Clustering Based Ensemble Classification for Spam Filtering

Robert Neumayer
neumayer@ifs.tuwien.ac.at

Vienna University of Technology
Department of Software Technology and Interactive Systems
Favoritenstrasze 9-11/188, 1040 Vienna, Austria

Abstract. Spam filtering has become a very important issue throughout the last years as unsolicited bulk e-mail imposes large problems in terms of both the amount of time spent on and the resources needed to automatically filter those messages. Text information retrieval offers the tools and algorithms to handle text documents in their abstract vector form. Thereon, machine learning algorithms can be applied. This work deals with the possible improvements gained from ensembles, i.e. multiple, differing classifiers for the same task. Those individual classifiers can fit parts of the training data better and therefore may improve classification results, when the best fitting classifier can be found. Basic classification algorithms as well as clustering are introduced. Furthermore the application of the ensemble idea is explained and experimental results are presented.

1 Introduction

Unsolicited commercial e-mail has become a serious problem for both private and commercial e-mail users. By now, spam is estimated to be a very high percentage of all e-mail traffic (in some estimates up to 80 % or even more). The resulting waste of resources (hardware, time, etc.) is enormous.

The currently employed infrastructure for e-mail transfer, the simple mail transfer protocol (SMTP), hardly provides any support for detecting or preventing spam. We are also lacking a widely accepted and deployed authentication mechanism for the sending of e-mails. Thus, until a new global e-mail infrastructure has been developed which allows for better controlling of this problem, two major approaches show the greatest potential for coping with the problem: *detecting* spam based on *content filtering* or *preventing* spam based on increasing the costs associated with sending out e-mail messages.

This work concentrates on the application of machine learning techniques to the spam problem and therefore clearly belongs to the content filtering category. The main task of machine learning is to "learn" labels of instances (like legitimate or non-legitimate for e-mail messages) from a given training set, and subsequently tag new or unknown instances (e.g. incoming e-mails).

More specifically, the feasibility of decomposing spam training sets into smaller, more specialised subsets for spam filtering is investigated. The main idea behind this is that the similarity of incoming messages to all those subsets can be computed, one of which the message is likely to fit in best. That subset can then be used to label that specific instance. Therefore, the ensemble classification approach was chosen. Usually, one classifier is built on one training set, and this classifier is then used to categorise incoming messages. Ensembles comprise several classifiers for the same task, varying either in parameter settings or data used for training. This work relies on the latter type of ensemble.

The remainder of this paper is structured as follows. Chapter 2 gives an overview of previous research, Chapter 3 introduces the main principles of text categorisation and the application of machine learning techniques. Furthermore, Chapter 3 deals with the problem of feature selection in the context of text information retrieval. The application of ensemble techniques to the spam problem is explained in Chapter 5. Chapter 6 gives an overview about clustering and experimental results. Finally, Chapter 7 gives a short outlook and summarises the work done.

2 Related Work

Numerous spam filtering strategies with varying degrees of efficiency have been proposed and developed. Many of them implement static or rule based approaches, which leave opportunities for spammers to react and circumvent existing anti-spam solutions. As a result, many of the existing approaches yield good classification results only over short periods of time (until spammers have found a workaround). This aspect manifests itself in an ongoing "arms race" between spammers and developers of spam filtering software. The most important

and the most widespread example for such an approach is Apache's SpamAssassin [1], currently a de-facto standard for e-mail classification and spam filtering. Here, a large number (currently over 700) of many different types of tests and checks are combined. In this case the decision whether a message is spam or not is based on a simple thresholding mechanism. The weightings for the different tests are computed by a linear perceptron based upon large spam and ham corpora. Beyond that, many other products and tools have been developed for spam filtering (both commercial and open source). Although they may have different names, they often rely on similar methods, and some are even directly based on SpamAssassin.

More generally, by omitting the special nature of e-mail traffic, spam filtering can be regarded as a classic *text categorisation* task. Thorough research has been conducted in this field.

