Searching and Classifying Non-Textual Information

Will Archer Arentz
Department of Computer and Information Science
Norwegian University of Science and Technology
willa@idi.ntnu.no

May 13, 2004

Abstract

This dissertation contains a set of contributions that deal with search or classification of non-textual information. Each contribution can be considered a solution to a specific problem, in an attempt to map out a common ground. The problems cover a wide range of research fields, including search in music, classifying digitally sampled music, visualization and navigation in search results, and classifying images and Internet sites.

On classification of digitally sample music, as method for extracting the rhythmic tempo was disclosed. The method proved to work on a large variety of music types with a constant audible rhythm. Furthermore, this rhythmic properties showed to be useful in classifying songs into music groups or genre.

On search in music, a technique is presented that is based on rhythm and pitch correlation between the notes in a query theme and the notes in a set of songs. The scheme is based on a dynamic programming algorithm which attempts to minimize the error between a query theme and a song. This operation includes finding the best alignment, taking into account skipped notes and additional notes, use of different keys, tempo variations, and variances in pitch and time information.

On image classification, a system for classifying whole Internet sites based on the image content, was proposed. The system was composed of two parts; an image classifier and a site classifier. The image classifier was based on skin detection, object segmentation, and shape, texture and color feature extraction with a training scheme that used genetic algorithms. The image classification method was able to classify images with an accuracy of 90%. By classifying multi-image Internet web sites this accuracy was drastically increased using the assumption that a site only contains one type of images. This assumption can be defended for most cases.

On search result visualization and navigation, a system was developed involving the use of a state-of-the-art search engine together with a graphical front end to improve the user experience associated with search in unstructured data. Both structured and unstructured data with the help of entity extraction can be indexed in a modern search engine. Combining this with a multidimensional visualization based on heatmaps with navigation capabilities showed to improve the data value and search experience on current search systems.

Acknowledgments

I am grateful to the many people that have provided inspiration and support during the course of this work. In particular, I would like to thank my supervisor Professor Bjørn Olstad for his support and useful suggestions during the course of this work. I would also like to thank John Reidar Mathiassen for his fruitful discussions and comments.

Will Archer Arentz, May 13, 2004

Contents

Contents

List of Figures

1 Introduction

Search in large unstructured data sets is currently a very active research area. The amount of data available is growing rapidly, whether it is textual information or multimedia content. Due to the availability of inexpensive storage devices, digital cameras and video recorders, the amount of multimedia content available on home computers and on the Internet has grown dramatically over the last few years. For this reason an increasing need to efficiently access this information has developed. However, since most of the information is unstructured and to avoid the tedious task of manually annotating the data, good content based search solutions for searching and classifying multimedia content need to be developed.

Distance metrics for multimedia content can be used as a basis for both classification and search algorithms. Calculating multimedia distance metrics can be done in many different ways and is dependent on many factors. Some distance metrics will for instance be limited to boolean or binary evaluation. Furthermore, different distance metrics can be combined in order to create flexible models for information importance. The work in this thesis has been focused on developing new types of distance metrics for multimedia content that can be both efficiently computed and applied in large scale applications. Given a data set and a method for calculating the distance between two elements, the data can be searched simply by matching the query example element against all elements in the data set and generating a sorted result list in accordance to the distance metric. Likewise, clustering can for example be done by using the k-nearest neighborhood algorithm according to the defined distance metric.

Retrieval solutions need to deal with both textual information, multimedia and numerical data. Information retrieval research has mainly been focused on the textual component. The introduction of Internet web search engines introduced large scale search systems as a research topic in information retrieval. Multimedia has been popular for a long time, but the recent availability of cheap storage and input devices, has accelerated the

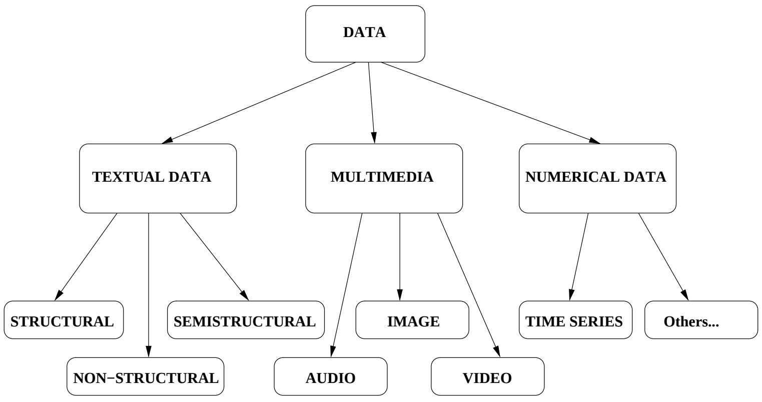


Figure 1: Three key data types in information retrieval applications

usage and importance of multimedia content. Finally, numerical data is an important data type which for example has been a key focus in database design. In addition to data modeled in SQL database schemas, numerical data includes data types and associated applications such as vectors, matrices and time series describing sensor signals. All of the three data types have subclasses with different characteristics. For most of these subclasses there have been attempts to define standard distance metrics. The three main data types with some subclasses can be found in figure 1.

1.1 Background

Textual information retrieval is a quite mature field of research. For decades research has been conducted in this area, and already in 1986 it was broad consent among researchers that automatic text-retrieval had come a long way in replacing manual indexing[87].

Since then, a lot of work has been done both in text and multimedia information retrieval communities, and there are numerous surveys available explaining the current status of content-based image information retrieval[24, 111, 89, 106, 85, 18, 30, 41, 42, 48], music and audio information retrieval[34, 76, 109], and content-based video information retrieval systems[63].

Furthermore, as seen in figure 1, a third group if information retrieval exists. Researches in this field attempts to create information retrieval systems, that operate on numerical data. Numerical data cover such a large range of applications that it is usually necessary to define the data model before generic retrieval solutions can be defined. However, for many isolated problems, successful attempts have been made to create distance measuring techniques. Also, with some groups of numerical data, like Keogh et al.'s work on time series[49, 50], has resulted in general algorithms with matching and classification abilities.

1.1.1 Commercial solutions

Several commercial solutions have recently become available within the field of multimedia information retrieval. Although there are still no content based search solutions for the whole Internet, several metadata based search engines for multimedia content has emerged. The Google image search engine[1] is one example. The Google image search is based on text, anchor text and image attributes extracted from the image header.

In the enterprise segment there has for a long time been various products available that do some kind of media analysis. For example has OCR (Optical Character Recognition) software been around for more than a decade, although it is still subject to improvements. A context based OCR product was recently introduced by SRI International[2]. The context based OCR techniques combine the characters recognized by an OCR engine into words that are part of an intelligible address. These techniques take advantage of the inter word contextual constraints inherent in an address.

[1]http://images.google.com
[2]http://www.sri.com

Such an approach, which is more powerful than conventional recognition approaches, has been implemented in the USPS Recognition Coprocessor (RCP) and used in other SRI OCR and document understanding projects.

The recent focus on terrorism has escalated the development of another recognition technology, namely facial recognition. Several products are currently available on the marked, with FaceIt ARGUS from Identix[3] being one of best known. FaceIt ARGUS is a scalable, off-the-shelf facial recognition system that detects and identifies humans as they pass through a camera's field of view. It attempts to maximize the value of CCTV (Closed Circuit Television) systems by increasing deterrence, increasing active surveillance functionality, increasing investigative power and drastically diminishing or eliminating the challenges that exist when relying solely on operators to conduct surveillance.

Virage, a company in the Autonomy Group, has combined OCR, facial recognition technologies together with a few other technologies, and created the product VideoLogger[4]. The VideoLogger can automatically detect faces and text in the video, and use this to annotate the video, thus making it searchable with common textual search systems.

Other companies have established a more general focus. Products such as ImageSeeker from the French company LookThatUp technologies[5], performs search for image content similarity, by creating database of features for a set of images and matching the query image features against the database to find the closest match.

Nexidia[6] offers a product for searching in spoken audio. The search system scans a phonetic conversion of the audio in a sequential manner, to allow for inaccurate translations of audio without severely affecting the match.

[3]http://www.identix.com
[4]http://www.virage.com
[5]http://www.lookthatup.com
[6]http://www.nexidia.com

2 Problem definition

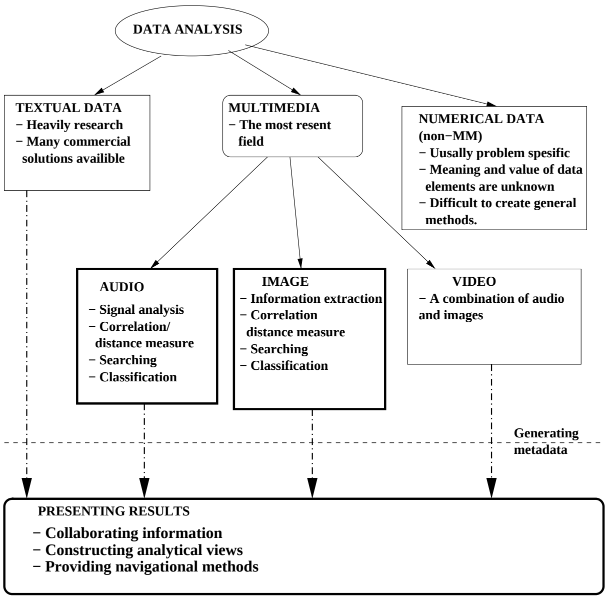

DATA ANALYSIS
TEXTUAL DATA
− Heavily research
− Many commercial solutions availible
MULTIMEDIA
− The most resent field
NUMERICAL DATA (non−MM)
− Uusally problem spesific
− Meaning and value of data elements are unknown
− Difficult to create general methods.
AUDIO
− Signal analysis
− Correlation/ distance measure
− Searching
− Classification
IMAGE
− Information extraction
− Correlation distance measure
− Searching
− Classification
VIDEO
− A combination of audio and images
Generating metadata
PRESENTING RESULTS
− Collaborating information
− Constructing analytical views
− Providing navigational methods

Figure 2: An overview of the work in the dissertation and how it relates to other research areas.

Searching and classifying non-textual information spans several large fields of research. Hence the focus of the dissertations had to be focused on key improvement opportunities. In figure 2 an overview of the field of search and classification is depicted, where the thick lined boxes represent the work that has been the focus of this dissertation. The figure first illustrates how content based data analysis is subdivided into three types: text, multimedia and numerical data sources. Due to the maturity of textual analysis it was decided to focus on non-textual information in this dissertation. Another type of data is numerical data such as sensor logs, time-series, 3D models, etc., but an application model needs to be introduced before this problem domain can be addressed for similarity based search in a generic manner. Hence, it was decided to focus the dissertation work on multimedia content. Multimedia content can be further subdivided into audio, images and video.

Content based retrieval of audio based data is still a quite open field. Much work is yet to be done. Two of the papers presented in this dissertation are focused on audio. The first paper, “Beat extraction from digital music”, presents work based on signal analysis, where rhythmic properties are extracted from digitally sampled music. The second paper, “Retrieving musical information based on rhythm and pitch correlations”, propose a distance metric for symbolic music. This work deals with search and correlation, while the methods can easily be extended to classification.

Analysis of image content has been worked on since the introduction of digital images, most often for very specific problems with very controlled image data. Several systems for content based image retrieval are described in the literature[24, 89, 111]. Although, we have yet to see a global image retrieval system, like the ones available for textual information. This is mainly due to processing speed and accuracy issue for the available content based analysis. For this reason it was decided to create a fast system for information extraction from images, which was scalable to the global Internet. This work deals with distance metrics applied to classification, although the techniques can easily be extended to information retrieval problems. The paper is entitled “Classifying offensive sites”.

Finally, video is a key data source in multimedia retrieval. A few systems for content based video retrieval are presented in the literature[84, 108, 120], although it is still a quite unexplored field. The limited success for video information retrieval so far is mainly due to the large storage and processing demands combined with a need to embed application logic in the similarity metrics. Furthermore, as video is a composition of audio and images, work done in these fields, can easily be extended to video later. For this reason, no work on video data was included in this dissertation.

The lower part of figure 2 represents the most visible part of an information retrieval system. When information has been added to a search engine either as raw data, or as generated or attached metadata, the results must be converted to concepts that support information discovery processes. The traditional way of representing search results is by listing the closest matching results with a title, a teaser and a link to the data. This view possesses no navigational abilities, and makes it virtually impossible for the user to get a good overview of the complete result set. Several attempts have been made to visualize the information retrieval process [56, 121, 122, 44, 97, 54], although they are often not very informative, and lack navigation and entity extraction capabilities. Therefore, it was decided to develop a system capable of visualizing the result set in an more informative way, and at the same time provide an easy and intuitive way of navigating through the search results. It was designed to work on top of a state-of-the-art search engine with entity extraction capabilities from unstructured and semistructured data. This work is documented in the paper "Multidimensional Visualization and Navigation in Search Results". The examples in the paper use metadata for scientific publications. It is on the other hand evident that the same technology can be used to correlate and visualize relationships between textual and multimedia metadata. The proposed framework is hence a possible way to navigate across textual, multimedia and numerical data sources.

2.1 "Beat extraction from digital music"

2.1.1 Description of the problem

Storing of music on home computers, and in multi-user large databases has over the last few years become very normal. Operating systems come with support for easy copying and compression of traditional Compact Discs (CD) to local music databases. File formats like *mp3* and *ogg* deliver more than 10 times compression from the raw sampled data, without audible reduction in sound quality. It is now possible to store several thousand songs on a home computer. This again creates a need for automatic classification of songs.

A good classification criteria, and also a reasonable determinant for the musical genre for a song is the rhythm. Since the rhythm is mostly repetitive with few variations, and also the most energy dominant element in a song, it can be easily spotted in the temporal signal. Also, the range rhythmic tempos for most music is fixed, and thus limiting the search space. A straight forward solution to this problem would be to run autocorrelation on the whole search space. However, this proves to be very computationally expensive, and also very sensitive to tempo changes within the song. Another approach would be to search in the very low frequency components describing the speed of the rhythm in the frequency domain. Although use of frequency information could be used to extract much more than just the tempo, it has proven to be rather inaccurate and sensitive to noise.

2.1.2 Summary of contribution

A fast algorithm is presented for detecting the most authoritative beat sequence, independent of tempo changes within the song. The method is based on extracting sample peaks in the temporal domain, before investigating the recurrence of peaks at time $\Delta t \times x$. Setting the limitation of possible tempos to 70-160 BPM, showed to included most popular songs. This means that Δt must be in the interval $[\frac{3}{8}, \frac{6}{7}]$ seconds. The final Δt is chosen where most recurrences occur.

In addition, some heuristics allowing for missing peaks and temporal inaccuracies in the recurrency sequences were implemented.

2.2 "Retrieving musical information based on rhythm and pitch correlations"

2.2.1 Description of the problem

An information retrieval problem that has grown in popularity over the last couple of years is *Query By Humming* (QBH). The large databases of music easily available today has contributed in attempts to create easier and perhaps more natural ways of data access. One such attempt is QBH, which basically means that one use humming, whistling or singing as query when searching for a song. This can often be easier since people tend to forget names while remembering tunes and refrains.

The key problem is to create a system able to find the right match even from just a short theme, without knowing the position of the theme in the song. Furthermore, since most people's musical abilities are quite substandard, any successful system must be able to handle various types of errors.

Since different errors have different penalties in the sense of human recognition ability after errors are applied, a system would benefit from punishing different errors independently. For example, when a person sings a song, no listeners will pay any attention to small errors in timing of the different notes, nor will they care if the singer use wrong key, or maybe even changes the key within the song. They will, however, notice if the singer repeatedly misses the pitch of his notes. This suggests that punishing pitch should be more forceful than for example timing/rhythm errors.

Most QBS-systems developed are symbol based. That means that both database and query is represented as symbols such as musical notes. Such symbols are given certain properties, such as timestamp, duration, press-strength, volume, key and pitch value. Even though, not all of this information is needed in a musical retrieval algorithm, pitch and timestamp seem to carry the most musical value. Despite this obvious fact, most current work on query by humming has been done on either pitch or timing.

Thus, resulting in systems unable to understand the human interpretation of distance between a query theme and a song.

2.2.2 Summary of contribution

An algorithm is presented that uses both pitch and timing information to search in musical symbol databases with a query theme containing symbols with timestamp and pitch. Additionally, the algorithm was designed to handle key transposes, additional notes, missing notes and tempo variations with independent penalties.

The algorithm is also able to handle polyphonic songs. That is, songs containing more than one instrument track, while monophonic songs only contain one single instrument track. Maneuvering multi track songs, is simply done by sequentially searching through all tracks, before selecting the match with the smallest error to represent the best match for the given song.

The dynamic programming algorithm attempts to find the alignment of a theme within a track or song with the smallest error. The dynamic algorithm's table is composed of the query on one axis and the instrument track on the other. In other words, a sequential search through the song is not necessary, thus reducing computational costs. Tests of the fully developed QBS-system have shown very satisfactory results both in regards of computational costs and retrieval capabilities.

Even though the presented paper only discuss use of the algorithm on music matching, the method is general, and could also be used on i.e. phoneme matching in voice recognition systems. Either just to look up words, or to search through large data sets of speech.

2.3 “Classifying offensive sites”

2.3.1 Description of the problem

The recent development of multimedia availability on the Internet has enabled Internet users around the world to access enormous amounts of video-, picture- and audio-files. However, this has also lead to new business-opportunities for the adult industry. Furthermore, due to this industry’s aggressive marketing, it has become very difficult to avoid stepping into an adult site from time to time, be it through web-surfing, spam-mail, or search results from a web search engine. This may be very upsetting to some people, especially parents with Internet surfing children. A system preventing the browser to enter such sites has been requested on several occasions. This problem is especially evident in web search engines, where a company often has some responsibility for the quality of the search results.

A solution to this problem is to blacklist URLs, so that applying the proper filter can stop these pages and pictures from being read. Concentrating on image content only, classifying each image on the web individually is virtually impossible. This is mainly due to imperfect image analysis tools, but also to the fact that the various degree of offensiveness is very hard to concretize even for the human eye.

Doing classification on all images on a site collectively is one way of improving the performance of such a classification scheme. Fortunately, most Internet sites only contain one class of images, so in most cases such an approach should work quite well.

On the actual image analysis, most attempts has been focused around skin detection and calculating the size of the skin region in an image. Such an approach works fine for some images, but is not general enough when considering i.e. portrait or beach pictures. Using texture, relative location, size and shape are all useful contributions to improve the accuracy of such a process.

2.3.2 Summary of contribution

A technique for classifying offensive sites, based on image content is proposed. This technique is independent of image analysis tool, but a fast algorithm for doing image analysis is also presented.

The site classification is based on the assumption that all images on a site is either offensive or non-offensive. Furthermore, they are assumed to be independent. This opens up for the possibility of using binomial probability distribution to calculate the probability of a site being classified correctly given the number of images on the site, and the classification accuracy of the image analysis algorithm. The experiments done in the paper shows that this scheme results in very small probabilities of classifying a site wrong.

The image analysis tool disclosed was designed to provide very fast analysis of individual images. This was done using a simple thresholding skin segmentation as data reduction, before extracting feature vectors composed of shape, texture, color, size and location for each segment. Grouping these segments into offensive or non-offensive classes with associated probabilities, made it possible to calculate a probability that a given picture is offensive. Using this scheme made it possible to achieve high classification rate, at very fast speed. And in our experiments the developed image analysis tool complemented the site classification scheme in a very good way.

2.4 “Multidimensional Visualization and Navigation in Search Results”

2.4.1 Description of the problem

Search engines, especially like the ones indexing the web, have grown rapidly in popularity over the last decade. As the amounts of information indexed by a search engine grows, the harder it is to provide good relevant results, and thus the harder it is to navigate in the search results. Various methods for viewing search results have been developed. Some improved the search experience for the more experienced users, while the overall conclusion has been that novice users require training in order to fully benefit from the visualization.

The most common search interface today is simply an input text field and a “Search” button, while the results are shown as a list of text results. The results are usually composed of a title, a teaser, a hypertext link, and sometimes the document size. This interface is rather easy to understand, but very limited in terms of information value and overview. However, the simplicity of the interface makes it easy to start using and very suitable for novice users.

