higher values are better) is,

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

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. As we will see later his approach can only produce meaningful results when used with 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,

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

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

### 5.2 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,[11] 1269 of these 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

[11] http://www.oefai.at/~elias/wa/dict.txt

music genres (and subgenres), instruments, or adjectives. The dictionary is far from complete and contains terms which should be removed (e.g. *world*, *uk*, *band*, and *song*). 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 two 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. Considering the size of the collection this might not be significant. However, 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% classifcation accuracies on our set of 224 artists. However, such results could not be generalized to other music collections. Furthermore, in our current approach the specialized dictionary can be replaced at any time without impact on the hierarchical structure.

## 6 Results and Discussion

In this section we first describe the user interface we implemented to demonstrate the approach. Second, we compare 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.

### 6.1 User Interface

To demonstrate our approach we implemented a very simple HTML interface.[12] There are two 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 term's 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

[12] http://www.oefai.at/~elias/wa

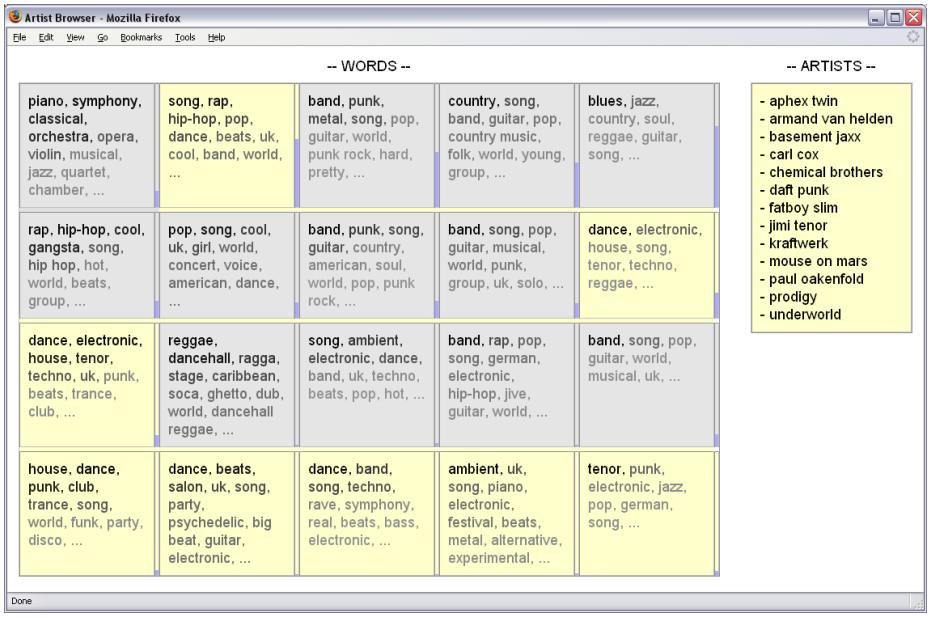


**Fig. 3.** Screenshot of the HTML user interface.

possible to display the list of all ranked 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).

### 6.2 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 general we have made the following observations:

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

| | 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 |
| | $\chi^2$ | |
| 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 |
| | $\chi^2$· 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, southern rap, east coast rap | pies, grandmaster, hash, tricky, pimp |
| n3 | pop-metal, melodic metal, detroit rock, flamenco guitar, math rock | roisin, pies, hash, dez, voulez |
| n4 | new traditionalist, british folk-rock, progressive bluegrass, gabba, slowcore | csn, dez, voulez, shapes, daltrey |
| n5 | rockabilly revival, new beat, progressive country, vocalion, freakbeat | hodges, precious, shanty, broonzy, dez |
| | $\chi^2$· 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.

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 ($tf \times idf$, LabelSOM) is very poor. On the other hand, all discriminative approaches ($\chi^2$, 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.

### 6.3 Discussion

One of the main problems is that our approach relies on 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 yesterday's 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. [26]). We studied this in [8] and the study was continued in [27]. So far we observed significant changes in the Google ranks, but these did not have a significant impact on the similarity measure.