Machine learning techniques are widely used in automated text classification, including spam filtering (see [2] for a survey of techniques used). Due to their simplicity and performance advantages over many other sophisticated approaches *naïve Bayes classifiers* [3] are maybe the most prevalent. They have been used for spam filtering with great success, since online training is particularly well supported. Many implementations are available, most of them based on Paul Graham's ideas [4, 5]. Case-based techniques, concentrating on editing the case base and counteracting concept drift in spam mails, are described in [6]. [7] shows that n-gram frequencies do not necessarily boost classification performance in the spam context as well as the importance of meta data for this task.

In terms of classifier techniques, the feasibility of applying support vector machines (SVM) for text classification has been investigated in [8]. The use of latent semantic indexing (LSI) for spam filtering is assessed in [9]. A comprehensive comparison of different text categorisation methods is also described in [10]. A k-NN classifier is compared to different LSI variants and support vector machines. SVMs and a k-NN supported LSI performed best, all of which having both advantages and disadvantages.

Many of the topics described in this paper are explained in more detail in [11].

2.1 Ensemble Classification

Ensembles are groups of machine learning instances that work together and are used to improve the classification results of the overall system. The most popular ones are boosting and bagging [12]. Those algorithms train classifier instances on different subsets of the overall data set. Bagging combines the results of classifiers trained on sub samplings of the data set. A good overview of different ways of constructing ensembles as well as an explanation about why ensembles are able to outperform its single members is pointed out in [13]. The importance of diversity for the successful combination of classifiers is given in [14].

An impressive application of ensemble techniques for decision trees are Random Forests [15], where several decision trees are constructed for the same problem and their results are aggregated to find the best overall classification decision.

In this context, ensembles are used in the sense that several classifiers are trained on the same problem (spam detection), but on different subsets of spam and ham messages. Assume that there are two distinct groups in spam data: medical and investment topics. Furthermore there are two main groups of private messages like private and business mails. A newly incoming e-mail is likely to best fit into one category only, depending on its content. For that scenario, four classifiers would be trained, one for each combination of ham and spam:

1. Medical / private
2. Medical / business
3. Investment / private
4. Investment / business

Every new message is now classified by all of those four classifiers, their possible output labellings and levels of confidence (for one incoming message) could be:

1. Spam / 1.00
2. Spam / 0.66
3. Spam / 0.33
4. Ham / 0.66

This message would clearly be tagged as spam as most classifiers are confident that it is. This example, of course, needs the data to be categorised into subgroups a priori, which usually is not the case. Manual sorting of a mail repository into those categories is not very feasible as well. The clustering part (identifying existing subgroups of the data) will be covered in more detail in Chapter 6.

The basic objective of this work is to provide e-mail classification based on the concept of the vector space model which has been introduced in [16] and is widely used, particularly in the area of information retrieval [17, 18]. The next chapters will cover the application of that model to text data and introduce the techniques associated with it as well as the e-mail categorisation task in general. After the process of sub sampling and ensemble classification will be explained in detail.

3 Text Categorisation and Text Information Retrieval

In the information retrieval context, an e-mail is hence treated as a simple text document and is represented by a set of (characteristic) words or *tokens*. Models for text representation range from lists of whole words to vectors of *n-grams* (i. e., tokens of size n). Tokenisation may include stemming, i. e., stripping off affixes of words leaving only word stems. It is very common to use lists of stop words, i. e., static, predefined lists of words that are removed from the documents before further processing (see [19] or *ranks.nl*[1] for a sample list of English stop

[1] `http://www.ranks.nl/tools/stopwords.html`

words). Features in this category are purely content-based, computed only from the message text itself (without aggregated "meta-information" of any kind) and will be denoted as "low-level" features in this paper.

In classic text categorisation low-level features are computed from a labelled training set of sufficient size. New messages can be assigned to the class represented by the most "similar" messages in terms of word co-occurrences.

An introduction to Information Retrieval as such is given in [20]. The basic idea is to treat text as a bag of words or tokens. IR abstracts from any kind of linguistic information and is often referred to as statistical natural language processing (NLP). Documents are represented as term vectors. A document collection containing the following two documents:

```
This is a text document.
```

and

```
And so is this document a text document.
```

would represent its documents by a vector of length 7, details are shown in Table 1.

Table 1. Cluster validation experimental results for a subset of 500 ham messages of the TREC corpus.