Since modern search engines are capable of handling a multiple hundred queries per second on a single CPU, this can be exploited to dig deeper into the data set. Some prototypes have been developed that sends the initial query to multiple search engines, before collecting and sorting the results in a single view. However, little work has been done on using multiple queries on a single search engine to improve the search result information value. State-of-the-art search engines also support automatic entity extraction for unstructured data. This can be exploited to construct views and visualizations, that provide more information to the users. It can also be utilized to navigate within the initial search result set.

2.4.2 Summary of contribution

A visualization and navigation interface was built on top of a state-of-the-art search engine. The interface was designed to visualized multidimensional data, gathered using a sequence of resubmitted queries. The results from the initial query were first analyzed, before resubmitting a set of queries to fill out the needed information. The visualization was done using heatmaps, where the axis represented entity classes. The cell color could have different meanings, while the obvious would be that it represented the number of search results in the result set or an indication of time. As default the system automatically choose a heatmap view with the entity classes that provided the best discrimination as axis. The axis could then be changed upon request.

As the search engine allows for resubmitting queries by modifying entities, this was exploited in the disclosed application to navigate in the search results. Specifying an entity in a search reduces the result space. By specifying multiple entities, an initially uncomprehendible result space may become small and easy to browse trough.

The proposed combination of visualization with heatmaps and navigation in the search result spaces showed to increase the information value and improve the search experience. Although the more experienced user may benefit more from this than the untrained novice.

2.4.3 Additional Experiments

The proposed visualization scheme's power and applicability can most easily be understood from examples. As the developed application was completely general, it could be used with any existing Fast Data Search installation. Some additional example views showing the information value available trough visualization with heatmaps can be found in the appendix. All the examples use a data index composed of medical articles.

1. This example shows the whole process of investigating an authors publication pattern. The initial query shown in figure 3 is "echocardiography". This query returns a rather large search result set, making up every article that has something to do with echocardiography. By specifying the author on the x-axis and and publication year on the y-axis in the heatmap visualization, one can easily read out the authors activity over they years in the field of echocardiography. This is shown in figure 4. From this figure it can also be seen that the author *J. Seward* has been very active in this field. If we want

to investigate this author further, we can add the “J. Seward” to the query by clicking on the column label in the heatmap. As the new query is resubmitted, information about the J. Seward’s publication pattern may be viewed.

Figure 5 displays the most frequent co-authors in J. Seward’s publications. The list was obtained by specifying authors on the y-axis. In the same way his research topics was found in figure 6. Finally, we view where he has published his work in correlation with what chemical substances they contain.

2. Also in this example is the query “echocardiography”. Here the view is selected to be MeSH-terms in correlation with chemical substance (see figure 8). Notice for example that the MeSH-term “dog” has correlation with the chemical substance “Contrast Media”. The reason for this is that dogs are used in contrast research.

3. As an example of marked intelligence, this example searches on two medical appliance companies. The first search is on “Vingmed”, which is a company delivering ultrasound scanners primarily used in cardiography. The search on Vingmed is shown in figure 9. Another search is done on “Acuson” in figure 10. This company also produces ultrasound scanners, although they are use in a much broader range of fields than just cardiography. It can be seen from the figures that different people use the different appliances. Also, the topics differ.

4. The last example shows the result heatmap from the query “radiodurans”. Radiodurans is a micro-organism that can withstand very high levels of radiation. The view in figure 11 shows the correlation between topics and publication year. It should be noted that there are publications about complete genome only in 1999 and 2000. This seems natural as the discovery of how to map the complete genome is a very recently discovery.

References

[1] Collaborative filtering research papers.
`http://jamesthornton.com/cf/`.
Last visited 27th of January, 2004.

[2] Data searching and tries.
`http://wannabe.guru.org/alg/node38.html`.
Last visited 29th of August, 2000.

[3] The flamenco search interface project.
`http://bailando.sims.berkeley.edu/flamenco.html`.
Last visited 27th of January, 2004.

[4] NIST PRISE search engine.
`http://www.itl.nist.gov/iaui/894.02/works/papers/zp2/zp2.html`.
Last visited 27th of January, 2004.

[5] A note on the average depth of tries.
`http://citeseer.nj.nec.com/cidcontext/1602764`.
Last visited 29th of August, 2000.

[6] A summary of the sarbanes-oxely act of 2002.
`http://www.aicpa.org/info/sarbanes_oxley_summary.htm`.
Last visited 14th of February, 2004.

[7] C. Zhou A. Kitamoto and M. Takagi. Similarity Retrieval of NOAA Satellite Images by Graph Matching. *Storage and Retrieval for Image and Video Databases*, 1908:60–73, 1993.

[8] R.B. Allen, P. Obry, and M. Littman. An interface for navigating clustered documents sets returned by queries. In *Proceedings of SIGOIS, Milpitas CA*, pages 203–208, 1993.

[9] P. Alshuth, T. Hermes, C. Klauck, J. Kreyss, and M. Roper. Iris - image retrieval for images and videos, 1996.

[10] E. Angelopoulou. Understanding the color of human skin. pages 243–251.

[11] Will A. Arentz. BPM analysis of digital music. Technical report, Norwegian University of Science and Technology, 2000.

[12] Will Archer Arentz. MSc. Thesis, *Live Media Search*, 2000.

[13] AudioWorks Inc. *Waw2midi - Convert monophonic digital music to midi*. `http://www.audioworks.com/w2m/w2m.htm`.

[14] S. Baheerathan, F. Albregtsen, and H. E. Danielsen. New texture features based on the complexity curve. In *Pattern Recognition*, volume 32, pages 605–618, 1999.

[15] Richard Bellman. *Dynamic Programming*. Cambridge Studies in Speech Science and Communication. Princeton University Press, 1957.

[16] S. Benford, D. Snowdon, C. Greenhalgh, R. Ingram, I. Knox, and C. Brown. VR-VIBE: A virtual environment for co-operative information retrieval. In *Eurographics '95, Maastricht, The Netherlands*, pages 349–360, 1995.

[17] Michael W. Berry and Murray Browne. *Understanding search engines - Mathematical modeling and text retrieval*. Society for Industrial and Applied Mathematics, 1999.

[18] Alberto Del Bimbo. Visual information retrieval. In *Morgan Kaufman*, 1999.

[19] S. Blackburn and D. D. Roure. A tool for content-based navigation of music. In *Proceedings of ACM International Multimedia Conference (ACMMM)*, 1998.

[20] Lars Bretzner, Ivan Laptev, and Tony Lindeberg. Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering, 2002.

[21] S. Brumby, N. Harvey, S. Perkins, R. Porter, J. Szymanski, J. Theiler, and J. Bloch. A genetic algorithm for combining new and existing image processing tools for multispectral imagery, 2000.

[22] S. Brumby, J. Theiler, S. Perkins, N. Harvey, J. Szymanski, J. Bloch, and M. Mitchell. Investigation of image feature extraction by a genetic algorithm, 1999.

[23] A. L.P. Chen, M. Chang, and J. Chen. Query by music segments: An efficient approach for song retrieval. In *Proceedings of IEEE International Conference on Multimedia and Expo.*, 2000.

[24] Chan Kin Ching, Lam Ka Wing, Lin Chi Wing, Hsieh Ding Fei, Li Yuk Hin, and Ng Tsz Hin. Content-based image retrieval. *http://www.cs.ust.hk/faculty/dimitris/COMP530/ImageRetrieval.pdf*.

[25] S. Cohen and L. Guibas. Shape-based image retrieval using geometric hashing, 1997.

[26] Charles Darwin. *On the Origin of Species by Means of Natural Selction*. John Murray, 1859.

[27] David R. Bull David Beasley and Ralph R. Martin. An Overview of Genetic Algorithms: Part 1, Fundamentals. *University Computing*, 15(2):58–69, 1993.

[28] David R. Bull David Beasley and Ralph R. Martin. An Overview of Genetic Algorithms: Part 2, Resarch Topics. *Univeristy Computning*, 15(4):170–181, 1993.

[29] L. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, 1991.

[30] John P. Eakins and Margaret E. Graham. Content-based image retrieval, a report to the JISC technology applicationi programme. Technical report, Institute for Image Data Research, University of Northumbria at Newcastle UK, 1999.

[31] E. Efthimiadis. Interactive query expansion and relevance feedback for document retrieval systems. In *PhD thesis, City University, London, UK*, 1992.

[32] J.D. Ferguson. *Hidden Markov Analysis: An Indroduction in Hidden Markov Models for Speech*. Institute for Defense Analysis, Princeton, 1980.

[33] Margaret M. Fleck, David A. Forsyth, and Chris Bregler. Finding naked people. In *ECCV (2)*, pages 593–602, 1996.

[34] Jonathan Foote. An overview of audio information retrieval. *Multimedia Systems*, 7(1):2–10, 1999.

[35] D. Forsyth and M. Fleck. Identifying nude pictures, 1996.

[36] R. Fowler, B. Wilson, and W. Fowler. Information navigator: An information system using associative networks for display and retrieval. In *Report NAG9-551, No. 92-1, Univ. of Texas - Pan American, Edinburg TX*.

[37] Stephen I. Gallant. *Neural Network Learning and Expert Systems*. MIT Press, 1993.

[38] S. Geman and D. Geman. Stochastic relaxation, gibbs distribution and the bayesian restoration of images. In *IEEE Tansaction on Part. Analysis and Machine Intelligence*, volume 6, pages 721–741, 1984.

[39] Asif Ghias, Jonathan Logan, David Chamberlin, and Brian C. Smith. Query by humming: Musical information retrieval in an audio database. In *ACM Multimedia*, pages 231–236, 1995.

[40] J.J. Grefenstette. *Algorithms and Simulated Annealing*. Pitman, 1987.

[41] W. I. Grosky. Multimedia information systems. *IEEE Multimedia*, 1(1):12–24, 1994.

[42] V. N. Gudivada and V. V. Raghavan. Content-based image retrieval systems. *IEEE Computer*, 29(9):18–31, 1995.

[43] M. Hearst and C. Karadi. Cat-a-cone: An interactive interface for specifying searches and viewing retrieval results using a large category hierarchy. In *Proceedings 20th Annual International ACM/SIGIR Conference, Phil PA*, 1997.

[44] M.A. Hearst and J.O. Pedersen. Visualizing information retrieval results: A demonstration of the titlebar interface. In *CHI'96 Proceedings*, 1996.

[45] J.H. Holland. *Adaption in Natural and Artificial Systems*. MIT Press, 1975.

[46] R.-L. Hsu, M. Abdel-Mottaleb, and A. Jain. Face detection in color images. pages 1046–1049.

[47] A. Jain and A. Vailaya. Image retrieval using color and shape, 1996.

[48] R. Jain. Infoscopes: Multimedia information system. In *Multimedia Systems and Techniques*, pages 217–253, 1996.

[49] E. Keogh and M. Pazzani. An enhanced representation of time series witch allows fast and accurate classification, clustering and relevance feedback. In *Proceedings of the 4th Int'l Conference on Knowledge Discovery and Data Mining,* pages 239–241, 1998.

[50] E. Keogh and P. Smyth. A probabilistic approach to fast pattern matching in time series databases. In *Proceedings of the 3th Int'l Conference on Knowledge Discovery and Data Mining*, pages 24–30, 1997.

[51] J. Koenemann and N.J. Belkin. A case for interaction: A study of interactive information retrieval behaviour and effectiveness. In *CHI'96 Proceedings,* 1996.

[52] Erwin Kreyszig. *Advanced Engineering Mathematics*. John Wiley & Sons, 1995.

[53] J. Lamping, R. Rao, and P. Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *CHI'95 Proceedings: Conference on Human Factors in Computer Systems: Mosaic of Creativity, Denver CO*, 1995.

[54] C. Lee and Y.-T. Chen. Distributed visual reasoning for intelligent information retrieval on the web. In *Interacting with Computers* **12**, pages 445–467, 2000.

[55] K. Lemstrom, P. Laine, and S. Perttu. Using relative interval slope in music information. retrieval. In *Proceedings of International Computer Music Conference*, pages 317–320, 1999.

[56] W.-S. Li, J. Shim, and J.S. Candan. WebDB: A system for querying semi-structured data on the web. In *Journal of Visual Languages and Computing* **13**, pages 3–33, 2002.

[57] A. Lindsay. Using contour as a mid-level representation of melody, 1994.

[58] Philip L. Long and David C. Hartmann. *Study of message text formats, bibliographic search queries*. Library of Congress, 1979.

[59] G. Loy. Musicians make a standard: The midi phenomenon. In *Computer Music Journal 9(4)*, pages 8–26, 1985.

[60] B.S. Atal L.R. Rabiner and M.R. Sambur. LPC Prediction Error - Analysis of its variation with the position of the Anaslysis Frame. *IEEE Trans. Acoustics, Speech , Signal Proc.*, ASSP-25(5):434–442, 1977.

[61] R. Duncan Luce. *Sound & Hearing - A Conceptual Introduction*. Erlbaum, 1993.

[62] Heikki Mannila and Pirjo Ronkainen. Similarity of event sequences (revised version). In *Proceedings of the Fourth International Workshop on Temporal Representation and Reasining, TIME 97*, pages 136–139, 1997.

[63] Stephane Marchand-Maillet. Content-based video retrieval: An overview. Technical report, CUI-Universite de Geneve, 2002.

[64] Birgitta Martinkauppi Maricor. Behavior of skin color under varying illumination seen by different cameras at different color spaces.

[65] J.D. Markel and Jr. A.H. Gray. *Linear Prediction of Speech*. Springer-Verlag, 1976.

[66] Hannes Kruppa Martin. Skin patch detection in real-world images.

[67] Michael Meyer. *Methods of text and discourse analysis - In search of meaning*. Sage, 2000.

[68] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, 1992.

[69] National Radio Systems Committee. *RBDS versus RDS - What are the differenc and who can recievers cope with both systems?*, 1998. Available from `http://www.nab.org/scitech/rbdsrds.pdf`.

[70] National Radio Systems Committee. *UNITED STATES RBDS STANDARD*, 1998. Available from `http://www.nab.org/scitech/rbds1998.pdf`.

[71] G.B. Newby. Empirical study of 3d visualization for information retrieval tasks. In *Journal of Intelligent Information Systems* **18**, pages 31–53, 2002.

[72] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[73] J.S. Park and D.H. Chang. 2-d invariant descriptors for shape-based image retrieval.

[74] Steffen Pauws. Cubyhum: A fully operational query by humming system. In *Proceedings of ISMIR*, pages 187–196, 2002.

[75] W. Niblack R. Barber W. Equitz M. Flickner E. Glassman D. Petkovic and P. Yanker. The QBIC project: querying images by content using colour, texture and shape, 1993.

[76] Jeremy Pickens. A survey of feature selection techniques for music information retrieval. *http://citeseer.nj.nec.com/460365.html*.

[77] Ken C. Pohlmann. *Principles of Digital Audio, 3rd edition*. McGraw-Hill, 1995.

[78] Riccardo Poli. Genetic programming for image analysis. In John R. Koza, David E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 363–368, Stanford University, CA, USA, 28–31 1996. MIT Press.

[79] A.B. Poriz. Linear predictive hidden Markov models and the speech signal. *Proc. ICASSP 82*, pages 1291–1294, 1982.

[80] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals Of Speech Recognition*. Prentice Hall, 1993.

[81] J. M. Rehg, K. P. Murphy, and P. W. Fieguth. Vision-based speaker detection using bayesian networks. pages 110–116.

[82] E. Rennison. Galaxy of news: An approach to visualizing and understanding expansive news landscapes. In *Proceedings of UITS'94*, 1994.

[83] Brian D. Ripley. *Stochastic Simulatioin*. John Wiley & Sons, 1999.

[84] Volker Roth. Content-based retrieval from digital video. *Image and Vision Computing*, 17(7):531–540, 1999.

[85] Yong Rui, Thomas S. Huang, and Shih-Fu Chang. Image retrieval: Current techniques, promising directions and open issues. *Journal of Visual Communication and Image Representation*, 10(1):1–23, 1999.

[86] Maytham Safar, Cyrus Shahabi, and Xiaoming Sun. Image retrieval by shape: A comparative study. In *IEEE International Conference on Multimedia and Expo (I)*, pages 141–144, 2000.

[87] Gerard Salton. Another look at automatic text-retrieval systems. *Communications of the ACM*, 29(7), 1986.

[88] David Sankoff and Joseph Kruskal, editors. *Time Warps, String Edits, and Macromolecules : The Theory and Practice of Sequence Comparison*. CSLI Publications, reissue edition, 1999.

[89] Raimondo Schettini, Gianluigi Ciocca, and Silvia Zuffi. A survey of methods for colour image indexing and retrieval in image databases. *http://citeseer.ist.psu.edu/489355.html*.

[90] M.M. Sebrechts, J. Vasilakis, M.S. Miller, J.V. Cugini, and S.J Laskowski. Visualization of search results: A comparative evaluation of text, 2d and 3d interfaces. In *Research and Development in Information Retrieval*, pages 3–10, 1999.

[91] Min C. Shin, Kyong I. Chang, and Leonid V. Tsap. Does colorspace transformation make any difference on skin detection?

[92] I. Shmulevich, O. Yli-Harja, E. Coyle, D. Povel, and K. Lemstrom. Perceptual issues in music pattern recognition - complexity of rhythm and key finding, 1999.

[93] J. R. Smith and S. F. Chang. *VisualSeek: A fully automated content-based image query system*. International Conference on Image Processing, 1996.

[94] S. W. Smoliar, J. D. Baker, T. Nakayama, and L. Wilcox. Multimedia search: An authoring perspective. In *Proceedings of the First International Workshop on Image Databases and Multimedia Search*, pages 1–8, 1996.

[95] S. W. Smoliar and H. Zhang. Content based Video Indexing and Retrieval. *IEEE Multimedia*, 1(2):62–72, 1994.

[96] Jungmin Song, So Young Bae, and Kyoungro Yoon. Mid-level music melody representation of polyphonic audio for query-by-humming system. In *Proceedings of ISMIR*, pages 133–139, 2002.

[97] D. Stenmark. To search is great, to find is greater: a study of visualisation tools for the web. In *See: http://webb.informatik.gu.se/ dixi/publ/mdi.htm*.

[98] Graham A. Stephen. *String Search Algorithms*. World Scientific Publishing Company, 1994.

[99] W. Richard Stevens. *Advanced Programming in the UNIX Enivronment*. Addison Wesley, 1998.

[100] W. Richard Stevens. *UNIX Network Programming, Vol.1, 2nd edition*. Prentice Hall, 1998.

[101] Moritz Störring, Hans J. Andersen, and Erik Granum. Skin colour detection under changing lighting cond itions. In Helder Araujo and Jorge Dias, editors, *7th International Symposium on Intelligent Robotic Systems*, pages 187–195, Coimbra, Portugal, 1999.

[102] A.G. Sutcliffe, M. Ennis, and J. Hu. Evaluating the effectiveness of visual user interfaces for information retrieval. In *International Journal of Human Computer Studies* **53**, pages 741–763, 2000.

[103] R.C. Swan and J. Allan. Aspect windows, 3d visualizations, and indirect comparison of information retrieval systems. In *Proceedings of the 21st Annual International ACM/SIGIR Conference, Melbourne Australia*, 1998.

[104] J. Szymanski, S. Brumby, P. Pope, D. Eads, D. Esch-Mosher, M. Galassi, N. Harvey, H. McCulloch, S. Perkins, R. Porter, J. Theiler, A. Young, J. Bloch, and N. David. Feature extraction from multiple data sources using genetic programming, 2002.

[105] Walter Alden Tackett. Genetic programming for feature discovery and image discrimination. In Stephanie Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*, pages 303–309, University of Illinois at Urbana-Champaign, 17-21 1993. Morgan Kaufmann.

[106] Hideyuki Tamura and Naokazu Yokoya. Image database systems: A survey. *Pattern Recognition*, 17(1):29–43, 1984.

[107] Charles E. Leiserson Thomas H. Cormen and Ronald L. Riverst. *Introduction to Algorithms*. MIT Press, 1990.

[108] Roland Tusch, Harald Kosch, and Laszlo Boszormenyi. Videx: An integrated generic video indexing approach.

[109] George Tzanetakis and Perry Cook. Audio information retrieval (AIR) tools. *http://www.cs.princeton.edu/ gtzan/papers/ismir2000.ps*.

[110] R. Veltkamp and M. Hagedoorn. State-of-the-art in shape matching, 1999.