## 7 Conclusions

In this paper we demonstrated possibilities to hierarchically organize music at the artist level. In particular we suggested using hierarchical clustering with overlapping clusters which are described using a domain-specific dictionary. 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.

## 8 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 Ministries bm:bwk and bmvit.

## References

1. Pachet, F., Cazaly, D.: A taxonomy of musical genres. In: Proc RIAO Content-Based Multimedia Information Access (2000)
2. Pachet, F., Westerman, G., Laigre, D.: Musical data mining for electronic music distribution. In: Proc WedelMusic Conf (2001)

3. Whitman, B., Lawrence, S.: Inferring descriptions and similarity for music from community metadata. In: Proc Intl on Computer Music Conf (2002)
4. Baumann, S., Hummel, O.: Using cultural metadata for artist recommendation. In: Proc WedelMusic Conf (2003)
5. Zadel, M., Fujinaga, I.: Web services for music information retrieval. In: Proc Intl Conf Music Information Retrieval (2004)
6. Schedl, M., Knees, P., Widmer, G.: A web-based approach to assessing artist similarity using co-occurrences. In: Proc Workshop Content-Based Multimedia Indexing (2005)
7. Ellis, D., Whitman, B., Berenzweig, A., Lawrence, S.: The quest for ground truth in musical artist similarity. In: Proc Intl Conf Music Information Retrieval (2002)
8. Knees, P., Pampalk, E., Widmer, G.: Artist classification with web-based data. In: Proc Intl Conf Music Information Retrieval (2004)
9. Whitman, B., Ellis, D.: Automatic record reviews. In: Proc Intl Conf Music Information Retrieval (2004)
10. Logan, B., Kositsky, A., Moreno, P.: Semantic analysis of song lyrics. In: Proc IEEE Intl Conf Multimedia and Expo (2004)
11. Whitman, B., Smaragdis, P.: Combining musical and cultural features for intelligent style detection. In: Proc Intl Conf Music Information Retrieval (2002)
12. Baumann, S., Pohle, T., Shankar, V.: Towards a socio-cultural compatibility of MIR systems. In: Proc Intl Conf Music Information Retrieval (2004)
13. Pampalk, E.: Islands of music: Analysis, organization, and visualization of music Archives. MSc thesis, Vienna University of Technology (2001)
14. Pampalk, E., Rauber, A., Merkl, D.: Content-based organization and visualization of music archives. In: Proc ACM Multimedia (2002)
15. 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 Intl Conf Music Information Retrieval (2002)
16. Schedl M.: An explorative, hierarchical user interface to structured music repositories. MSc thesis, Vienna University of Technology (2003)
17. Pampalk, E., Hlavac, P., Herrera, P.: Hierarchical organization and visualization of drum sample libraries. In: Proc Intl Conf Digital Audio Effects (2004)
18. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management **24** (1988)
19. Kohonen, T.: Self-Organizing Maps. Springer (2001)
20. Miikkulainen, R.: Script recognition with hierarchical feature maps. Connection Science (1990)
21. Koikkalainen, P., Oja, E.: Self-organizing hierarchical feature maps. In: Proc Intl Joint Conf Neural Networks (1990)
22. Dittenbach, M., Merkl, D., Rauber, A.: The growing hierarchical self-organizing map. In Proc Intl Joint Conf Neural Networks (2000)
23. Rauber, A.: LabelSOM: On the labeling of self-organizing maps. In: Proc Intl Joint Conf Neural Networks (1999)
24. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proc Intl Conf Machine Learning (1997)
25. Lagus, K., Kaski, S.: Keyword selection method for characterizing text document maps. In: Proc Intl Conf Artificial Neural Networks (1999)
26. Lawrence, S., Giles, C. L.: Accessibility of information on the web. In: Nature (1999)
27. Knees, P.: Automatic classification of musical artists based on web-data (in German). MSc thesis, Vienna University of Technology (2004)