Document/Token	**this**	**is**	**a**	**text**	**document**	**and**	**so**
1	1	1	1	1	1		
2	1	1	1	1	2	1	1
Document frequency	2	2	2	2	3	1	

This representation is subsequently used to calculate distances between or similarities of documents in the vector space.

Once a text is represented by tokens, more sophisticated techniques can be applied. In the context of a vector space model a document is denoted by d, a term (token) by t, and the number of documents in a corpus by N.

The number of times term t appears in document d is denoted as the *term frequency* $tf(d)$, the number of documents in the collection that term t occurs in is denoted as *document frequency* $df(t)$, as shown in Table 1. The process of assigning weights to terms according to their importance or significance for the classification is called "term-weighting". The basic assumptions are that terms that occur very often in a document are more important for classification, whereas terms that occur in a high fraction of all documents are less important. The weighting we rely on is the most common model of *term frequency inverse document frequency* [21], where the weight $tfidf$ of a term in a document is computed as:

$$tfidf(t,d) = tf(d) * ln(N/df(t)) \tag{1}$$

This results in vectors of weight values for each document d in the collection. Based on such vector representations of documents, classification methods can be applied.

3.1 Application Scenario

Spam filtering can be regarded as a binary classification problem. There are two categories—called *spam* and *ham*, and each incoming message has to be assigned to one of them.

More generally, the two categories are also called *positives* and *negatives*. In spam filtering the term *positive* usually denotes spam, since this terminology may be ambivalent, the terms of accuracy rates by class are emphasised on, i.e. sensitivity and specificity.

4 Feature Selection and Dimensionality Reduction

When tokenizing text documents one often faces very high dimensional data. Tens of thousands of dimensions are not easy to handle, therefore feature selection plays a significant role. Document frequency thresholding achieves reductions in dimensionality by excluding terms having very high or very low document frequencies. Terms that occur in almost all documents in a collection do not provide any discriminating information. It is similar for terms that have a very low document frequencies, although those features might be helpful if they are not distributed evenly across classes. If the term "golf" has a low document frequency it can still help to discriminate hams and spams if it only occurs in ham messages or vice versa.

4.1 Information Gain

Information Gain is a technique originally used to compute splitting criteria for decision trees. Different feature selection models including Information Gain are described in [22]. The basic idea behind IG is to find out how well each single feature separates the given data set.

The overall entropy I for a given dataset S is computed by:

$$I = \sum_{i=1}^{C} p_i log_2 p_i \tag{2}$$

where C denotes the available classes and p_i the proportion of instances that belongs to one of the i classes. Now the reduction in entropy or gain in information is computed for each attribute or token.

$$IG(S, A) = I(S) - \sum_{v \epsilon A} \frac{|S_v|}{|S|} I(S_v) \tag{3}$$

where v is a value of A and S_v the number of instances where A has that value.

This results in an Information Gain value for each token extracted from a given document collection, documents are represented by a given number of tokens having the highest Information Gain values for the content-based experiments.

5 Ensemles of Classifiers for Spam Classification

All datasets used were indexed and all attachments as well as German and English stopwords taken from ranks.nl [2] were removed. After that tokenization was done via the Apache Lucene [3] standard tokenizer.

5.1 Decision Function

The scenario of several classifiers needs another *meta classifier* that evaluates the results of the single classifiers, the two basic methods to evaluate the results of an ensemble of classifiers are described below.

Majority Count Ensemble An instance is tagged as the class that is chosen by the majority of the classifiers. If a message is classified by nine classifiers [4] and it is classified as spam by five, spam is the result of the ensemble classification.

Winner(s) Take All Ensemble The 'winner take all' concept only uses the results from the classifier that is most confident of its choice. For k-NN classifiers this means that the classifier that found the most matching neighbours is the winner and is solely responsible for a given message. The ensemble of 3-NN classifiers used for the experiments often shows several winners (i.e. classifiers that find either three or zero neighbours for a given class). Therefore, majority count is used again. For example, if four classifiers find three spam neighbours and five classifiers find zero spam neighbours, a message will be classified as ham.