[111] Remco C. Veltkamp and Mirela Tanase. Content-based image retrieval systems: A survey. *http://give-lab.cs.uu.nl/cbirsurvey/*.

[112] G.M. White and R.B. Neely. Speech Recognition Experiments with Linear Prediction, Bandpass Filtering , and Dynamic Programming. *IEEE Trans. Acoustics, Speech , Signal Proc.*, ASSP-24(2):183–188, 1976.

[113] Saul A. Teukolsky Willam H. Press, William T. Vetterling and Brian P. Flannery. *The Art of Scientific Computing*. Cambridge University Press, 1992.

[114] J. Wise. The ecological approach to text visualization. In *JASIS*, 1998.

[115] Derick Wood. *Data Structures, Algorithms, and Performance*. Addison-Wesley Publishing Compagny, 1993.

[116] Jie Yang, Weier Lu, and Alex Waibel. Skin-color modeling and adaptation. In *ACCV (2)*, pages 687–694, 1998.

[117] Alfred C. She Yong Rui and Thomas S. Huang. Automated Region Segmentation Using Attraction-Based Grouping in Spatial-Color-Texture Space.

[118] Alfred C. She Yong Rui and Thomas S. Huang. Modified Fourier Descriptors for Shape Representation - A Practical Approach.

[119] H. Zhang and S. W. Smoliar. Developing Power Tools for Video Indexing and Retrieval. *Storage and Retrieval for Image and Video Databases*, 2185:140–149, 1994.

[120] Hongjiang Zhang, J. Y. A. Wang, and Yucel Altunbasak. Content-based video retrieval and compression: A unified solution. In *ICIP (1)*, pages 13–16, 1997.

[121] X. Zhou, J.D. Yates, and G. Chen. Using visual spatial search interface for WWW applications. In *Information Systems* **26**, pages 61–74, 2001.

[122] D. Zlobin and A. Roudometkine. Visual representation of document-oriented information on the web. In *Proceedings of the APL Berlin 2000 Conference*, pages 247–254, 2000.

Appendix - Additional figures for paper (D)

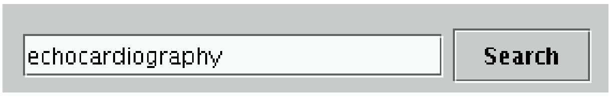


Figure 3: The initial search query

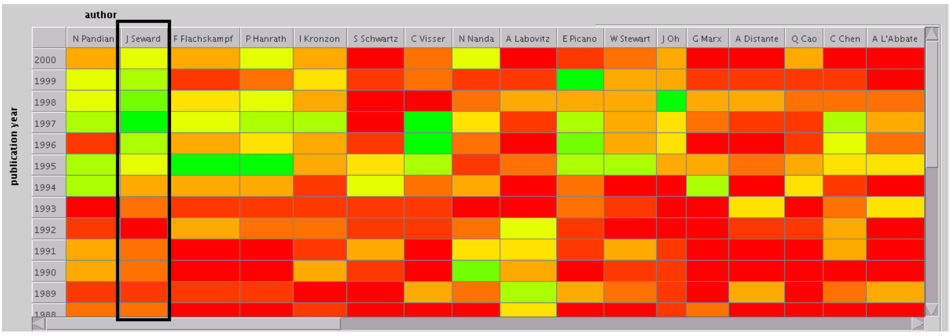
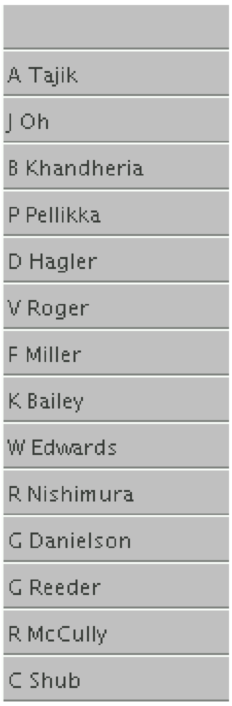


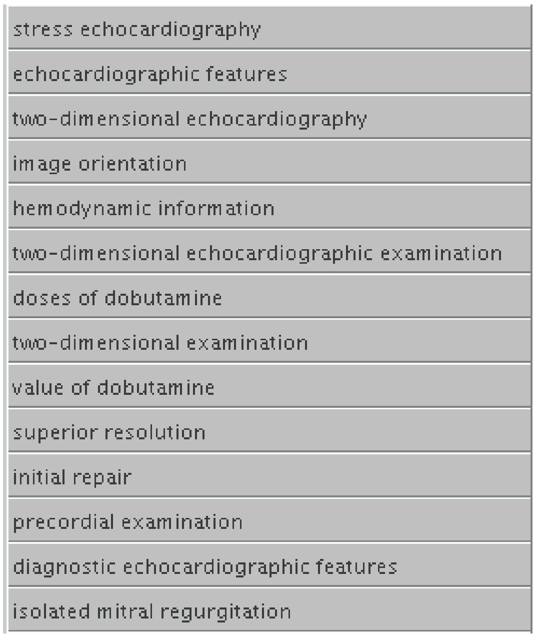
Figure 4: A heatmap displaying the search results from the query "echocardiography", viewed as the correlation between author and publication year.



A Tajik
J Oh
B Khandheria
P Pellikka
D Hagler
V Roger
F Miller
K Bailey
W Edwards
R Nishimura
G Danielson
G Reeder
R McCully
C Shub

Figure 5: J. Seward’s most frequent co-authors in echocardiography.

Appendix



stress echocardiography

echocardiographic features

two-dimensional echocardiography

image orientation

hemodynamic information

two-dimensional echocardiographic examination

doses of dobutamine

two-dimensional examination

value of dobutamine

superior resolution

initial repair

precordial examination

diagnostic echocardiographic features

isolated mitral regurgitation

Figure 6: J. Seward’s most frequent topics in echocardiography

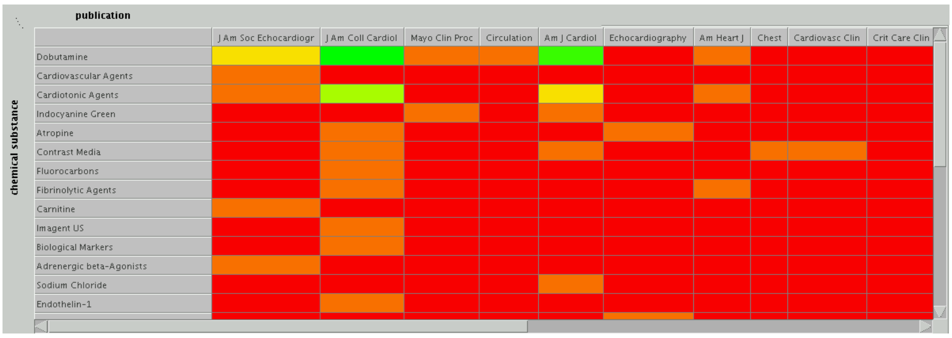


Figure 7: A heatmap showing the correlation between chemical substance and publication journal for J. Seward in the field of echocardiography

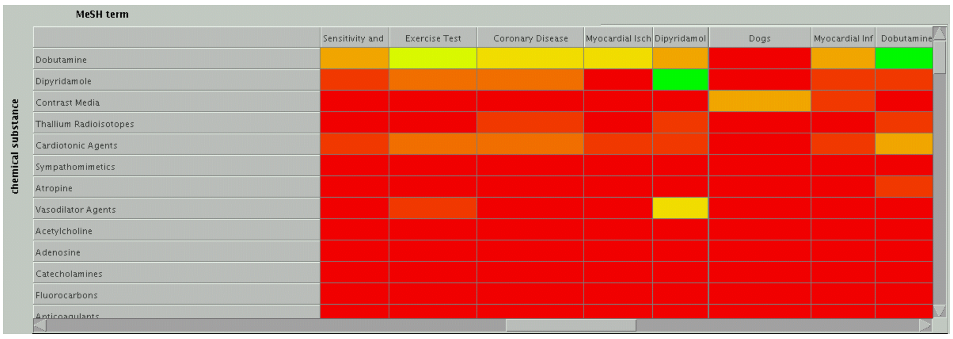
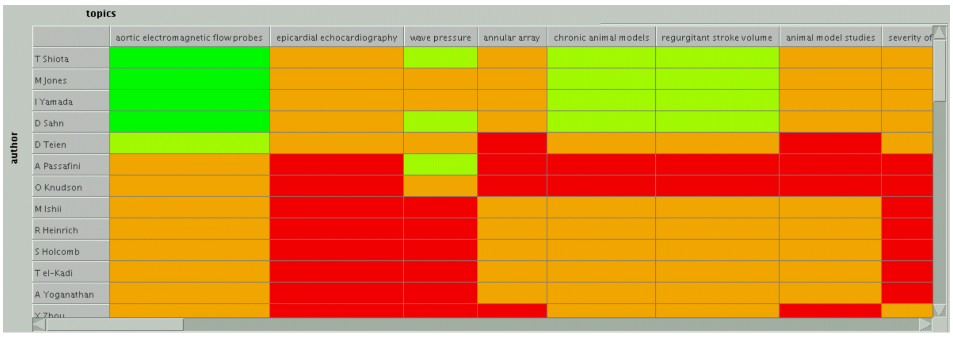


Figure 8: A heatmap showing the correlation of chemical substances versus MeSH-terms in the field of echocardiography.

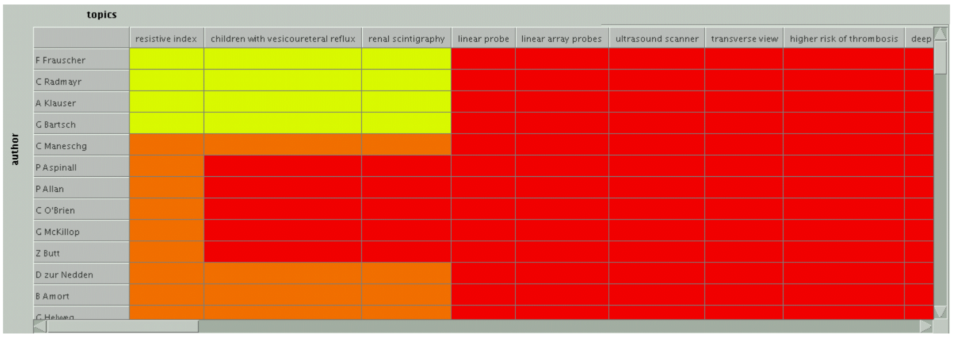
Appendix


topics

author

	aortic electromagnetic flow probes	epicardial echocardiography	wave pressure	annular array	chronic animal models	regurgitant stroke volume	animal model studies	severity of
T Shiota	🟩	🟧	🟨	🟧	🟨	🟨	🟧	🟧
M Jones	🟩	🟧	🟧	🟧	🟨	🟨	🟧	🟧
I Yamada	🟩	🟧	🟧	🟧	🟨	🟨	🟧	🟧
D Sahn	🟩	🟧	🟨	🟧	🟨	🟨	🟧	🟧
D Teien	🟨	🟧	🟧	🟥	🟧	🟧	🟥	🟧
A Passafini	🟧	🟥	🟨	🟥	🟥	🟥	🟥	🟥
O Knudson	🟧	🟥	🟧	🟥	🟥	🟥	🟥	🟥
M Ishii	🟧	🟥	🟥	🟧	🟧	🟧	🟧	🟥
R Heinrich	🟧	🟥	🟥	🟧	🟧	🟧	🟧	🟥
S Holcomb	🟧	🟥	🟥	🟧	🟧	🟧	🟧	🟥
T el-Kadi	🟧	🟥	🟥	🟧	🟧	🟧	🟧	🟥
A Yoganathan	🟧	🟥	🟥	🟧	🟧	🟧	🟧	🟥
X Zhou	🟧	🟥	🟥	🟥	🟧	🟧	🟥	🟧

Figure 9: A heatmap showing who is using Vingmed medical appliances in their research.


topics

author

	resistive index	children with vesicoureteral reflux	renal scintigraphy	linear probe	linear array probes	ultrasound scanner	transverse view	higher risk of thrombosis	deep
F Frauscher	🟨	🟨	🟨	🟥	🟥	🟥	🟥	🟥	🟥
C Radmayr	🟨	🟨	🟨	🟥	🟥	🟥	🟥	🟥	🟥
A Klauser	🟨	🟨	🟨	🟥	🟥	🟥	🟥	🟥	🟥
G Bartsch	🟨	🟨	🟨	🟥	🟥	🟥	🟥	🟥	🟥
C Maneschg	🟧	🟧	🟧	🟥	🟥	🟥	🟥	🟥	🟥
P Aspinall	🟧	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
P Allan	🟧	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
C O'Brien	🟧	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
G McKillop	🟧	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
Z Butt	🟧	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
D zur Nedden	🟧	🟧	🟧	🟥	🟥	🟥	🟥	🟥	🟥
B Amort	🟧	🟧	🟧	🟥	🟥	🟥	🟥	🟥	🟥
C Helweg	🟧	🟧	🟧	🟥	🟥	🟥	🟥	🟥	🟥

Figure 10: A heatmap showing who is using Acuson medical appliances in their research.

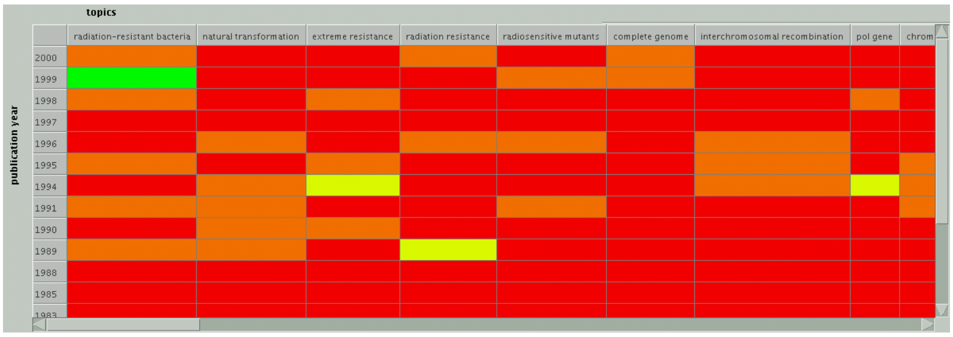


Figure 11: A heatmap the most recent publications topics about micro-organism “radiodurans”.

Beat extraction from digital music

Will Archer Arentz
Department of Computer and Information Science
Norwegian University of Science and Technology
willa@idi.ntnu.no

Abstract

A new algorithm for rhythmic beat extraction from digital music is proposed. The feature defined as the number of beats-per-minute (BPM) is a real number describing the rhythmic tempo of a song. A beat is most often a big-drum stroke. As most popular songs contain a determinable BPM, this value can be used in terms of content based retrieval (CBR). Two scenarios of using BPM for CBR of digital music are, search based on similarity, and search for songs of a specific type. Furthermore, one could imagine songs beeing searched by clapping hands, or making the beat sound in another way. The proposed algorithm uses amplitudic peaks and repetitiveness of such, from digitally sampled music, to determine the time between repetitive peaks. Thus, giving the number of beats per minute. The method proved to give accurate results as well as showing robustness to different kinds of music. When testing the method on 20 selected songs, the BPM value was found to be correct in 19 or 95% of the cases.

1 Introduction

In resent years, great amounts of music has become available on the Internet. To index this music for search based only on the filename and meta-data, limits the search engines capabilities with regards to separability as well as accuracy. In addition, it becomes sensitive to spam, since anyone with a web-page and some simple audio-software can make an audio-commercial appear as the latest song from a popular artist.

Lots of work has been done on content based image retrieval (CBIR) [14, 1, 10, 15, 11, 13], but the best known system for CBIR from large databases is QBIC [6], which allows an operator to specify various properties of a desired image.

Content based music retrieval (CBMR) is, however, a much more unexplored field with limitless possibilities. Furthermore, there are many parallels between CBMR and technics used in speech recognition [9, 5, 12, 4, 8]. AudioWorks has, for example, developed a system that can convert monophonic music to notes [3]. Also, [2] shows a technic for searching in live musical content. Nevertheless, extracting features is only half of the job. There must also be a way to make the extracted features searchable. The search must be sensible to the operator, and not too technical. It is therefor feasible to be able to search after a song based on rhythmic tempo in addition to meta-information like title and artist. Furthermore, for many types of music, the rhythm is what classifies what group it belongs to. Based on this, it is possible to group many songs based on their rhythmic characteristics.

This opens up for two immediate features in a search engine. First, one can search for a specific song in a defined group (rock, pop, etc.) that is automatically categorized. And second, the user can either select a song indexed in the search engine, or commit a song from his computer, followed by a request to the search engine for songs with similar rhythmic properties.

2 Table based BPM extraction

The proposed method is a simple algorithm for extracting the rhythmic property beats per minute (BPM), from any given song. Five different methods were tested by the author in this regard, but only the most successful one will be discussed in this paper. One of the unsuccessful methods were based on self-correlation, while the three others tried out various technics, using discrete Fourier transforms. Using self-correlation proved to give reasonable results in some cases, but the results was considerably poorer than for the table based method described in this paper. Also, the accuracy couldn't quite keep up with the table-method. The reason for the limited performance, is that some songs may change tempo in the middle of the song. Other times, a "fill-in" at some point can get the rhythm after this point out of sync with the initial rhythm. Thus, making both correlation and Fourier transformation unusable.

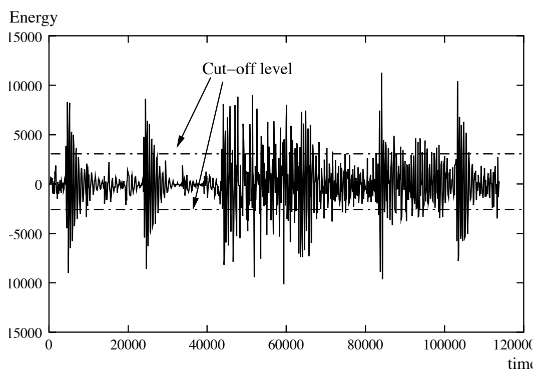


Figure 1: The amplitudic cut-off level (here: 2600)

Most music with an audible beat, has a tempo of 60 to 160 beats per minute. The assumption that any given song has a BPM-value within this range is used in the algorithm.

The sample-resolution for digital audio is very high when applied on musical content. This makes it possible to pinpoint events with high precision in time. For CD quality audio, the time-resolution is 1/44100 second. Likewise, the potential resolution of the BPM is also high. The frequencies are, however, merged together, so that determining what is unwanted high-frequency throb, and what is wanted low-frequency beats, can not be easily accomplished. However, because tones with lower frequency have more energy, they appear more dominant and are more easily identified.

Since the goal is to map the time-deltas between peaks, it is natural to try to isolate these. Figure 1 shows how it could be drawn a line at a given energy-level, and how deleting everything with less energy would result in a graph with few positive values, located in groups with the requested interval between (see figure 2). This was accomplished by extracting the 5% most energic amplitudes into a table.
Because different music has different characteristics and volume-level, each song corresponds to unique cut-off level. This cut-off level must be resolved in a unique way for each song. The solution is to go trough all samples and fill a table of the n largest elements together with their position (n=5% of the total samples, showed to be reasonable). In other words, a record of 2 fields, k (position) and e (energy/amplitude) (see table 1) with dimension n, is introduced.

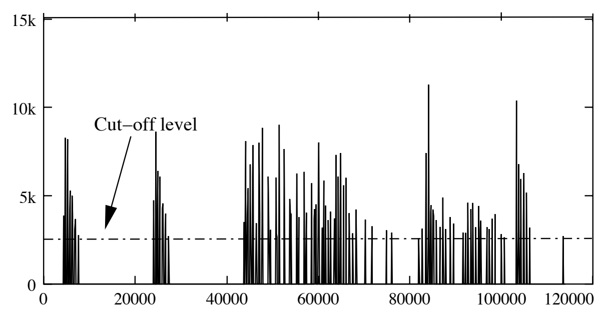


Figure 2: The remaining samples after cutoff

k	***e***
64598	14421
104446	13915
104450	13915
64602	13173
104445	13085
64594	12710
104440	12510
45678	12443
45677	12363
24751	12356

Table 1: Peak Table

The next thing to do is to parse this table for time-deltas with repetitiveness. Thus, another table (Delta Table) is generated. In the new table the Δk that matched the BPM-interval 70 ($\Delta k = 37800$) to 160 ($\Delta k = 16500$), together with the number of multiplums found after each Δks, is stored. The result is a data structure like table 2.
Finally, the record with the greatest amount of mulitiplums is selected, and the BPM calculated from Δk as shown in equation (1):

$$BPM = \max_{\Delta k_i}(\frac{60S}{\Delta k_i}), \tag{1}$$

where S is the sampling rate in Hz, and k_i is the i'th element in the Delta Table (table 2). For CD-quality audio S is set to 44100 Hz.

Δk	$\#multiplums$
19926	4
19927	4
19920	3
19921	3
19932	2
19928	2
19931	2
19927	2

Table 2: Delta Table

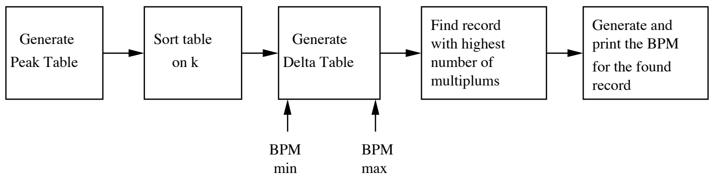

Generate Peak Table
Sort table on k
Generate Delta Table
Find record with highest number of multiplums
Generate and print the BPM for the found record
BPM min
BPM max

Figure 3: BPM determination by peak extraction

3 Experimental Results

3.1 The Evaluation Database

The goal of this project was to determine the BPM of any song containing an audible BPM. It was therefore natural to create a small database of songs to run the algorithm on, and thereby identifying the abilities and disabilities of each tried method. This database contained 20 songs listed in table 3. As shown, they are of different type and BPM-extraction difficulty. The BPM-values in this table were set from manually listening to music and counting beats. Due to the factor of human error, a diversion of ± 2 must be tolerated. The recognition difficulty level was set by listening to the song and judging from the clarity of the the beat.

	Song	**Artist**	**M**	**O**	**D**
1	I can't be with you	Cranberries	130	129	8
2	No need to argue	Cranberries	0	113	10
3	Fly me to the moon	Frank Sinatra	117	115.1	9
4	I got 5 on it	Luniz	87	87	3
5	Eye Of The Tiger	Survivor	109	109.3	7
6	Too cool too be cool	Soda	115	115	4
7	Far Away	Soda	128	128	3
8	Tell Me	Soda	132	132	4
9	Fortunes	Domination	137	137.4	1
10	Rejected	Allure	137	137.3	2
11	Runner	Vdm	138	137.9	1
12	1998	Binary Finary	139	138.8	2
13	Downbreaker	Luke cage	140	140.1	1
14	Blue	Eiffel 65	128	128	5
15	Opa opa	Antique	124	124	5
16	Dragonfly	Dj sakin and friends	140	140	4
17	Turn Around	Phats and small	132	132	5
18	Airwolf theme	Unknown	124	123.9	7
19	Anywhere is	Enya	97	96.8	8
20	Book Of days	Enya	120	120	9

Table 3: Result from BPM extraction

3.2 Performance of proposed method

From table 3, containing Song, Artist, M[1], O[2] and D[3], it can be seen that 19 of 20, or 95% of the songs, were recognized with correct BPM value. The only song incorrectly recognized was Cranberries' "No need to argue". This song did not have a determinable BPM by manually listening to it. The algorithm, however, found it to be 113.

Besides, returning good results, the algorithm is also quite fast. BPM-analysis of the whole evaluation-database was done in less than 10 minutes.

[1]M = Manually counted BPM
[2]O = Observed BPM from algorithm
[3]D = Difficulty to determine BPM

Furthermore, the CPU-time spent on each song was dependent on the songs difficulty level, from less than 10 seconds (most songs), to about 5 minutes if a song is difficult to analyze.

However, if the tests were carried out on music with slower BPMs, like slow waltz, the result would be considerably poorer. The purposed, algorithm used a limited range for searching in BPMs (60-160), even though some music-types, like slow waltz, may beat down to 30 BPM. The reason for the significant decrease in performance on a song beating on 30 BPM, is that 60, 90, 120 and 150 all are multiplums of 30, and therefore candidates to be the preferred result. Moreover, the highest value is often preferred. This problem could maybe be fixed by multiplying the decision making value of *number of multiplums* with a value, $w = 1/BPM_{candidate}$, for each of the BPM candidates.

4 Conclusion and Discussion

The proposed algorithm, showed great capabilities in terms of time accuracy in BPM extraction, with a usual inaccuracy of less that ± 0.1 BPM. Moreover, being able to extract the BPM from 95% of songs, is also a good achievement.

The biggest problem with the algorithm, is that it sometimes returns a BPM value for a song that doesn't have an audible beat. The reason for this is that frequencies are not separated in the time-domain, leaving high-frequent sounds with repetitiveness open for recognition by the algorithm. A suggested way to remove this is to run the whole audio-stream through a low-pass filter, removing the high frequencies. Nevertheless, this problem is very small.

The speed of the algorithm is also reasonably good. Depending on the song, some would need a deeper analysis to determine the BPM. This increases the CPU usage from less than 10 seconds upto 5 minutes.

As a conclusion to this project, beat detection using this simple algorithm can be used as one of several means for annotated digital music based on the content. Thus, making it searchable on more interesting terms for the user, namely on content.

References

[1] C. Zhou A. Kitamoto and M. Takagi. Similarity Retrieval of NOAA Satellite Images by Graph Matching. *Storage and Retrieval for Image and Video Databases*, 1908:60–73, 1993.

[2] Will Archer Arentz. MSc. Thesis, *Live Media Search*, 2000.

[3] AudioWorks Inc. *Waw2midi - Convert monophonic digital music to midi.* `http://www.audioworks.com/w2m/w2m.htm`.

[4] J.D. Ferguson. *Hidden Markov Analysis: An Indroduction in Hidden Markov Models for Speech.* Institute for Defense Analysis, Princeton, 1980.

[5] J.D. Markel and Jr. A.H. Gray. *Linear Prediction of Speech.* Springer-Verlag, 1976.

[6] W. Niblack R. Barber W. Equitz M. Flickner E. Glassman D. Petkovic and P. Yanker. The QBIC project: querying images by content using colour, texture and shape, 1993.

[7] Ken C. Pohlmann. *Principles of Digital Audio, 3rd edition.* McGraw-Hill, 1995.

[8] A.B. Poriz. Linear predictive hidden Markov models and the speech signal. *Proc. ICASSP 82*, pages 1291–1294, 1982.

[9] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals Of Speech Recognition.* Prentice Hall, 1993.

[10] J. R. Smith and S. F. Chang. *VisualSeek: A fully automated content-based image query system.* International Conference on Image Processing, 1996.

[11] S. W. Smoliar and H. Zhang. Content based Video Indexing and Retrieval. *IEEE Multimedia*, 1(2):62–72, 1994.

[12] G.M. White and R.B. Neely. Speech Recognition Experiments with Linear Prediction, Bandpass Filtering , and Dynamic Programming. *IEEE Trans. Acoustics, Speech , Signal Proc.*, ASSP-24(2):183–188, 1976.

[13] Alfred C. She Yong Rui and Thomas S. Huang. Automated Region Segmentation Using Attraction-Based Grouping in Spatial-Color-Texture Space.

[14] Alfred C. She Yong Rui and Thomas S. Huang. Modified Fourier Descriptors for Shape Representation - A Practical Approach.

[15] H. Zhang and S. W. Smoliar. Developing Power Tools for Video Indexing and Retrieval. *Storage and Retrieval for Image and Video Databases*, 2185:140–149, 1994.

Retrieving musical information based on rhythm and pitch correlations

Will Archer Arentz
willa@idi.ntnu.no

Magnus Lie Hetland
magnus@hetland.org

Bjørn Olstad
bjorn.olstad@fast.no

Abstract

Music Information Retrieval and Query-by-Humming systems has recently been given much attention. One of the reasons for this is the versatile of attractive applications that can be developed with these techniques. Imagine having a tune on your tongue, but you can not really place it. What is the name of the song and who is the artist? The proposed method is designed to help in pursuing such request. Both pitch and rhythmic information is utilized to determine the most closely matched song for any given theme. This also includes polyphonic songs, which may not contain the wanted theme in the main melody, but rather in one of it's accompaniments.

Due to humans poor ability to accurately reproduce a piece of music, whether it is caused by poor memory or poor music skills, such methods must be robust in a way that makes people's perception of similarity highly correlated with the distances calculated by the algorithm. Hence, our method has incorporated such measurements to ensure receiving the very best perceptual results. The proposed dynamic programming algorithm finds 93% of the wanted hits among the top-10 results when the timing is distorted with a standard deviation of 0.2 seconds and the pitch is distorted with a standard deviation of 0.3 notes, using only 7 note queries.

1. Introduction

Typically people do not learn all about a song the first time they hear it. Information such as title, composer and performer, are often learned in a much later stage of a persons relationship with a song. Furthermore, the human brain may easily forget such information after some time, though the melody still remains as something we frequently remember. It is therefore obvious that we only know the melody of most songs, and thereby have little chance to find out what we are humming as we wander around.

However, the recent development in music information retrieval (MIR) has brought forward several promising query-by-humming (QBH) systems that attempts to develop applications where the hummed or whistled theme of a song can be search through a large database, thus returning the closest matching song. The input interface to such a system may for example be a digital piano or simply a microphone, where the user is requested to whistle or sing the query into the microphone, before a signal processing system turns the audio into musical notes. Actually, the processed sequence of notes often has several errors in comparison to the original piece of music. A major reason for this is the humans poor music-reproduction-ability. For example the theme may sound similar, even if a few notes are missing, the key is wrong, the tempo is to slow and a long tone was reproduce as 3 shorter tones.

Despite all the differences mentioned here, a QBH system should take the given scenario into account and calculate the distance based on perceptible differences. The method proposed in this article has taken such information into account, and it can therefor provide a very robust and accurate search, even for a very short query.

1.1. Prior work

Much work has recently been done in the area of music information retrieval and query-by-humming systems. And some of it is summarized in Pickens survey of feature selection techniques in MIR [10]. Obviously, the most basic approach is to base the search on a monophonic sequence of notes, with their accompanying pitch and duration. Thus, simplifying the problem to one dimension. Moreover, the pitch is extracted and the duration is ignored, or vice versa. Both pitch and duration may be used in the final system, but as features they are in most work treated separately. [5, 3] are exceptions to this. Furthermore there is a question of using absolute or relative measures, where relative is the most popular because changes in tempo or transposition across keys does not significantly alter the music information expressed [4, 6, 5, 2].

The use of dynamic programming (DP) to calculate the distance between a set of query notes and all the notes in within all the songs in the database in a sliding window manner, has recently been attempted [13, 9]. Both projects have limited the research to only include the pitch feature, thus ignoring the information in the duration feature. However, they both claim to have fully operational query-by-humming systems. This paper shows how to include the timing feature as well as the pitch and thus filling the gap in prior work.

This paper is organized as follows: Section 2 explains the proposed algorithm, followed by Section 3 which goes through the experimental results. Finally Section 4 presents some concluding remarks, and points out some future interests of research.

2. The Algorithm

A tune or motif can be represented as a sequence of notes (frequencies), $z = (z_i)$, and a corresponding timestamp function $t(z_i)$ which gives the starting point for each note. Two tunes $x = x_1, \ldots, x_n$ and $y = y_1, \ldots, y_n$ are said to *match* if $x_i - x_{i-1} = y_i - y_{i-1}$ for $i = 2 \ldots n$.

When comparing a query motif $q = q_1, \ldots, q_m$ with a tune $s = s_1, \ldots, s_n$ from a music database, we would like to find a subsequence in s that matches q. This means that all notes in the query must be accounted for, but the matching notes from s need not be contiguous.

Formally, we define an *alignment* of a query q and a tune s as a strictly increasing sequence of indices $i = i_1, \ldots, i_m$. A matching alignment is an alignment i such that $s_{i_1}, \ldots, s_{i_m}$ matches q. Figure 1 shows a query q aligned with a tune s at index i.

Given the notion of a matching alignment, we can now define a dissimilarity measure between

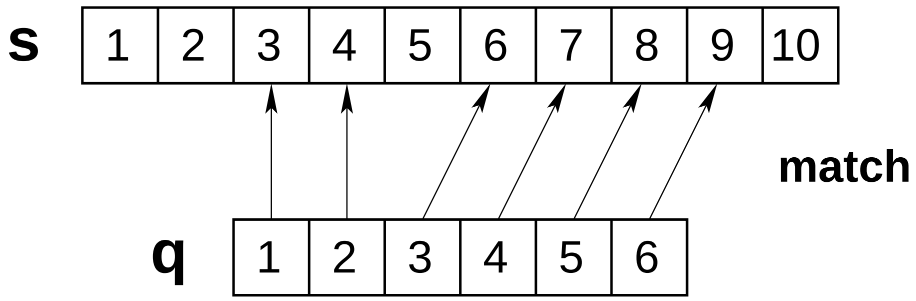


Figure 1: Match of query q and tune s with alignment $i = 3, 4, 6, 7, 8, 9$

a query q and a tune s for a specific alignment i as follows:

$$d(q, s, i) = \sum_{j=2}^{m} w(q_{j-1}, q_j, s_{i_{j-1}}, s_{i_j})^2, \tag{1}$$

where $w(q_a, q_b, s_c, s_d)$ represents the cost of pairing up the note pair (q_a, q_b) in the query with the note pair (s_c, s_d) in the tune. The cost function w is defined as

$$w(q_a, q_b, s_c, s_d) = \alpha\{t(s_d) - t(s_c)\} - \{t(q_b) - t(q_a)\}, \tag{2}$$

where α is a scaling factor, used to account for tempo differences between the two tunes, and $t(s_i)$ is the timestamp for the given note $s_i \in s$. The dissimilarity between a query q and a tune s can then be defined as

$$d(q, s) = \min_i \{d(q, s, i)\}. \tag{3}$$

For a given alignment i, $d(q, s, i)$ is minimized by choosing α as

$$\alpha = \frac{\sum_{j=2}^{m} (t(s_{i_j}) - t(s_{i_{j-1}}))(t(q_j) - t(q_{j-1}))}{\sum_{i=2}^{m} (t(s_{i_j}) - t(s_{i_{j-1}}))^2}. \tag{4}$$

Since the optimal value for α is given as a function of the alignment i, the optimization task in (3) becomes a matter of finding an optimal alignment.

Assuming, for now, that the value of α is known, the optimization may be expressed recursively as follows:

$$d(q_{1:a}, s_{1:b}) = \min_c \{d(q_{1:a-1}, s_{1:c}) + w(q_{a-1}, q_a, s_c, s_b)^2\} \tag{5}$$

where $q_{1:a}$ and $s_{1:b}$ are prefixes of q and s, of length a and b, respectively.

This equation may be solved iteratively, by dynamic programming [1]. The basic solution simply consists of constructing a two-dimensional array E of size $m \times n$ for storing the partial solutions, and iterating over the two prefix-lengths. See [11] for examples of the same technique applied to the problem of computing the Levenshtein distance (edit distance) and Euclidean distance under dynamic time warping. An application of the basic Levenshtein distance algorithm to timestamped data, similar to ours, can be found in [8].

There are two issues that must be addressed before such a dynamic programming solution can be implemented: We need to find the value of α, and we need to determine the allowable values for c in (5).

Preferably, α should be computed according to (4), but not all the required values are available during the stepwise computation of the dynamic programming algorithm. We approximate α associated with a given c when calculating $d(q_{1:a}, s_{1:b})$ with a recursive filter, as follows:

$$\hat{\alpha}(a,b) = \begin{cases} 1, & a = 1 \\ \beta \cdot \hat{\alpha}(a-1,c) + (1-\beta) \cdot \frac{t(q_a)-t(q_{a-1})}{t(s_b)-t(s_c)}, & a > 1 \end{cases} \tag{6}$$

The β value should be relatively high to avoid that the matching algorithm degenerates, accepting any rhythm variations due to note-to-note variations in $\hat{\alpha}$. We used $\beta = 0.85$ in our experiments.

The optimization parameter c will take on the values $b-1, b-2, \dots$. The c is simply the index of the note in the tune s that was matched by the previous note in the query q; if we restricted c to $b-1$, no extra notes would be allowed between the notes in the query. As a way of pruning the search for an optimal alignment we may restrict the values of c, either by setting an absolute limit on the number of extra notes are allowed in the tune between two query notes, or by placing an upper limit on the ratio

$$\frac{\hat{\alpha}(t(s_b)-t(s_c))}{t(q_a)-t(q_{a-1})},$$

as well as placing upper and lower limits on $\hat{\alpha}$ (for example, 2 and 0.5, respectively). We also retain the requirement of a matching alignment, that is, we only accept values of c that satisfy

$$s_b - s_c = q_{a+1} - q_a. \tag{7}$$

Assuming that the frequencies of each note are given as notes in the twelve-tone scale, octave information can be ignored by interpreting Equation 7 as modulo 12.

Pseudocode that illustrates the main ideas of the algorithm can be seen in Figure 2 on the following page. Where it should be noted that ε denotes a small value, for example 5 milliseconds. This is to adjust for inaccuracies occurring during manual entering of accords. The running time for the algorithm is $O(mn)$ for each song in the database.

3. Experimental results

The described algorithm was tested by interfacing it with a humming/whistling-to-notes converter, and letting several musicians and non-musicians perform searches. A similar test was performed when the humming-to-notes converter was replaced by a music-keyboard with midi-interface to a computer. Here, the test subjects could enter queries on the musical-keyboard and thus removing any inaccuracies that may appear in the humming-to-note conversion.

The results of these two tests are somehow subjective, and depends greatly on the test-subjects musical abilities. So even if the overall opinion of the test-subjects was that the system performed very well, the results are somehow unreliable as a performance measurement.

Therefore, to achieve a more accurate measurement of the systems actual performance, a different evaluation-scheme was set up. The idea was therefor to extract a theme from a random place in a song with some set length l. Then modify this theme before using it as a query on the database. The database used in the experiments contained 1564 monophonic songs. These songs were single channel midi files [7], which were used as ring tones on mobile phones. The midi files were found

```
**match**($q$, $s$)
  **for** $j \in 1 \ldots n$
    $E[1,j] := 0$
    $\alpha[1,j] := 1$
  **for** $i \in 2 \ldots m$
    **for** $j \in i \ldots n$
      $k_{max} := j - 1$
      $k_{min} := k_{max} - maxskip$
      **if** $k_{\min} < i - 1$
        $k_{\min} := i - 1$
      $\delta := t(q_i) - t(q_{i-1})$
      $k_{best} := -1$
      **for** $k \in k_{min} \ldots k_{max}$
        **if** $s_k = s_j + q_{i-1} - q_i$
          **if** $E[i-1,k] < \infty$
            $E_{cur} := E[i-1,k]+$
                $(\alpha[i-1,k] \cdot (t(s_j) - t(s_k)) - \delta)^2$
            **if** ($k_{best} \geq 0$ **and** $E_{cur} < E_{best}$) **or** $k_{best} < 0$
              $k_{best} := k$
              $E_{best} := E_{cur}$
      **if** $k_{best} \geq 0$
        $E[i,j] := E_{best}$
        **if** $t(s_j) - t(s_{k_{best}}) < \varepsilon$
          $\alpha[i,j] := \alpha[i-1,k_{best}]$
        **else**
          $\alpha[i,j] := \beta \cdot \alpha[i-1,k_{best}]+$
              $(1-\beta) \cdot (\delta / t(s_j) - t(s_{k_{best}}))$
      **else**
        $E[i,j] := \infty$
  $best := \infty$
  **for** $j \in m \ldots n$
    **if** $E[m][j] < best$
      $best := E[m,j]$
  **return** *best*
```

Figure 2: Pseudocode for the matching algorithm.