5.2 Ensemble Classification

By contrast to only a single classifier, multiple classifiers can be used to determine whether a message is spam or ham. Therefore the dataset was split by the initial corpora assignments, consisting of six corpora, each being assigned to either spam or ham. As shown in Algorithm 1, each of those six corpora is split into a training and a test set. A classifier is now trained for every plausible combination of corpora, i.e. only spam/ham combinations, being evaluated against the testing set of this one. Then the evaluation results are rounded and set to the majority count. In the case of six corpora each corpus is used for training three $k - NN$ classifiers (number of corpora not being the same class). E.g., the first ham

[2] http://www.ranks.nl/stopwords
[3] http://lucene.apache.org
[4] Three sets $\times$ two classes

corpus is used together with all spam corpora, three in total. The test set of that spam corpora is then evaluated by all three classifiers. Those results are summarized into one by majority count, whatever most of the classifiers assign to the message, is compared to it's actual class, yielding in very stable and good results.

Algorithm 1 Classification Using Ensembles of k-NN Classifiers

```
for each corpus do
  split into corpus.training and corpus.test
  for each corpus_comp ¬ corpus and corpus_comp.class ¬ corpus.class do
    knn = knn(corpus.data, corpus_comp.data, corpus.target, corpus_comp.target)
    result = knnforward(corpus.target)
  end for
  compute average accuracy
  compute voting accuracy
end for
```

6 Clustering of the TREC Spam Corpus

To come closer to a real world scenario, I decided to redo the experiments for another corpus, determining the subclusters using the Self-Organizing Map. The TREC spam corpus was used for the 2005 TREC experiments, comparable values exist. It is a corpus of about 92.000 messages, 42.000 ham, 50.000 spam. The sheer size of it would make experiments very difficult, so I decided to use a subset of 25 per cent each.

The next step was to subsample that corpus according to the data points mapped best to each unit, i.e. taking the best n mapped units from each unit and assume those to be the clusters for the ensembles. Those clusters did not improve the classification results, therefore cluster validation techniques were used to find an optimal number of units to fit the data.

6.1 Cluster Validation Techniques: Silhouette Value

The measuring of the clustering quality produced by either different algorithms or for different parameter settings is a vital issue in clustering. This is mostly used to find the right setting for the number of clusters. The following experiments assume that the number of units of the Self-Organizing Map is the number of clusters and therefore is the value to be optimized.

$$S(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \tag{4}$$

Where i is an index over all data vectors, $a(i)$ the average distance of i to all other vectors of that cluster, $b(i)$ the average distance of i to all data vetors in the closest cluster. The value lies between -1 and 1.

$$-1 \leq s(i) \leq 1 \tag{5}$$

$S(i)$ therefore is the silhouette value for data vector i, the overall silhouette value for a clustering is the average over all single silhouette values.

$$\sum_{i=1}^{n} S(i)/n \tag{6}$$

Let n be the number of instances. Analogously, the silhouette for single clusters can be defined as:

$$\sum_{j=1}^{m} S(j)/m \tag{7}$$

where c is the number of clusters and j the instances assigned to that cluster.

6.2 Cluster Validation of the Self-Organizing Map

Due to performance issues, the silhouette validations compares every unit to all other vectors assigned to that unit and to all vectors in the closest unit, I introduced modfications to fit the Self-Organizing Map scenario better.

Let each comparison be based on unit's weight vectors rather than the actual data vectors, $a(i)$ is defined as follows.

$$a(i) = dist(w(i), i) \tag{8}$$

$b(i)$ is defined as:

$$b(i) = dist(i, wc(i)) \tag{9}$$

Where $w(i)$ denotes the weight vector of the unit data point i is assigned to and $sc(i)$ denotes the weight vector of the closest unit. The overall silhouette computation is then based on those values for $a(i)$ and $b(i)$. The experimental evaluation from now on is done this technique, because it needs significantly less computational power. Hence, the quality of different clusterings can be compared by their Silhouette values. Furthermore the results can be used to visualize the correctness of the clustering.