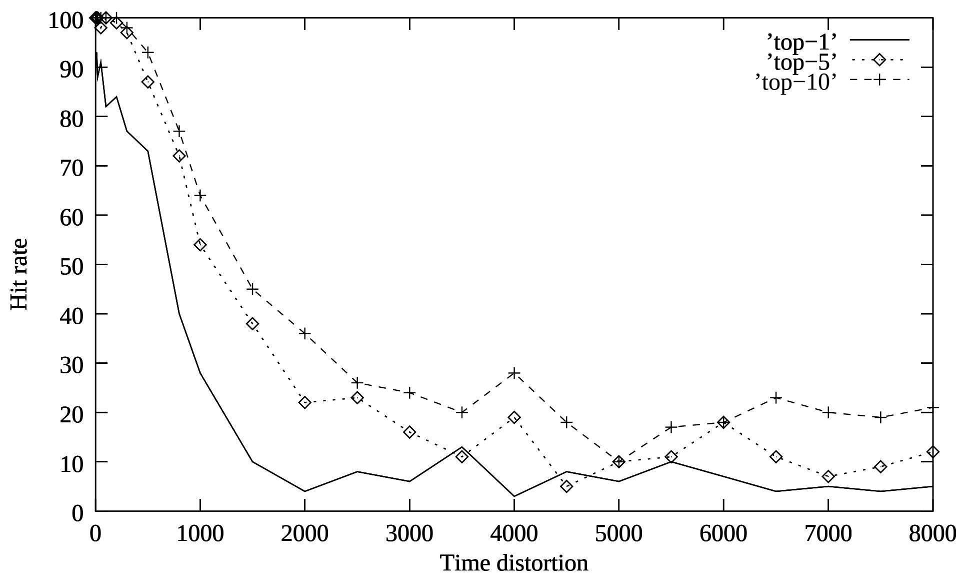


Figure 3: The algorithms hit rate as a function of time distortion(σ_{time})

by searching on the Internet. The proposed algorithm can handle polyphonic songs as well, but for simplicity it was chosen to use monophonic songs for the experiments in this paper, as the handling of polyphonic files is basically traversing each channel sequentially, like if they were separate files. In the database used in the experiments, several recordings were present for some of the songs.

When simulating the errors of humans as they try to recreate a musical piece it is important to understand that everybody has different ability to accurately reproduce right frequency at the time. Specially is the timing important in this context, where as most of the errors occur. By measuring the errors done by a few musicians trying to perfectly recreate a piece of music, we discover that the errors can be approximated to the normal distribution:

$$n(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Thus, we can use the Box-Muller algorithm to sample new values into a query $q = q_1, ..., q_l$, from the exact motif $m = m_1, ..., m_l$ extracted from the song.

The time of the query element q_i is set by

$$t(q_i) = [\sqrt{-2ln(U_2)}cos(2\pi U_1)]\sigma_{time} + t(m_i) \tag{8}$$

where U_1 and U_2 are uniformly distributed random numbers, and σ is set to some appropriate standard deviation. The greater the standard deviation, the greater the error introduced into the query. Figure 3 shows how the increasing standard deviation decreases accuracy in the result. For each chosen standard deviation, one hundred queries were run, and the results measured as being the top match, among the top 5 or among the top 10 result hits. Each query had a size of 10 notes, which usually suffice to get a good result. Throughout the experiments it was chosen to use 100 queries to determine the hit rate. That means the percent of the queries that found it's match amount the top-n (usually top-10) results from the 100 queries run through the search algorithm.

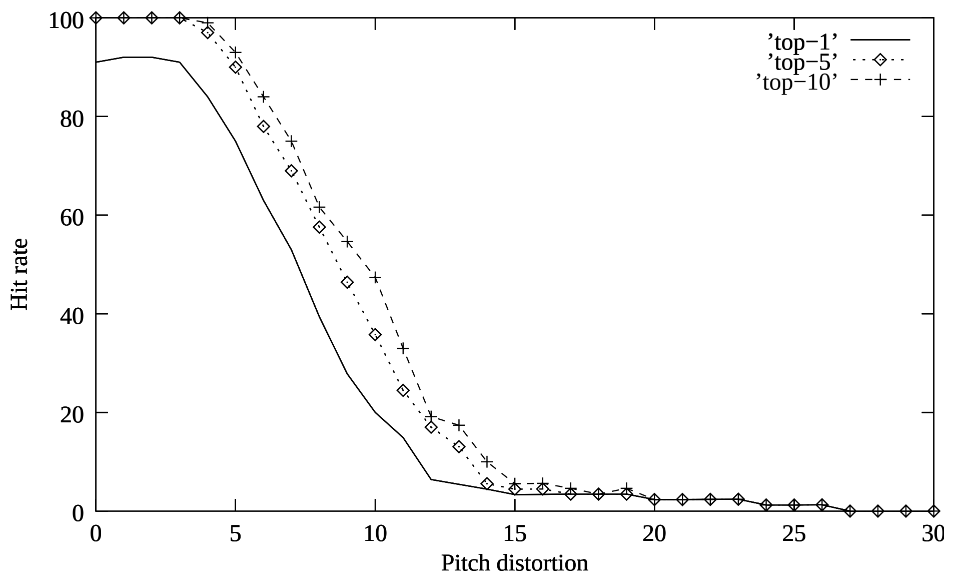


Figure 4: The algorithms hit rate as a function of pitch distortion(σ_{pitch})

Most likely, the number of queries used for calculating the hit rate was much too small, and hence allowing some degree of error into measured results.

Furthermore, since some songs are entered into a computer rather than played on an instrument, several songs may have the same sequence of 10 notes and thereby leaving additional potential for errors. And, more importantly, some songs had more than one entry in the database, but with different name. This makes it impossible for the algorithm to understand which one we are searching for. Thus, the top-1 measure is not very reliable, so the top-5 and specially the top-10 should rather be given the attention. It can be seen from the figure that, even with a standard deviation of 8000 milliseconds, that is 8 seconds, the requested result is among the top-10 in 21% of the queries.

Figure 4 shows how the algorithm performs as the pitch is distorted using the Box-Muller method as described in (8). Also here the query consisted of 10 notes and 100 queries were tested for each choice of σ_{pitch}. The values describing pitch change (x-axis) express one step change multiplied with 10. That means, that if a C equals 600, then a C# equals 610. It can be seen that the penalty for making a pitch mistake, is higher than the penalty for making a severe rhythmic mistake. This is due to the fact, that the human understanding of pitch mistakes are perceived as significantly more serious than timing mistakes [12]. Thus, our algorithm should act on the same principles.

By adding distortion both to the pitch and to the time, the hit rate response of the algorithm can be seen from figure 5. Here a “hit” is defined as included in the top-10 matches returned from the algorithm. For each point in the graph 100 queries consisting of 10 notes were run through the algorithm.

However, the hit rate is also dependent on the number of notes in the query. In figure 6 it can be seen how the function improves it’s ability to find the right results as the query length increases. In the figure distortion was introduced in both the time ($\sigma_{time} = 500$) and the pitch ($\sigma_{pitch} = 5$) data.

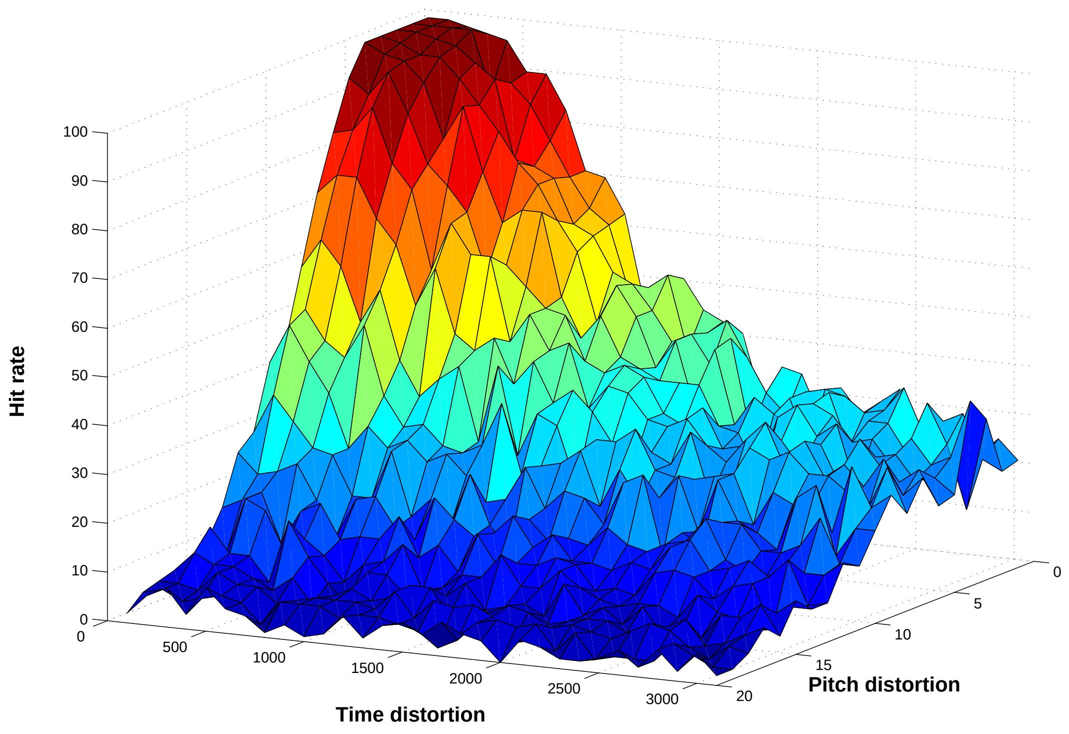


Figure 5: The algorithms hit rate as a function of time and pitch distortion

Table 1 shows how the length of the query has impact on the hit rate, in some typical distortion scenarios. It should be noted that even with as few as 5 notes in the query, good results can be achieved with medium distortion.

4. Conclusion

This paper has presented a method for accurately searching in music databases with short queries, allowing distortion in both pitch and time domain. The dynamic programming algorithm finds 93% of the wanted hits among the top-10 results when the timing is distorted with a standard deviation of 0.2 seconds and the pitch is distorted with a standard deviation of 0.3 notes ($\sigma_{pitch} = 3$), using only 7 note queries.

The algorithm has currently a linear runtime, so future work includes researching how to index the search so that the speed can be increased. Furthermore, a generalized framework for applying and adjusting the algorithm for other applications, also deserves further investigation.

All in all the proposed method has shown to be very successful, and it has generated much commercial interest, as well as being incorporated into several applications. This is mainly due

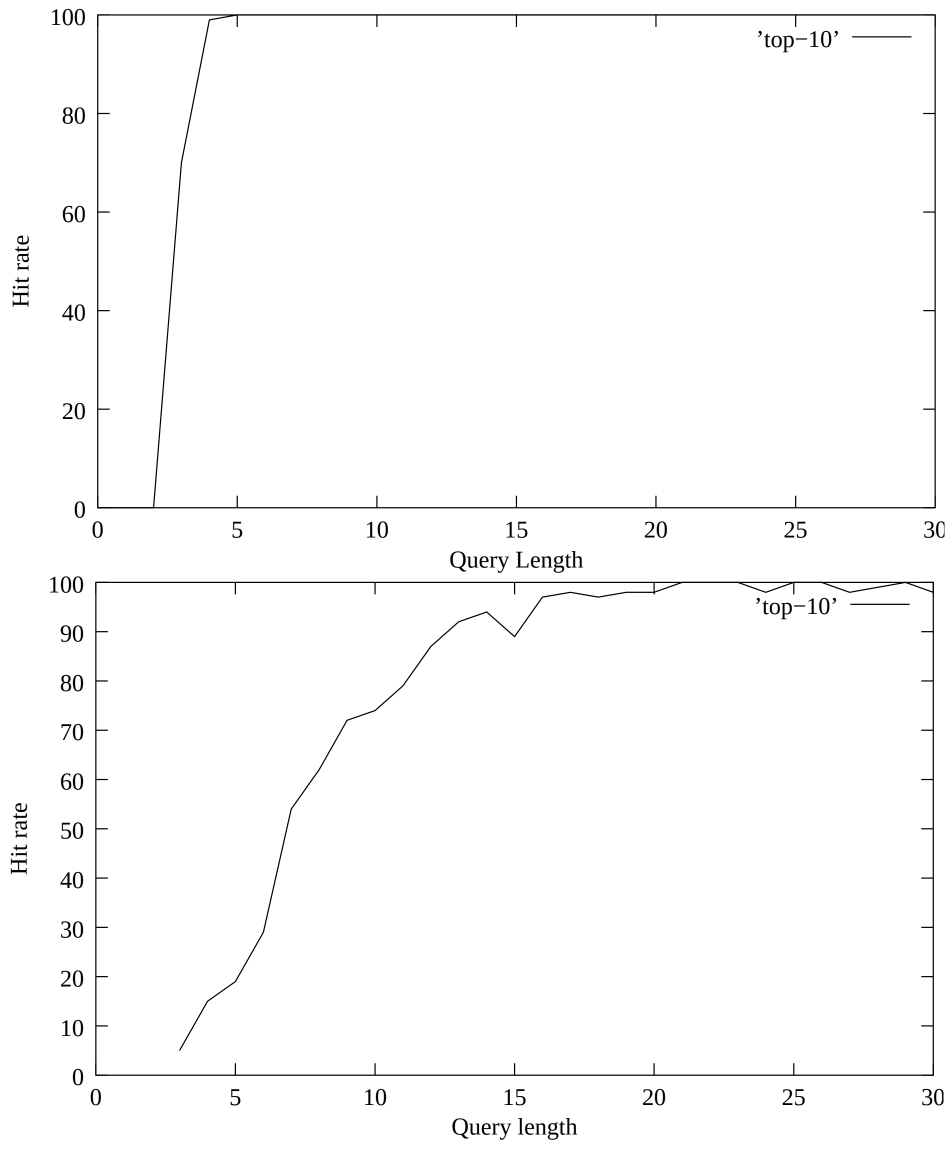


Figure 6: The algorithms hit rate as a function of query length (top:$\sigma_{time} = 0, \sigma_{pitch} = 0$, bottom:$\sigma_{time} = 500, \sigma_{pitch} = 5$)

Table 1: The hit rate (% in top-10 from tested 100 queries) as function of the query length and some typical distortions

Query length	**3**	**5**	**7**	**10**	**15**	**20**	**25**
$\sigma_{time} = 0$ $\sigma_{pitch} = 0$	76	100	100	100	100	100	100
$\sigma_{time} = 50$ $\sigma_{pitch} = 1$	43	93	100	100	100	100	100
$\sigma_{time} = 100$ $\sigma_{pitch} = 2$	22	89	100	100	100	100	100
$\sigma_{time} = 200$ $\sigma_{pitch} = 3$	15	63	93	98	100	100	100
$\sigma_{time} = 300$ $\sigma_{pitch} = 4$	16	37	74	94	100	100	100
$\sigma_{time} = 400$ $\sigma_{pitch} = 5$	5	29	53	78	96	100	100
$\sigma_{time} = 500$ $\sigma_{pitch} = 6$	4	22	47	59	96	95	97

to the algorithms ability to differentiate between themes in the same way as the human perception does.

References

[1] Richard Bellman. *Dynamic Programming*. Cambridge Studies in Speech Science and Communication. Princeton University Press, 1957.

[2] S. Blackburn and D. D. Roure. A tool for content-based navigation of music. In *Proceedings of ACM International Multimedia Conference (ACMMM)*, 1998.

[3] A. L.P. Chen, M. Chang, and J. Chen. Query by music segments: An efficient approach for song retrieval. In *Proceedings of IEEE International Conference on Multimedia and Expo.*, 2000.

[4] Asif Ghias, Jonathan Logan, David Chamberlin, and Brian C. Smith. Query by humming: Musical information retrieval in an audio database. In *ACM Multimedia*, pages 231–236, 1995.

[5] K. Lemstrom, P. Laine, and S. Perttu. Using relative interval slope in music information. retrieval. In *Proceedings of International Computer Music Conference*, pages 317–320, 1999.

[6] A. Lindsay. Using contour as a mid-level representation of melody, 1994.

[7] G. Loy. Musicians make a standard: The midi phenomenon. In *Computer Music Journal 9(4)*, pages 8–26, 1985.

[8] Heikki Mannila and Pirjo Ronkainen. Similarity of event sequences (revised version). In *Proceedings of the Fourth International Workshop on Temporal Representation and Reasining, TIME 97*, pages 136–139, 1997.

[9] Steffen Pauws. Cubyhum: A fully operational query by humming system. In *Proceedings of ISMIR*, pages 187–196, 2002.

[10] Jeremy Pickens. A survey of feature selection techniques for music information retrieval.

[11] David Sankoff and Joseph Kruskal, editors. *Time Warps, String Edits, and Macromolecules : The Theory and Practice of Sequence Comparison*. CSLI Publications, reissue edition, 1999.

[12] I. Shmulevich, O. Yli-Harja, E. Coyle, D. Povel, and K. Lemstrom. Perceptual issues in music pattern recognition - complexity of rhythm and key finding, 1999.

[13] Jungmin Song, So Young Bae, and Kyoungro Yoon. Mid-level music melody representation of polyphonic audio for query-by-humming system. In *Proceedings of ISMIR*, pages 133–139, 2002.

Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Computer Vision and Image Understanding 94 (2004) 295–310
Computer Vision and Image Understanding
www.elsevier.com/locate/cviu

Classifying offensive sites based on image content

Will Archer Arentz* and Bjørn Olstad

Department of Computer and Information Science, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway

Received 1 December 2002; accepted 29 October 2003

Abstract

This paper proposes a method for helping to identify adult web sites by using the image-content as means of detecting erotic material. The image content is classified by investigating probable skin-regions, and extracting their feature vectors. These feature vectors are based on color-, texture-, contour-, placement-, and relative size-information for a given region. The importance of the different elements in the feature vector is determined by a genetic algorithm. For each picture, the algorithm gives the probability that a certain picture has erotic content. By mapping all the images in a web site, and running the image-based classifier on the whole collection, we were able to set up a histogram of images with regards to the log-likelihood of erotic content for each image. Hence giving a good overview of the web site's content and at the same time leaving room for errors in the image-based classifier.

The algorithm proved to be quite successful in our tests where all 20 sites where classified correctly. The image-based classifier is able to properly identify 89% of the evaluation images at an average processing speed of 11 images per second.

Although this experiment focused on classifying adult web sites, small alterations to the system can be done, enabling classification of other kinds of images and web sites.
© 2003 Elsevier Inc. All rights reserved.

Keywords: Object recognition; Erotica/pornography; Internet; Web site classification; Color; Image retrieval

* Corresponding author.
E-mail addresses: willa@idi.ntnu.no (W.A. Arentz), Bjorn.Olstad@fast.no (B. Olstad).

1077-3142/$ - see front matter © 2003 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2003.10.007

1. Introduction

The recent development of multimedia availability on the Internet has enabled Internet users around the world to access enormous amounts of video-, picture-, and audio-files. However, this has also lead to new business-opportunities for the adult-industry. Furthermore, due to this industry's aggressive marketing, it has become very difficult to avoid stepping into an adult site from time to time, be it through web-surfing, spam-mail, or search results from a web search-engine. This may be very upsetting to some people, especially parents with Internet-surfing children. A system preventing the browser to enter such sites has been requested on several occasions. This problem is especially evident in web search engines, where a company often has some responsibility for the quality of the search results.

The solution is to use a content-based image-retrieval algorithm to identify naked or scantily dressed people in images. This is not a new idea; Forsyth and Fleck [8] managed to create a fully automated system for detecting such pictures. However, their system was much too slow to be used as part of the data gathering process in a search engine, where speed is of the essence. Currently existing systems use advanced text-analysis and linguistics to determine if a web-page is offensive or not. Unfortunately, some adult sites just have a listing of un-suspicious-looking file-names, while other sites have all the text incorporated into bitmaps displayed on the web page. Hence, image understanding is the only way to determine what kind of site it actually is.

When dealing with detection of nudity, it is important to have a good method of recognizing skin regions. Much work has been done in this regard.

Detection of human skin and the effect different cameras, light-settings, human race, and color spaces has on the recognition process is discussed in [2,4,10,12,13,19,20,24]. Furthermore, [8,17] also use texture information as a component in the skin detection. As a third component in object recognition, many have looked at the shape of the object as a last stage in the information gathering process [1,7,11,15,18]. Veltkamp and Hagedoorn [23] have written a survey of different state-of-the-art shape matching methods.

With a set of features describing the image, such as color, texture, and shape for segmented objects, it is possible to build a joint feature vector, where all the information is stored. And thereafter, using a training-method for weighting the importance of each element in the vector. Using a genetic algorithm (GA) is one way to do this, and several people have attempted to use GA in similar feature vector based image retrieval schemes [5,6,16,21,22].