6.3 Finally: Subsampling of the TREC Corpus

Table 2 lists the overall SOM Silhouette values for different numbers of units and the 25 per cent ham of the TREC corpus. The optimal number of units was chosen as 9 (3×3) and will used as the number of units for ham clustering from now on. Four subsamples were now chosen from each corpus (ham and spam). To better resemble the nature of the Self-Organizing Map clustering those subsamples were overlapping by about 10 per cent each. This resulted in eight clusters of about 1.000 to 2.000 messages each. All upcoming experiments

Table 2. Cluster validation experimental results for the 25 per cent ham messages of the TREC corpus using SOM Silhouette Validation.

Number of units	SOM Silhoutte value
4	.15606009446852034
5	.10004831518442062
6	.05601588449922962
7	.07248564371459906
9 (3×3)	.10156611162236466
9 (9×1)	.04731680857423294
12 (4×3)	.00125944139418076

use those clusters for training and are validated by large holdout test set of about 10.000 messages.

Clustering quality for spam messages also peaked for 9×9 units.

Figure 1 show the clustering of the TREC corpus.on a 3×3 Self-Organizing Map which will consequently be used .

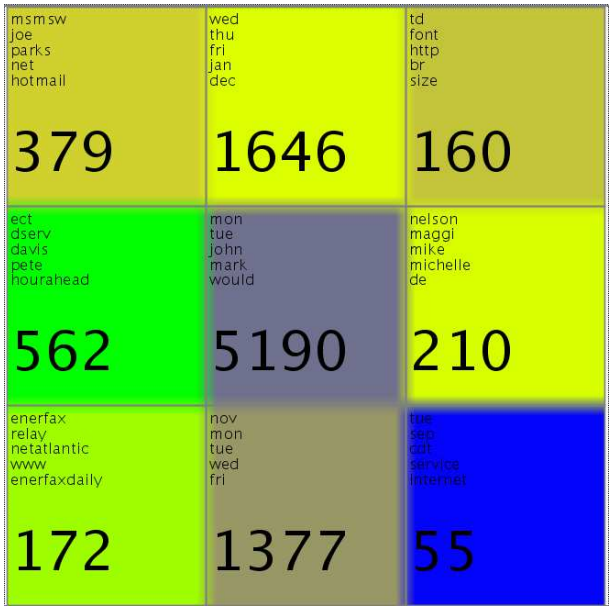

Fig. 1. Clustering of the TREC ham corpus on a 3×3 Self-Organizing Map.

7 Conclusions and Future Work

Results for both corpora used show that exploiting natural clusters in ham and spam corpora is promising in general. I, however, failed to prove that ensembles of k-NN classifiers outperform a single classifier that is trained on all the data vectors. I could show that the ensemble results are always better than the average result obtained from classifiers within the ensemble. Experimental results also showed that the decrease in performance from lower dimensionality is stronger for the monolithic classifier.

Table 3. Final results.

Result Type	**IG**	**Ind. IG**
Self-Organizing Map clusters of TREC data set		
Monolithic k-NN	0.960	X
k-NN ensemble	X	0.946

Future work may include classifiers based on individual feature selection but full training set since the classification accuracy of the single classifier trained on the full set could never be outperformed by any of the ensemble solutions.

Clustering quality may not be at an optimum. The Self-Organizing Map clustering is best suited for exploratory data analysis, the application of different clustering algorithms is more likely to produce better results. Particularly the cluster validation technique fails because the assumption that clusters and units are equivalent does not hold. Therefore cluster based algorithms seem more promising.

Since the more different the corpora are, the more the results benefit from ensemble classification, there's probably much improvement to be found in improving the clustering quality, which is definitely worth wile.

References

1. Apache Software Foundation: Spamassassin open-source spam filter. Internet, Open Source (2005) `http://spamassassin.apache.org/`.
2. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys **34(1)** (2002) 1–47
3. R.O.Duda, P.E.Hart: Pattern Classification and Scene Analysis. John Wiley (1973) Chapter 2, Bayes Decision Theory.
4. Graham, P.: A plan for spam (2002) `http://www.paulgraham.com/stopspam.html`.
5. Graham, P.: Better bayesian filtering (2003) `http://www.paulgraham.com/better.html`.
6. Delany, S.J., Cunningham, P., Coyle, L.: An assessment of case-based reasoning for spam filtering. Technical report, Dublin Institute of Technology (2004)

7. Berger, H., Köhle, M., Merkl, D.: On the impact of document representation on classifier performance in e-mail categorization. In: Proceedings of the 4th International Conference on Information Systems Technology and its Applications (ISTA'05). (2005) 1930
8. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J. Mach. Learn. Res. **2** (2002) 45–66
9. Gee, K.: Using latent semantic indexing to filter spam. In: ACM Symposium on Applied Computing, Data Mining Track. (2003) 460–464 Available: `http://ranger.uta.edu/~cook/pubs/sac03.ps`.
10. Cardoso-Cachopo, A., Oliveira, A.L.: An empirical comparison of text categorization methods. In Nascimento, M.A., Moura, E.S.D., Oliveira, A.L., eds.: Proceedings of SPIRE-03, 10th International Symposium on String Processing and Information Retrieval, Manaus, BR, Springer Verlag, Heidelberg, DE (2003) 183–196 Published in the "Lecture Notes in Computer Science" series, number 2857.
11. Neumayer, R.: Ensemble classification for spam filtering based on clustering of text corpora. Master's thesis, Trinity College Dublin, Faculty of Engineering and Systems Sciences, School of Engineering (2006)
12. Breiman, L.: Bagging predictors. Machine Learning **24**(2) (1996) 123–140
13. Dietterich, T.G.: Ensemble methods in machine learning. Lecture Notes in Computer Science **1857** (2000) 1–15
14. Adeva, J.J.G., Beresi, U.C., Calvo, R.A.: Accuracy and diversity in ensembles of text (0)
15. Breiman, L.: Random forests. Machine Learning **45**(1) (2001) 5–32
16. Salton, G., Lesk, M.: Computer evaluation of indexing and text processing. Journal of the ACM **15(1)** (1968) 8–36
17. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM **18**(11) (1975) 613–620
18. Wong, S.K.M., Raghavan, V.V.: Vector space model of information retrieval: a reevaluation. In: SIGIR '84: Proceedings of the 7th annual international ACM SIGIR conference on Research and development in information retrieval, Swinton, UK, UK, British Computer Society (1984) 167–185
19. Manning, C.D., Schutze, H.: Foundations of Statistical Natural Language Processing. MIT Press (1999)
20. Salton, G., McGill, M.: Introduction to Modern Information Retrieval. McGraw-Hill, New York, NY (1983)
21. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management **24(5)** (1988) 513–523
22. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In Fisher, D.H., ed.: Proceedings of ICML-97, 14th International Conference on Machine Learning, Nashville, US, Morgan Kaufmann Publishers, San Francisco, US (1997) 412–420
23. Gansterer, W., Ilger, M., Lechner, P., Neumayer, R., Strauß, J.: Anti-spam methods - state of the art. Technical report, Institute of Distributed and Multimedia Systems, Faculty of Computer Science, University of Vienna (2005)
24. Neumayer, R., Dittenbach, M., Rauber, A.: Playsom and pocketsomplayer: Alternative interfaces to large music collections. In: Proceedings of the Sixth International Conference on Music Information Retrieval (ISMIR 2005), London, UK (2005) 618–623
25. Neumayer, R., Lidy, T., Rauber, A.: Content-based organization of digital audio collections. In: Proceedings of the 5th Open Workshop of MUSICNETWORK, Vienna, Austria (2005)