Classifying Internet sites is currently being done by several search engines. The most famous is perhaps Google, where PageRank [14] is used to classify the importance of web pages and web sites based on the hyperlink information. Even though there has been much work done on classifying documents, little published work has been done in labeling the actual web sites. This is what we were aiming to do, based on the content of images on the site.

In the next section of this paper, we explain the design of the content-based image retrieval algorithm, while the site classification is discussed in Section 3. Furthermore, Section 4 reviews the results of our experiments, before some ideas about future work are revealed in Section 5, followed by a final conclusion in Section 6.

2. The image content examiner (ICE)

Identifying regions with skin in the images is the key element in identifying nudity. Also, since we want fast processing, filtering out all non-skin regions can significantly reduce the amount of data processed. Although there is a lot of nudity on the web, most of the images do not belong to a this category, and would hence be trivially rejected as non-offensive.

According to Shin et al. [19] YCbCr is the best 2D (CbCr) color space for skin detection of the nine tested. However, it is argued that the RGB color space gives the best performance in 3D, and surpasses the performance of CbCr. Nevertheless, since we wish to have a fast system, a smaller data representation implies faster processing. Thus, we choose CbCr to be the color representation in our system.

The initial filter, which removes non-skin pixels, is simply based on rejecting pixels outside a given range of Cb and Cr in the CbCr color space. The pixel range was determined by generating a color histogram for the Cb and Cr color components, using a database of 500 skin-dominant images. As can be seen from Fig. 2, Cb values from 78 to 135 and Cr values from 85 to 185 have very high readings, thus suggesting the use of these values as initial filter thresholds. We choose to use a rather large range at this stage, allowing some non-skin pixels to pass through the filter, since we wish to accept as much of the probable skin pixels as possible.

The pixels that remain after the initial filtering are grouped into connected regions and labeled, using a simple recursive labeling algorithm. Generally, recursive algorithms are considered slow, but the use of pointers in an efficient implementation made this process quite fast.

The labels are stored in the array L, and the array P contains the number of pixels in a given region or object. This array is sorted so that the N largest objects may be chosen and further analyzed. In our experiments we set $N_{\max} = 10$.

Furthermore, an overview of the image processing system can be found in Fig. 1, while the training process is depicted in Fig. 4.

2.1. The composite object feature vector

The next stage in the analysis is to extract a feature vector for each of the objects. The feature vector is composed of information about the color, texture, and shape of the object. In addition some information about placement and size is included. The composition of the color feature, is simply two histograms of Cb and Cr pixel-values.

Next, the texture feature is extracted based on the complexity curve [3]. This method has the advantage of being very computationally efficient, and Baheerathan et al. shows that the method gives a robust second-order description of the textures. The actual implementation is done by calculating the difference between a pixel and

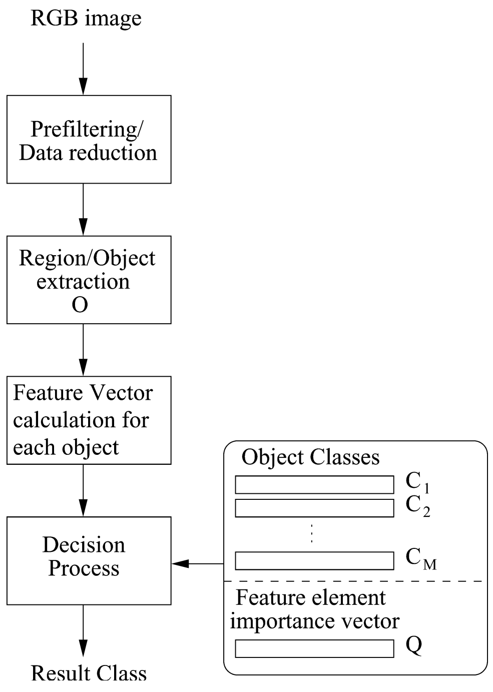

RGB image
Prefiltering/ Data reduction
Region/Object extraction O
Feature Vector calculation for each object
Decision Process
Result Class
Object Classes
C_1
C_2
C_M
Feature element importance vector
Q

Fig. 1. An overview of the image analysis process.

its right neighbor in the luminance picture, and entering these values into a histogram. This histogram is then entered into the composite feature vector. High intensity in the lower part of this histogram would therefore, suggest smooth transitions, while high intensity in the higher areas would suggest a noisy picture. Both the color and texture section of the feature vector is normalized by taking the feature element and dividing it by the total number of pixels in the object.

The third piece of information extracted from an object is a description of the object's shape. This task is accomplished by clockwise tracing the outer borders of the object and storing the distance from the object's centroid. The sequence of distances are stored in an array and normalized so that the sum of distances to the center is constant, before finally applying the Fast Fourier Transform. The 28 most significant energy-coefficients from the Fourier transform are then added to the object's total feature vector.

The three final components stored in the feature vector are the relative x- and y-positioning of the object's centroid, and the relative size of the object. This produces a feature vector of 801 real number-elements for each object, as can be seen in Fig. 3.

2.2. Training by genetic algorithms

The preceding chapters explained how to calculate a set of feature vectors for some skin-dominant objects extracted from an image. Even though the composite

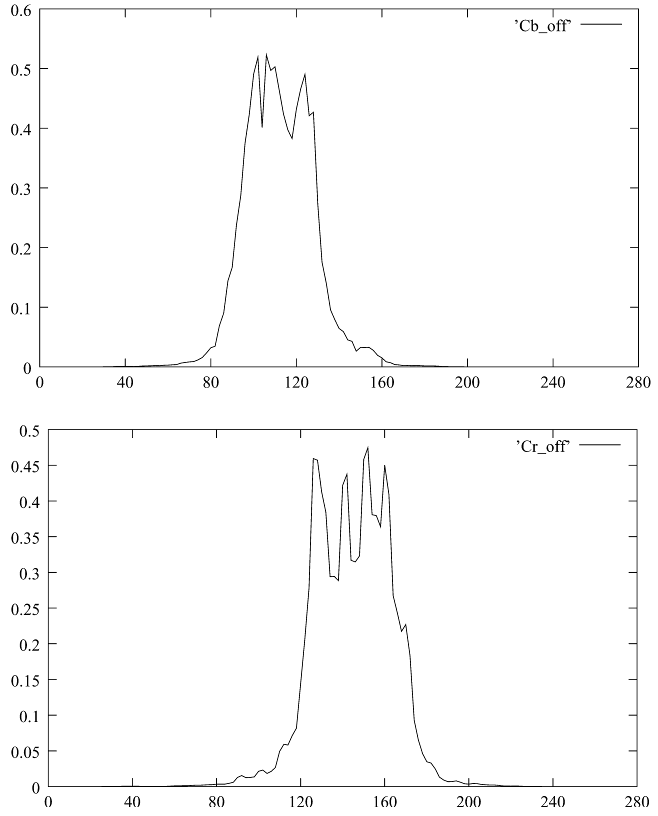


Fig. 2. Histograms of the pixels in the Cb (top) and Cr (bottom) color space.

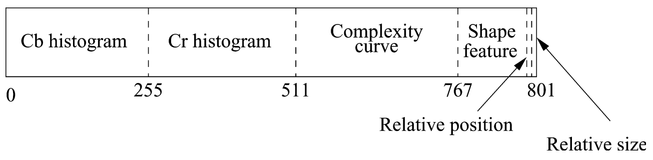


Fig. 3. The composition of the feature vector.

feature vector contains much information, we know little about the importance of the different elements. Hence, it is necessary to find a weighting scheme for this vector.

Furthermore, a set of objects classes **C** should be defined as reference, so that the retrieval process can be done by classifying which class each object belongs to. Then the probability that an image is offensive, can be calculated by accumulating the probabilities of the object-classes to which the image's objects belong. In the experiments 30 object classes were used.

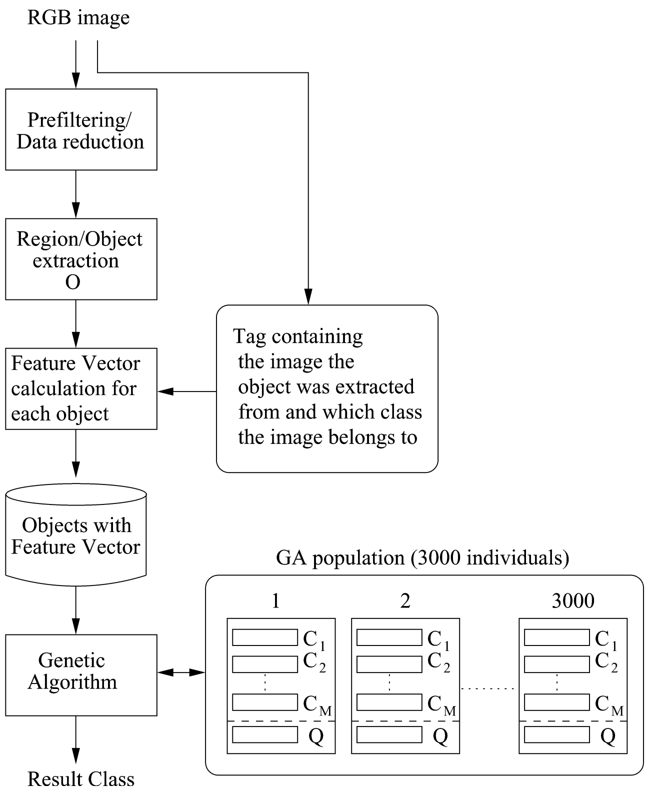

RGB image
Prefiltering/ Data reduction
Region/Object extraction O
Feature Vector calculation for each object
Tag containing the image the object was extracted from and which class the image belongs to
Objects with Feature Vector
Genetic Algorithm
Result Class
GA population (3000 individuals)
1
2
3000
C1
C2
CM
Q

Fig. 4. An overview of the training process.

Choosing the object classes could of course be done by some kind of un-supervised clustering algorithm like for example c-means clustering. However, in an attempt to find a more optimal distribution in the feature space, it was decided to use a genetic algorithm to find these classes, as well as finding the importance of the different elements in the feature vector. This last part was done by defining a vector Q with the same dimension as the feature-vector for an object O. The Q-vector is initialized with random values from 0.0 (not important) to 1.0 (most important). This vector is multiplied with the object's feature vector when calculating the distance between the object class C_j and the object O^k. This can be written as a function

$$d(O^k, C_j, Q) = \sum_i ((O_i^k - C_{ji})Q_i)^2, \qquad (1)$$

where $d(O^k, C_{j_k}, Q)$ is minimized with respect to the class C_j, so that the object O^k is assigned to the class which minimizes the distance between object and class. We write this as

$$C_{j_k}(O^k) = \underset{C_j \in \mathbf{C}}{\operatorname{argmin}}\, d(O^k, C_j, Q). \qquad (2)$$

A set of objects $O^1 \cdots O^k \cdots O^N$ is derived from an image I, and when processing images on a fully trained system, an object O^k is mapped to an object class $C_{j_k}(O^k)$ where j_k is the class number matching the kth object.

To decide whether an image is offensive or not it is sensible to calculate a statistical probability that an image is offensive based on the gathered information. In this regard A represents the offensiveness, such that the probability that a given object class C_j represents offensive material given the image I is calculated as in (3). The training objects are either from an offensive image or from a non-offensive image. All images are manually classified into the two groups before the training process.

$$P(A|C_j) = \frac{\text{number of offensive images with objects belonging to } C_j}{\text{number of images with objects belonging to class } C_j} \tag{3}$$

We define the probability of offensiveness in an object as in Eq. (4), where the right hand side is the probability defined in Eq. (3). In the manner shown in Eq. (2), the distances from an object to all the M object-classes are calculated to find the class with the shortest distance. This class is then chosen to represent the object. We set the probability of the object being offensive equal to the probability of the class, to which the object is assigned, being offensive. This is expressed as

$$P(A|O^k) = P(A|C_{j_k}(O^k)). \tag{4}$$

By applying Bayes formula, we calculate the probability that an image is offensive as shown in Eq. (5). Since $P(C_{j_1}(O^1), \ldots, C_{j_N}(O^N))$ is the probability that a set of classes $C_{j_1} \ldots C_{j_N}$ occur in an image, the denominator is constant and can therefore be neglected retaining proportionality.

$$\begin{aligned} P(A|O^1, \ldots, O^N) &= P(A|C_{j_1}(O^1), \ldots, C_{j_N}(O^N)) \\ &= \frac{P(C_{j_1}(O^1), \ldots, C_{j_N}(O^N)|A) \cdot P(A)}{P(C_{j_1}(O^1), \ldots, C_{j_N}(O^N))} \\ &\propto P(C_{j_1}(O^1), \ldots, C_{j_N}(O^N)|A) \cdot P(A) \end{aligned} \tag{5}$$

The objects extracted in the preprocessing are mutually exclusive. It is hence reasonable to assume that the object classes are independent. We will use this assumption for finding the discrimination function. Based on this we can write

$$P(C_{j_1}(O^1), \ldots, C_{j_N}(O^N)|A) = \prod_{k=1}^{N} P(C_{j_k}(O^k)|A), \tag{6}$$

which implies

$$\log(P(C_{j_1}(O^1), \ldots, C_{j_N}(O^N)|A)) = \sum_{k=1}^{N} \log(P(C_{j_k}(O^k)|A)). \tag{7}$$

$P(A)$ is the probability of offensiveness which would be 0.5 if the amount of offensive data equal the amount of non-offensive. A large investigation of the ratio of offensive versus non-offensive data on the Internet, may provide a reasonable value for $P(A)$. Nevertheless, the value is a constant and since we ultimately only are interested in the proportional equality we remove this term from the equation. Hence, we can join Eqs. (5) and (7), which gives us the log-likelihood of an image being offensive as

$$\log(P(A|C_{j_1}(O^1), \ldots, C_{j_N}(O^N))) = \sum_{k=1}^{N} \log(P(C_{j_k}(O^k)|A)) + K, \tag{8}$$

where K is a constant.

Given a set of objects extracted from an image and a set of object classes, we can now use (8) to calculate a value describing the offensiveness of the image. However, there is still a matter of setting the threshold for what values should be accepted as offensive. This value β separates the two content types, and can be found by calculating $\log[P(A|\mathit{Image})] = \log[P(A|C_{j_1}(O^1), \ldots, C_{j_N}(O^N))]$ for a large set of manually classified offensive and non-offensive images. An initial value for β is chosen, and for all $\log(P(A|\mathit{Image})) > \beta$ the images are marked as offensive. Then the number of erroneous classifications ϵ are counted. Different values of β are then tried out to minimize this error.

Both the vector Q and all object classes in C are found by using a genetic algorithm that attempts to maximize the ability to separate between offensive and non-offensive objects. As mentioned earlier the Q-vector as well as the C-vectors all consists of 801 real value elements. Since 30 classes are used in the experiment, this means that 24,831 values must be set by the genetic algorithm. This is a very large amount of values and other approaches were considered to replace the GA. In this regard one other method was attempted, namely simulated annealing [9]. Unfortunately this approach was too slow to be useful.

The first step of the GA is to set the initial values of the each individual. Since each individual consists of 24,831 values and we use 3000 individuals in our population, 74,493,000 elements are initially set to random values between 1.0 and 0.0. A large population is chosen because the versatile distribution provides some quite fit individuals, in the sense that they achieve very high initial fitness. This seems more powerful in reaching high fitness than running a high number of generations. Furthermore, a multi-point crossover method is chosen to improve the handling of the large data set. The crossover sections are set by running through the whole set of elements in an individual, marking the elements as a crossover start- or stop-point with a probability of 5%. Mutations are performed on an element with a probability of 0.5%. Upon a mutation the element is set to a new random value. The number of generations run is 50, although the the last eight generations provide little improvement in fitness. We used tournament selection as the method for selecting individuals in our GA implementation. The best individual from the previous generation is automatically forwarded to the next generation, while the rest of the population is set by tournament selection of the 50% best individuals from the previous generation.

The GA fitness calculation is the most crucial part of the training process. The value returned from the fitness function measures the ability of the image processing method to separate between offensive and non-offensive images. Arguments to the fitness function are the Q-vector and the class vectors. These are first determined by the GA, before being evaluated by the fitness function. Further arguments to the fitness function are a selection of offensive and non-offensive training images. As shown in

```
**fitness**($I_{offensive}$[],$I_{non-offensive}$[],$C$[][],$Q$[])
begin
  $O_{offensive}$:= all objects extracted from $I_{offensive}$
  $O_{non-offensive}$:= all objects extracted from $I_{non-offensive}$
  for all objects O find the closes class $C_j$ so that
    $C_{j_k}(O^k)$ denote the class $C_j$ containing the object $O^k$
  for each class $C_j$ make a list $L_{C_j}$ containing
    all objects belonging to the class $C_j$
  for all lists
    for all elements in $L_{C_j}$
      replace all objects O with a reference to the image they were
      extracted from I
      calculate $P(A|C_j) = \frac{\text{number of offensive images in } L_{C_j}}{\text{total number of images in } L_{C_j}}$
  for all images $I[l]$
    extract all objects for I
    $P(A)_{I[l]} := \sum_{k=1}^{N} log(P(C_{j_k}(O^k)|A)$
  set a value $\beta \in 0 \ldots 1$
  while did not investigate all possible $\beta$
    $\epsilon := 0$
    for all $P(A)_{I[l]} < \beta$
      if any image I[l] is offensive
        $\epsilon = \epsilon + 1$
    for all $P(A)_{I[l]} \geq \beta$
      if any image I[l] is non-offensive
        $\epsilon = \epsilon + 1$
    if $\epsilon < \epsilon_{opt}$ then $\epsilon_{opt} = \epsilon$
  return total number of images $- \epsilon_{opt}$
end
```

Fig. 5. The genetic algorithms fitness function.

Fig. 5, every fitness calculation involves loading a large set of objects from a training-image into memory, while labeling the objects as offensive or non-offensive. Following this, the closest matching class for each object is found using the function given in Eq. (2). Hence, each object class ended up with a set of object members, where (3) is applied to calculate the probability that the class represents offensive material. Finally, as a step in evaluating the performance and calculating the fitness; the images in the training set are classified as offensive or non-offensive utilizing the probabilities found above. That is, for each image all objects were loaded and their belonging class is determined. When all objects with their matching classes are known, (8) is used to find the total log-likelihood of an image being offensive. At last, β is determined and the total error rate counted. This process results in a number quantifying the classification ability, which in the end is used as the GAs fitness value.

3. Web site classification

The error analysis was based on the hypothesis that a site contains images of only one type. Either all the images are offensive or they are all non-offensive. This seems to be a fair assumption for virtually all offensive sites.

N	p=0.05	p=0.1	p=0.2	p=0.3	p=0.4
2	0.0025	0.0100	0.0400	0.0900	0.1600
3	1.11E-3	6.11E-3	0.0326	0.0840	0.1597
5	2.04E-4	2.17E-3	0.0215	0.0768	0.1784
10	2.75E-6	1.46E-4	6.34E-3	0.0473	0.1662
20	5.37E-10	7.09E-7	5.63E-4	0.0171	0.1275
50	5.54E-21	1.07E-13	4.92E-7	9.33E-4	0.0573
100	3.74E-39	6.32E-25	5.17E-12	9.03E-6	0.0168

Fig. 6. The probability that a site is wrongly classified as a function of N and p.

The developed image retrieval system is used to classify web sites after crawling all images on a given site. Next, all N images on the site are run through the ICE and classified into offensive or non-offensive. If we let Ω denote the number of offensive images on a site, then if $\Omega > \frac{N}{2}$ we assume that the site is an offensive site. This process repeats for all sites to be investigated.

Given the probability that an image is wrongly classified (p) and the total number of images on the site (N), one can use the binomial distribution (9) to find the probability that the whole site is falsely classified.

$$P(X = x) = b(x; n, p) = \binom{n}{x} p^x (1 - p)^{n-x} \tag{9}$$

Since we assume that a site only can contain one type of images it follows that a site with N images is misclassified only if $X > \frac{N}{2}$ single images are falsely classified. Thus, we can write the probability that the site is wrongly classified as

$$P\left(X > \frac{N}{2}\right) = \sum_{x=\frac{N}{2}+1}^{N} b(x; N, p) = \sum_{x=\frac{N}{2}+1}^{N} \binom{N}{x} p^x (1 - p)^{N-x}. \tag{10}$$

Eq. (10) can be used to build a table that expresses the probability of wrongly classifying a site, given the number of images on the site and the probability of wrongly classifying and image. This is depicted in Fig. 6 for the values $N = 2, 3, 5, 10, 20, 50, 100$ and $p = 0.05, 0.1, 0.2, 0.3, 0.4$.

As shown in the table, the probability of a site being falsely classified is very small, even with a quite high probability of wrongly classifying a single image. For example, a site with 20 images and a 30% probability of falsely classifying an image has a mere 1.7% probability of being incorrectly classified.

4. Experimental results

4.1. The image processing

When doing the experiments on the image retrieval system, the operation was divided into two parts; the training and the evaluation. Both parts indicating something about the performance of the system. The evaluation results were counted as

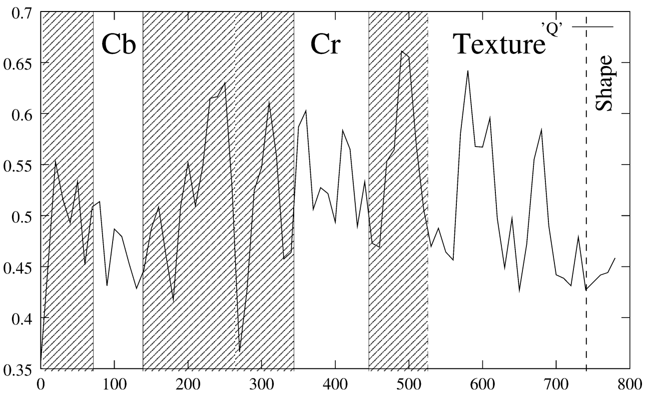


Fig. 7. The Q-vector describing the importance of the elements in the feature vector. The relative importance of the feature elements are texture 58%, Cr 25%, Cb 11% and shape 6%.

the official performance of the system. These results were usually just a few percent lower than the results from the training set. This indicates that the training to a large extent was able to generalize the findings.

The images used in the training were all gathered using AllTheWeb's[1] image-index. As many as 575 non-offensive images were randomly picked from the index, and 365 offensive images were selected by searching on English female names. As a restriction, no images with less than 200 pixels in each dimension were accepted.

The training process was run with a GA population of 3000 over 50 generations. The resulting Q-vector is shown in Fig. 7, and the separation level was found to be $\beta = 0.22$. The shaded areas represent the colors blocked by the initial filter, thus no information is stored in these areas. Like all values in the Q-vector these values are also set randomly during the GA initialization, and they are not excluded from any part of the GA process. For this reason they appear as in the figure and not as zeros. In an optimized version of the algorithm these areas would be removed to reduce computation. However, if the initial filtering is changed to a general segmentation, perhaps to identify other kinds of images, the shaded areas would become an active part of the feature vector.

It can be argued that the color information could be removed from the feature vector, since the initial filtering blocks out everything but the skin colors. Unfortunately, the initial filter accepts a greater range of colors, than that limited to skin. This is done to ensure that virtually all skin will be accepted by the filter. Thus, the later automatically trained process can decide whether certain colors represent offensiveness or not.

When examining Fig. 7 closely, we see that the Cr values are given a higher weight than Cb values. This is because Cr measures the red-green color ratio, whereas Cb

[1] http://www.alltheweb.com.

Type of images	Number of images	Images wrongly classified
Portraits	268	26.5%
Offensive	87	4.6%
Non-Offensive	25	12%

Fig. 8. The ability to classify different types of images.

measures the relationship between green and blue. Since skin color mostly is composed of red, it seems natural that Cr is more important.

Furthermore, the weights in the texture section shows that the lower gradients are given most importance. This means that smoother transitions are preferred, which corresponds to the nature of skin. The *Q*-vector also show a peak in the higher part of the texture section. This is perhaps because values in this range are used to determine that the object is not offensive since such large gradients do not occur in skin. Finally, the shape feature seems to have been given little importance.

The training resulted in 92.1% correct classifications of the images, while the evaluation on an independent data set suggested that 89.4% of the images were classified correctly. The evaluation data set was found in the same manner as the training set, and consisted of 500 offensive and 800 non-offensive images. However, by running the image classification on a dataset solely composed of portrait images, only 73.5% of the images were classified as non-offensive, in other words 26.5% of the 268 images in the dataset was falsely classified. This error is mainly due to the fact that the nature of portrait images is very similar to that of nude photos, with large regions of skin. As can be seen from Fig. 8, other kinds of images experience different kinds of error rates. The errors can probably be reduced at the cost of speed by replacing the initial skin filter, with a more complex image segmentation scheme.

The speed of the processing proved to be very satisfactory. On a single CPU system, an average of 11 images per second were processed. That equals 660 images per minute or 39,600 per hour. In comparison Forsyth and Fleck [8], operated with a processing rate of 10 images per hour but since their paper was published in 1996 it should be noted that the increased computational speed of modern computers also contributed to the increased processing rate experienced with our method. Needless to say, with faster processors or on a multiple CPU system processing speeds can be improved even further.

4.2. The site classification

A number of different sites were chosen at random and each site was crawled, downloading all images. As much as 20 of these sites were chosen for the experiment in Fig. 10. From the fact that none of the sites contained error-rates close to or above 50% we can conclude that all sites were classified correctly. The average image classification error rate for offensive sites were 14.1%, as compared to 9.8% for non-offensive.

Since a site containing N images, where Ω is the number of images determined by the image processing algorithm to be offensive, then as long as $\Omega \neq N - \Omega$, a choice can be made. If we assume that a site consists solely of one class of images, as we did in our work, then as long as the error rate of the image processing is less than 50% the site will be classified correctly. Note that sites are not always solely composed of one class of images, thus making it harder or sometimes impossible to do a correct classification. Fortunately, these sites are the exceptions.

The histograms showing the log-likelihood of the images being offensive, for two example sites, are shown in Fig. 9. The threshold β separates the two classes into offensive on the left and not offensive on the right. It appears quite clear that the sites contain different types of images, and that they can be classified into different classes.

As mentioned earlier, some types of images, like portrait photos, often experience high error rates. This error rate may seem appreciable, but it is far from being 50%, whereas a conclusive choice cannot be made. Moreover, as Internet sites usually consist of several images, the error rate becomes quite acceptable. As Fig. 6 shows, with a classification error rate of 30%, a site consisting of 10 images has a mere 4.7%

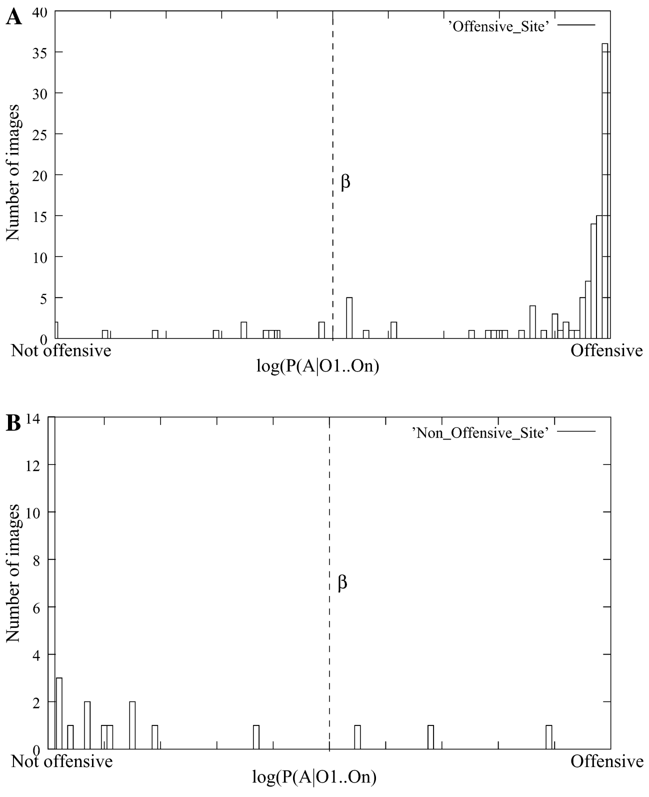


Fig. 9. Histograms of the log-likelihood of the images being offensive compared to the threshold β. (A) shows a histogram composed from an offensive site with 116 images, while (B) shows a histogram from a non-offensive site with 30 images.

Site Number	Images on site	Images classified as offensive	Images classified as non-offensive	Images wrongly classified
Offensive 1	87	83	4	4.6%
Offensive 2	112	87	25	22.3%
Offensive 3	78	62	16	20.5%
Offensive 4	81	68	13	16.0%
Offensive 5	88	76	12	13.6%
Offensive 6	87	68	19	21.8%
Offensive 7	82	77	5	6.1%
Offensive 8	81	70	11	13.5%
Offensive 9	81	72	9	11.1%
Offensive 10	84	74	10	11.9%
Non-Offensive 1	239	29	210	12.1%
Non-Offensive 2	144	26	118	18.0%
Non-Offensive 3	7	1	6	14.3%
Non-Offensive 4	97	2	95	2.0%
Non-Offensive 5	19	1	18	5.3%
Non-Offensive 6	70	2	68	2.9%
Non-Offensive 7	91	18	73	19.8%
Non-Offensive 8	57	6	51	10.5%
Non-Offensive 9	14	0	14	0.0%
Non-Offensive 10	53	7	46	13.2%

Fig. 10. The classified sites.

probability of being wrongly classified as a whole. This is, of course, assuming that offensive and non-offensive material is not mixed on the same site.

5. Further work

Much weight has been put into making the implementation as fast as possible. In this regard, some choices were made that may have had impact on the accuracy of the image retrieval process. Currently, we are working on a Bayesian approach for segmenting skin. The results are promising although the method is quite slow.

Further studies should be done on other methods for segmenting the image, as well as other methods of extracting features. In addition, further investigation on how to improve the training would be interesting. One can imagine using support vector machines in defining the regions that the objects belong to, or experimenting with different initializations of the GA. Perhaps positive values could be be sampled from a normal distribution with mean 0, and used as as initialization for the GA. Also it would be interesting to restrict the data manipulation to only work with the most significant bits, and thus reducing the amount of data to investigated. Furthermore, the unused values of the feature vector should be removed to reduce the size of the solution space. Some effort into adjusting the many parameters may also prove beneficial.

The use of textual information as an addition to the proposed system is an interesting topic. It is likely that one could use text as a manner of trivially rejecting the use of the image based algorithms to improve speed, as well as using it as a guide in the decision-making process. A study making use of existing methods for text analysis and classification would be highly relevant.

It would be interesting to apply this method to other datasets. Other types of images could be used by replacing the initial skin filter with a general segmentation. The training would then attempt to find typical objects represented in the training images. Moreover, the dataset could also be textual documents, where the feature vector is a vector of words. In this case, the system could be trained to search for a special document content, and thus Internet sites containing this content could be identified. Here, GA could be used to find the composition of words that identifies the requested document class.

6. Conclusion

The method proposed performs very well under the condition that the web sites investigated contain several images. All the 20 sites tested were correctly classified, and the image-retrieval part of the system was able to properly identify 89% of the images in the evaluation data set composed of 500 offensive and 800 non-offensive images. However, the system may wrongly classify sites with very few images.

Although the performance could be further improved by supplementing with text analysis, the system processed an average of 11 images per second, and is therefore fast enough to be used as a site classifier in a web search engine.

Acknowledgments

We would like to thank John Reidar Mathiassen, and several anonymous reviewers for their useful comments.

References

[1] P. Alshuth, T. Hermes, C. Klauck, J. Kreyss, M. Roper, Iris—image retrieval for images and videos, 1996.

[2] E. Angelopoulou, Understanding the color of human skin, in: Proc. SPIE Conf. on Human Vision and Electronic Imaging VI (SPIE) 2001, SPIE Vol. 4299, SPIE Press, May 2001, pp. 243–251.

[3] S. Baheerathan, F. Albregtsen, H.E. Danielsen, New texture features based on the complexity curve, Pattern Recogn. (1999) 605–618.

[4] L. Bretzner, I. Laptev, T. Lindeberg, Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering, 2002.

[5] S. Brumby, N. Harvey, S. Perkins, R. Porter, J. Szymanski, J. Theiler, J. Bloch, A genetic algorithm for combining new and existing image processing tools for multispectral imagery, 2000.

[6] S. Brumby, J. Theiler, S. Perkins, N. Harvey, J. Szymanski, J. Bloch, M. Mitchell, Investigation of image feature extraction by a genetic algorithm, 1999.

[7] S. Cohen, L. Guibas, Shape-based image retrieval using geometric hashing, 1997.
[8] D. Forsyth, M. Fleck, Identifying nude pictures, 1996.
[9] S. Geman, D. Geman, Stochastic relaxation, Gibbs distribution and the Bayesian restoration of images, IEEE Tans. Part. Anal. Mach. Intell. 6 (1984) 721–741.
[10] R.-L. Hsu, M. Abdel-Mottaleb, A. Jain, Face detection in color images, IEEE Trans. Patt. Anal. Mach. Intell. 24 (2002) 696–706.
[11] A. Jain, A. Vailaya, Image retrieval using color and shape, 1996.
[12] B. Martinkauppi, M. Soriano, M. Laaksonen, Behavior of skin color under varying illumination seen by different cameras in different color spaces, in: M.A. Hunt (Ed.), Proc. SPIE Machine Vision in Industrial Inspection IX, Vol. 4301, January 19–26, San Jose, California, pp. 102–103.
[13] H. Kruppa, M.A. Bauer, B. Schiele, Skin Patch Detection in Real-World Images, DAGM Symposium 2002, Zurich, Switzerland, to appear.
[14] L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank Citation Ranking: Bringing Order to the Web. Technical Report, Stanford Digital Library Technologies Project, 1998.
[15] J.S. Park, D.H. Chang, 2-d invariant descriptors for shape-based image retrieval, Pattern Recognition Letters (2001).
[16] R. Poli, Genetic programming for image analysis, in: J.R. Koza, D.E. Goldberg, D.B. Fogel, R.L. Riolo (Eds.), Genetic Programming 1996: Proceedings of the First Annual Conference, Stanford University, CA, USA, 28–31 1996, MIT Press, pp. 363–368.
[17] J.M. Rehg, K.P. Murphy, P.W. Fieguth, Vision-based speaker detection using Bayesian networks, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Ft. Collins, Colorado, 1999, pp. 110–116.
[18] M. Safar, C. Shahabi, X. Sun, Image retrieval by shape: a comparative study, in: IEEE Internat. Conf. on Multimedia and Expo (I), 2000, pp. 141–144.
[19] M.C. Shin, K.I. Chang, L.V. Tsap, Does colorspace transformation make any difference on skin detection? IEEE Workshop on Applications of Computer Vision, Orlando, FL, December 2002, pp. 275–279.
[20] M. Störring, H.J. Andersen, E. Granum, Skin colour detection under changing lighting conditions, in: H. Araujo, J. Dias (Eds.), 7th Internat. Symp.on Intelligent Robotic Systems, Coimbra, Portugal, 1999, pp. 187–195.
[21] J. Szymanski, S. Brumby, P. Pope, D. Eads, D. Esch-Mosher, M. Galassi, N. Harvey, H. McCulloch, S. Perkins, R. Porter, J. Theiler, A. Young, J. Bloch, N. David, Feature extraction from multiple data sources using genetic programming, 2002.
[22] W.A. Tackett, Genetic programming for feature discovery and image discrimination, in: S. Forrest (Ed.), Proc. 5th Internat. Conf. on Genetic Algorithms, ICGA-93, University of Illinois at Urbana-Champaign, 17–21 1993, Morgan Kaufmann, pp. 303–309.
[23] R. Veltkamp, M. Hagedoorn, State-of-the-art in shape matching, 1999.
[24] J. Yang, W. Lu, A. Waibel, Skin-color modeling and adaptation, in: ACCV, vol. 2, 1998, pp. 687–694.

Multidimensional Visualization and Navigation in Search Results

Will Archer Arentz[1] and Aleksander Øhrn[2]

[1] Norwegian University of Science and Technology,
NO-7491 Trondheim, Norway
Will.Archer.Arentz@idi.ntnu.no
[2] Fast Search & Transfer ASA,
P.O. Box 1677 Vika,
NO-0120 Oslo, Norway

Abstract. Search engines traditionally index unstructured text and return ranked lists of documents that match a given query. As search engines functionally move in the direction of traditional databases and offer scalable and efficient handling also of structured textual and numerical data, this way of presenting results needs to be revisited and augmented with alternative presentation modes. This paper presents a multidimensional visualization scheme with navigational capabilities, built on top of a data aggregation framework implemented in a state-of-the-art search engine. The proposed visualization scheme is designed to provide analytic understanding of the result set also for ad hoc queries, and correlates user-specified dimensions or attributes of the documents. By employing advanced linguistic and entity extraction techniques, the scheme can also be applied in situations where the original documents consist of unstructured text.

1 Introduction

As information spaces have become more complicated, substantial attention has been devoted to considering how spatial dimensions can be used to decrease the task demands. Although there are numerous prototypes for visualization large document spaces [1][2][6][7][15][21], there is little in the way of systematic comparison of the value of these approaches.

Sebrechts et al. [17] did a study where they compared viewing search results as text, 2D- and 3D-based visualization. Users were monitored and interviewed, and only under the right combination of tasks, user, and interface did the 3D visualization result in performance comparable to the 2D and textual tools.

There have also been a few studies of 3D-based visualization tools. The hyperbolic browser [10] simulated changes in appearance of documents spread over a 3D spherical surface. That tool used a focus+context fish-eye approach to visualize and manipulate large hierarchies. A laboratory experiment contrasted using the hyperbolic browser against a conventional 2D scrolling browser with a horizontal tree layout. Although users preferred the hyperbolic visualization, there were no performance advantage on the tasks of finding specific node locations.

More recently, Swan and Allan [20] performed a controlled study comparing a text-based system [14], a GUI oriented system, and the latter enhanced with 3D visualization of document clusters. They wanted to prove so-called "aspect-oriented" IR, which emphasizes finding some specified information, not documents, per se. In that context there was a small advantage for the 3D system over text-based, and for the text-based over the plain GUI. However, there was no evidence for the overall effectiveness of the use of 3D; the system's utility depended on the tasks and the users. Experienced users preferred the text-based system, while novices liked GUI systems. Some users thought the 3D approach was "worthless", others thought it natural and intuitive.

An empirical study of 3D visualization for information retrieval tasks performed by Newby [13] resulted with the same as many others, namely that 3D visualization does not usually improve performance. In fact, the use of 2D visualization is generally preferred when the users have mixed background. Sutcliffe et al. [19] suggest that users of a 3D visualization system for information retrieval require training to be able to use it efficiently.

Several other attempts have been made to utilize visualization in the information retrieval process [12] [22] [23] [8] [18] [11]. The general conclusion being that the simple 2D GUI is the most easy to use for novice users, while the more advanced 3D required training. Other studies [9] [5] have concluded that the use of feedback to the search engine (*relevancy feedback*) is a very effective and powerful tool to improve the quality of the search results. With this in mind the presented application was developed, using information rich 2D visualization and a simple GUI to enable the user to navigate rapidly through the search results.

The remainder of this article is organized as follows: Section 2 describes the contribution to traditional search engine technology, with the analytic functionality needed by the user interface; Section 3 explains the visualization and navigation interface, while section 4 goes through a user trial. Finally, section 5 summarizes with some concluding remarks.

2 LiveAnalytics

FAST Data Search[3] (FDS) is an enterprise search platform with technology proven to scale both with respect to data volume and query rate. LiveAnalytics is an umbrella name for a collection of FDS features that analyze and/or organize the set of documents that match a given query, to yield information about the result set that complements the traditional ranked list of documents. One of these LiveAnalytics features is query-driven aggregation of structured data of textual, numerical or temporal type.

Let a document d be composed of a collection of typed fields $F = F_s \cup F_u$, where F_s and F_u are disjoint and denote the sets of fields with structured and unstructured data, respectively. For example, given their natural interpretations, the field *body* $\in F_u$ would be an unstructured textual field, the field *size* $\in$

[3] http://www.fastsearch.com

F_s would be a structured numerical field, the field $author \in F_s$ would be a structured textual field, and the field $timestamp \in F_s$ would be a structured temporal field. Observe that a field in F_s might very well be algorithmically derived from one or more fields in F_u, e.g., by techniques that given large chunks of free text identify names of persons, corporations or products or other entities of interest. Also note that a field in F_s may take on zero or more values, i.e., structured fields are allowed to be sparse and multi-valued.

The basic data aggregation process is as follows:

1. Given any query q, compute the set of matching documents $R(q) = \{d \mid d \text{ matches } q\}$.
2. Let $N \subseteq F_s$ denote a specified set of fields. For each $f \in N$ do the following:
   (a) Compute the histogram of values found in field f in documents $R(q)$. If f is of numerical or temporal type, this involves applying a real-time discretization algorithm.
   (b) If f is of numerical or temporal type, compute selected scalar statistics for values found in field f in documents $R(q)$.
3. Sort the $|N|$ aggregation results.

As search engines are designed from the ground up for responsiveness and query volume, this all happens with very little latency. And since the implementation is done within a distributed architecture where the work is parallelized out over multiple machines, the latency stays low even for massive data volumes or when $R(q)$ is very high. Search engines can be configured to make use of various linguistic techniques, so for natural language queries or traditional keyword-based queries it is worth noting that a document d does not necessarily have to match q verbatim for it to be a member of $R(q)$.

As noted, fields of numerical or temporal type undergo special processing. For example, a histogram of raw values would be needlessly detailed for most uses. Instead, we would like the histogram buckets to cover intervals of various width. The process of computing these bucket widths according to certain criteria is referred to as discretization, and several discretization algorithms have been implemented. Discretization algorithms that produce variable-width buckets require special attention to implementation in a distributed setting, since the discretization and counting is done independently on each search/aggregation node and we have to be able to merge them to yield a global result.

It might be the case where $|N|$ is large, so large that a user might not have screen real-estate enough to display all of them, or need assistance in identifying the most "useful" aggregation results. To this end, the $|N|$ aggregation results are sorted before they are returned to the client. They are sorted according to a scoring function that combines a static and a dynamic measure of usefulness. The static measure reflects a field's a priori weight or importance, while the dynamic measure captures the information content or entropy of the field's histogram over $R(q)$.

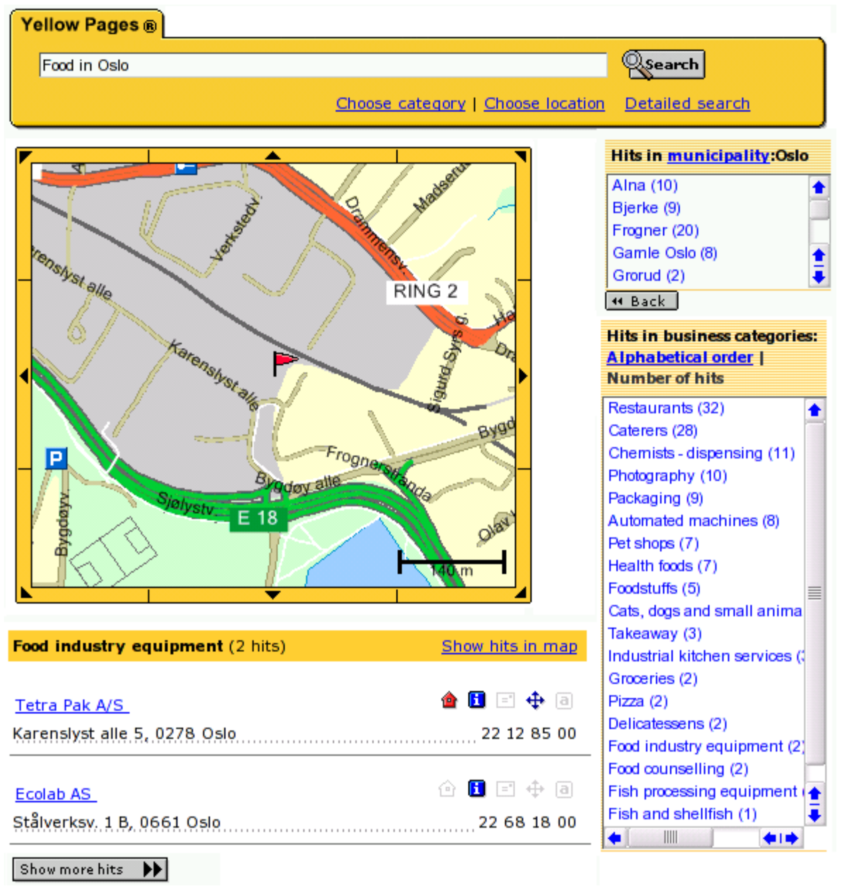
The data aggregation facilities of LiveAnalytics have a wide variety of potential uses, e.g.:

- The computed histograms enable faceted browsing [4] over the fields $f \in N$. Faceted browsing is sometimes also referred to as guided navigation or dynamic drilldown, and is often used in e-commerce applications.
- The scalar statistics computed for numerical or temporal fields describe the set $R(q)$ in a way that for some applications is very useful, e.g., when searching over financial data. The availability of these statistics also enable interesting queries to be made in a second pass, e.g., to return all documents which have an above-average value for a field $f \in N$.
- A document is just the basic retrieval unit. If we redefine the notion of a document to be a list of items, e.g., a shopping basket, we can effectively use the computed histograms to do collaborative filtering [3].
- The current data aggregation process is one-dimensional, i.e., the aggregation itself does not correlate fields. This may change in the future, but since a search engine's query latency is so low, we can often afford to do certain types of multi-dimensional aggregation by issuing multiple one-dimensional queries. This is explored in the following section.

3 Visualization

The information available through LiveAnalytics can be visualized in ways that give the users a good overview of the search results. The entity extraction applied in FDS, enables the use of navigational feedbacks in terms of specifying one or more entities before resubmitting the query to the search engine. Consider an index containing medical research publications, where entities like title, author, publication year, chemical substance, mesh-terms, and publication journal have been identified. A general search term like "myocardial infarction" will return a very large result set, leaving the user with a need to further specify the search to find what she is looking for. By specifying the range of publication years, the result set can i.e. be limited to only contain recent publications. Naturally, further specification of for example journal or author names will further narrow down the result set. In addition, entities can be organized into taxonomies or related to ontologies. This is done by either at query time or at index time associating a specific entity with either broader categories in a taxonomy or related concepts in an ontology. A record describing a restaurant in a Yellow Pages index (see figure 1) would for example be tagged with both the street name, municipality, city and state. The result presentation will then adaptively be able to select an appropriate abstraction level based on the category/entity distribution in the result set. If multiple states are present in the result set, navigation will be dynamically computed to visualize the state distribution. If on the other hand all results happen to be from the same city, navigation will dynamically illustrated the distribution between areas within that city.

The visualization step is benefiting from the ability to do both static and dynamic clustering with supervised and unsupervised methodologies. Hence, any of the attributes that are analyzed can be abstracted to an optimal level. The selected abstraction level is used as a basis for either 1-dimensional or multi-dimensional distribution analysis and presented in heat maps. The user can then select any of the dimensions and narrow down the targeted range that will be fed back into the analysis and visualization engine.



Food in Oslo

Hits in municipality:Oslo

Alna (10)
Bjerke (9)
Frogner (20)
Gamle Oslo (8)
Grorud (2)

Hits in business categories:
Alphabetical order |
Number of hits

Restaurants (32)
Caterers (28)
Chemists - dispensing (11)
Photography (10)
Packaging (9)
Automated machines (8)
Pet shops (7)
Health foods (7)
Foodstuffs (5)
Cats, dogs and small anima
Takeaway (3)
Industrial kitchen services (
Groceries (2)
Pizza (2)
Delicatessens (2)
Food industry equipment (2
Food counselling (2)
Fish processing equipment (
Fish and shellfish (1)

Food industry equipment (2 hits)

Tetra Pak A/S
Karenslyst alle 5, 0278 Oslo 22 12 85 00

Ecolab AS
Stålverksv. 1 B, 0661 Oslo 22 68 18 00

Fig. 1. Searching for “Food in Oslo” in a Yellow Pages index using LiveAnalytics.

A screen dump showing the LiveAnalytics functionality applied on a Yellow Pages index can be found in Fig. 1. Note the two boxes containing a class of *places* and a class of *business categories*. Belonging to each of these classes are several elements, which we call *modifiers*, while the classes are named *navigators*.

Examining the boxes in Fig. 1, each element has a number associated with it. This number says how many documents were found with this modifier applied. In the Yellow Pages example the entered query was “Food in Oslo”, and the “Restaurants (32)” element in the business category suggests that 32 restaurants were found.

A user may choose to select a set of modifiers. This can i.e. be useful while searching for publications from a set of authors in a scientific publication index. Furthermore, both numerical and textual navigators may both be used. If the dataset was a commercial product offering with navigators like name, description, time of delivery and price, the user would be able to limit her search to any price range she want. This feature also makes this system very attractive as an analytic tool in the finance and marked-sectors. By combining data sources and including a time line, a multidimensional visualization based on this framework could provide very useful to financial analysts.

3.1 Visualizing search results using heatmaps

Modern search engines can deliver multiple hundreds of queries per second (QPS) per CPU. This enables us to dig deep using multiple searches while gathering data for visualization. In the example in the previous section, one would maybe want to investigate the number of yearly publications by a specific author about a given topic. Straight forward, this information would be available by entering the search term before selecting the author as a modifier and resubmitting the query. Thereafter, by selecting the navigator "publication year" the number of publications for each year will be displayed in the modifier table.

If one would want to do this for all authors, or at least the n authors with most publications, multiple queries could be resubmitted after the initial query. Each new query specifying one of the authors as modifier. This would result in a table containing authors on one axis and year of publication on the other. The content of each cell $C_{author,year}$ would in this case be the number of publications an author had that year. A heatmap can be created from this table simply by color-coding the cells of the table. In the proposed application dark cells represented low values down to zero, while light cells represented high values. Heatmaps are commonly used in financial applications, but proved equally informant for visualizing search results. Fig. 2 shows an example on how heatmaps are used in our application. The initial search shows that "J. Seward" has a very large number of publications over many years. By selecting him and his co-authors as modifiers, we can see what research topics he has been involved in. Furthermore, by selecting publication journal against chemical substance, it can be seen where the author published work where a named chemical substance was involved.

As the framework for doing navigational search is completely general, a multitude of data sets may be used. Whether it is Yellow Pages, scientific publications or stock marked related data, it can all be visualized for greater understanding.

Some governments are introducing laws (i.e. the American *Sarbanes-Oxley Act of 2002* [16]) that require corporations to be able to store and retrieve all e-mail being sent in and out of the company over the past two years. This can be used to detect unauthorized communication. By utilizing heatmaps to visualize the content as previously described, one can see who are talking to each other, as well as what topic each person is talking about. By applying automatic entity extraction, elements like date, to- and from-addresses, topics, company names, personal names and more, can be extracted.

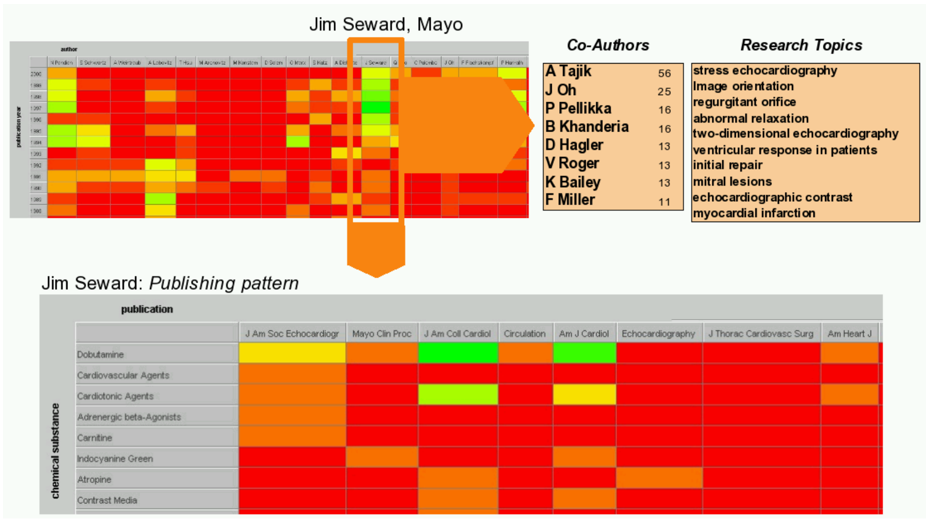


Fig. 2. Examining the publishing pattern of “Jim Seward”.

A useful content mapping of such e-mails can be done by specifying a topic in the search query, before mapping the from- and to-addresses to the heatmap axis. In addition, the color-coding of the cells can be based on the time where communication between the two individuals about this topic first appeared. This means that communication that happened several months ago will be expressed by light cells, while more recent communication is manifested in dark cells. From this heatmap it can be seen when a topic first emerged, and who was initially involved.

Another interesting view is the relationship between topics and people. By mapping each person related to an e-mail on the x-axis and the all topics on the y-axis, it is easy to investigate which topic a person has talked about, as well as how much each person has been involved in this topic. The color-coding for each cell, could be based on the frequency that a person sends or receives e-mails about a given topic, or it could be a separate color for sent, received, sent and received, and cc-ed. This visualization is somehow similar to that of Fig. 2.

4 User trial

To see how the user experienced our application, we let 10 users try out a version with medical publications as data set. The users were given no training, and were left alone to understand the interface. The subjects had different level of experience, and this level was determined through individual interviews.

All users were given two search interfaces to test. First, they were asked to use the traditional search engine interface with a string as input and a list of results as output. Second, they were asked to perform similar searches in the new interface with visualization and navigation capabilities. Finally, each subject was given 5 questions to compare the two interfaces. The questions with results can be found in Table 1.

Table 1. The five questions asked the test subjects with answer alternatives. The percentage numbers reveal the answer distribution.

Q1: Is the new user interface easy to use?				
Yes - *70%*			No - *30%*	
Q2: How is the information value in the new interface compared to the traditional?				
Trad. far better *10%*	Trad. a bit better *10%*	No difference *0%*	A bit better *40%*	Far better *40%*
Q3:Would you prefer to do most of your searching with the new interface?				
Yes - *60%*			No - *40%*	
Q4:How easy is it to find what you are looking for?				
Trad. much easier *0%*	Trad. a bit easier *0%*	No difference *10%*	A bit easier *40%*	Much easier *50%*
Q5:Do you find heatmaps as a good way of visualizing search results?				
Useless *0%*	Didn't grasp *20%*	Other vis. may be better *10%*		Informative *70%*

The group was test subjects were composed of 5 novice users and 5 experienced users. A few of the experienced users expressed a wish to see a user manual describing the system. However, no such manual was written, thus leaving the users to figure things out by themselves. The three users who found the interface hard to use were all experienced, holding at least a masters degree in computer science. However, none of the novice subjects expressed that the system was hard to use. The reason for this may be that novice users are accustomed to not understanding everything when using a computer, and that not understanding all the features still did not make it hard to use as long as they were able to perform searches. Another possibility is that novice users are more open to different interfaces, while experienced users are more biased towards traditional ways.

Most of the users agreed that the presented interface contained more valuable information than traditional search interfaces. Furthermore, 60% of the users would prefer to use the new interface on a daily basis. This result was very unexpected, since no training or user manual had been provided. It was expected that users would need more time using the system before showing interests to change habits. Moreover, all the test subjects felt it was easier, or equally easy, to find what they were looking for with the navigational interface compared to the traditional. As much as 90% said it was easier.

When asking the test subjects if they found heatmaps to be a useful visualization method for search results, 70% answered that heatmaps were very useful and informative. Although two users did not understand the point of using heatmaps, only one users thought that other visualization methods might be better.

After the subjects had finished answering the questions in Table 1, an attempt was made to explain a bit more about how one can use the interface. This was done by giving some examples where heatmaps were shown to be very informative and helpful to understand the result set. This seemed to increase the users motivation and appreciation for the interface.

5 Conclusion

A new method for visualizing and navigating in search results was presented. A group of users were asked to try out the system and evaluate it against traditional search interfaces. The results were promising, and several ideas for improvement were discovered. Like most studies of visualization tools also have concluded, some documentation or training would be helpful to fully appreciate the new interface.

Further work will be to redesign the interface according to the ideas discovered in collaboration with the test subjects. In addition an easy-to-understand user manual will be written, to make it easy for any users to get started. Furthermore, we wish to make the user trial with a larger user base, preferably comparing three interfaces. The third interface would preferably be a well known search result visualization.

References

1. Allen, R.B., Obry, P., Littman, M.: An Interface for Navigating Clustered Documents Sets Returned by Queries. Proceedings of SIGOIS, Milpitas CA (1993) 203–208
2. Benford, S., Snowdon, D., Greenhalgh, C., Ingram, R. Knox, I., Brown, C.: VR-VIBE: A Virtual Environment for Co-operative Information Retrieval. Eurographics '95, Maastricht, The Netherlands (1995) 349–360
3. http://jamesthornton.com/cf/
4. http://bailando.sims.berkeley.edu/flamenco.html
5. Efthimiadis, E.: Interactive Query expansion and Relevance Feedback for document Retrieval Systems. PhD thesis, City University, London, UK, (1992)
6. Fowler, R., Wilson, B., Fowler, W.: Information Navigator: An information system using associative networks for display and retrieval, Report NAG9-551, No. 92-1, Univ. of Texas - Pan American, Edinburg TX. See http://bahia.cs.panam.edu/info_vis/inf_nav/info_nav_tr_92.html
7. Hearst, M., Karadi, C.: Cat-a-Cone: An Interactive Interface for Specifying Searches and viewing Retrieval Results using a Large Category Hierarchy. Proceedings 20th Annual International ACM/SIGIR Conference, Phil PA (1997). See: http://www2.parc.com/istl/projects/ia/papers/cac-sigir97/sigir97.html
8. Hearst, M.A, Pedersen, J.O.: Visualizing Information Retrieval Results: A Demonstration of the TitleBar Interface. CHI'96 Proceedings (1996)
9. Koenemann, J., Belkin, N.J.: A Case For Interaction: A Study of Interactive Information Retrieval Behaviour and Effectiveness. CHI'96 Proceedings (1996)

10. Lamping, J., Rao, R., Pirolli, P.: A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. CHI'95 Proceedings: Conference on Human Factors in Computer Systems: Mosaic of Creativity, Denver CO (1995)
11. Lee, C., Chen, Y.-T.: Distributed visual reasoning for intelligent information retrieval on the Web. Interacting with Computers **12** (2000) 445–467
12. Li, W.-S., Shim, J. Candan, J.S.: WebDB: A system for querying semi-structured data on the web. Journal of Visual Languages and Computing **13** (2002) 3–33
13. Newby, G.B.: Empirical Study of 3D Visualization for Information Retrieval Tasks. Journal of Intelligent Information Systems **18** (2002) 31–53
14. NIST PRISE Search Engine: http://www.itl.nist.gov/iaui/894.02/works/papers/zp2/zp2.html
15. Rennison, E.: Galaxy of News: An approach to Visualizing and Understanding Expansive News Landscapes. Proceedings of UITS'94 (1994)
16. A summary of the Sarbanes-Oxely Act of 2002: http://www.aicpa.org/info/sarbanes_oxley_summary.htm
17. Sebrechts, M.M., Vasilakis, J., Miller, M.S., Cugini, J.V., Laskowski, S.J.: Visualization of Search Results: A Comparative Evaluation of Text, 2D and 3D Interfaces. Research and Development in Information Retrieval (1999) 3–10
18. Stenmark, D.: To Search is Great, to Find is Greater: a Study of Visualisation Tools for the Web. See: http://webb.informatik.gu.se/ dixi/publ/mdi.htm
19. Sutcliffe, A.G., Ennis, M., Hu, J.: Evaluating the effectiveness of visual user interfaces for information retrieval. International Journal of Human Computer Studies **53** (2000) 741–763
20. Swan, R.C., Allan, J.: Aspect Windows, 3D Visualizations, and Indirect Comparison of Information Retrieval Systems. Proceedings of the 21st Annual International ACM/SIGIR Conference, Melbourne Australia (1998)
21. Wise, J.: The Ecological Approach to Text Visualization. JASIS (1998)
22. Zhou, X., Yates, J.D., Chen, G.: Using visual spatial search interface for WWW applications. Information Systems **26** (2001) 61–74
23. Zlobin, D., Roudometkine, A.: Visual representation of document-oriented information on the Web. Proceedings of the APL Berlin 2000 Conference (2000) 247–254