26. Dittenbach, M., Neumayer, R., Rauber, A.: Playsom: An alternative approach to track selection and playlist generation in large music collections. In: Proceedings of the First International Workshop of the EU Network of Excellence DELOS on Audio-Visual Content and Information Visualization in Digital Libraries (AVIVDiLib 2005), Cortona, Italy (2005) 226–235 http://www.ifs.tuwien.ac.at/~andi/lop.html.
27. Neumayer, R.: Musical genre classification using a multi layer perceptron. In: Proceedings of the Fifth Workshop on Data Analysis (WDA2004). (2004)
28. Neumayer, R.: Clustering based ensemble classification for spam filtering. In: Proceedings of the Seventh Workshop on Data Analysis (WDA2006). (2004)
29. Neumayer, R.: The remote access generator engine, remote access via automatic code generation. Master's thesis, University of Vienna (2005)
30. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. 2nd edition edn. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2005)
31. Wong, M.W.: Spf - the sender policy framework (2004) http://spf.pobox.com.
32. Software, R.: Distributed checksum clearinghouse. Internet (2000) http://www.rhyolite.com/anti-spam/dcc/.
33. Prakash, V.V.: Vipul's razor (2005) http://razor.sourceforge.net.
34. Tobin, F.J.: Pyzor (2002) http://pyzor.sourceforge.net.
35. Zdziarski, J.: Dspam project open-source statistical anti-spam filter (2005) http://www.nuclearelephant.com/projects/dspam/.
36. Raymond, E.S.: Bogofilter open source mail filter (2005) http://www.bogofilter.org.
37. Warsaw, B.A., Hammond, M., Montanaro, S., Peters, T., Stone, T.: Spambayes statistical open source anti-spam filter (2005) http://spambayes.sourceforge.net/.
38. Back, A.: Hashcash - a denial of service counter-measure (2002) http://www.hashcash.org/papers/hashcash.pdf.
39. Abadi, M., Burrows, M., Manasse, M., Wobber, T.: Moderately hard, memory-bound functions. In: Network and Distributed System Security Symposium Conference Proceedings: 2003. (2003) http://www.isoc.org/isoc/conferences/ndss/03/proceedings/papers/2.pdf.
40. Rosenthal, D.: On the cost distribution of a memory bound function. In: LOCKSS TR2003-02. (2003)
41. Cunningham, S., Reeves, N., Britland, M.: An ethnographic study of music information seeking: Implications for the design of a musical digital library. In: Joint Conference on Digital Libraries. (2003)
42. Everitt, B.S.: The Analysis of Contingency Tables. first edn. Chapman and Hall (1977)
43. Sahami, Dumais, Heckerman, Horvits: A bayesian approach to filtering junk e-mail. In: AAAI. (XXX)
44. Downie, J.: Music information retrieval. In: Annual Review of Information Science and Technology. Volume 37. Information Today, Medford, NJ (2003) 295–340
45. Schedl, M., Knees, P., Widmer, G.: Discovering and Visualizing Prototypical Artists by Web-based Co-Occurrence Analysis. In: Proceedings of the Sixth International Conference on Music Information Retrieval (ISMIR'05), London, UK (2005)
46. Knees, P., Schedl, M., Widmer, G.: Multiple Lyrics Alignment: Automatic Retrieval of Song Lyrics. In: Proceedings of 6th International Conference on Music Information Retrieval (ISMIR'05), London, UK (2005) 564–569

47. Logan, B., Salomon, A.: A music similarity function based on signal analysis (2001)
48. Mahedero, J.P.G., Álvaro Martínez, Cano, P., Koppenberger, M., Gouyon, F.: Natural language processing of lyrics. In: MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia, New York, NY, USA, ACM Press (2005) 475–478
49. Logan, B., Kositsky, A., Moreno, P.: Semantic analysis of song lyrics (2004)
50. Baumann, S., Pohle, T., Shankar, V.: Towards a socio-cultural compatibility of mir systems. In: Proceedings of the 5th International Conference of Music Information Retrieval, Barcelona, Spain (2004)
51. Whitman, B., Ellis, D.: Automatic record reviews. In: Proceedings of the 5th International Conference of Music Information Retrieval, Barcelona, Spain (2004)
52. Pölzlbauer, G., Dittenbach, M., Andreas, A.: Visualization technique for self-organizing maps with vector fields to obtain the cluster structure at desired levels of detail. In: Proceedings of the International Joint Conference on Neural Networks, IJCNN, Montreal, Quebec (2005)
53. Marc Torrens, Patrick Hertzog, J.L.A.: Visualizing and exploring personal music libraries. In: ISMIR 2004, User Interfaces, Barcelona, Spain (2004) 421–424
54. Vignoli, F., van Gulik, R., van de Wetering, H.: Mapping music in the palm of your hand, explore and discover your collection. In Fox, E., Rowe, N., eds.: ISMIR 2004, User Interfaces, Barcelona, Spain (2004) 409–414
55. Foote, J.: An overview of audio information retrieval. Multimedia Systems **7**(1) (1999)
56. Bainbridge, D., Nevill-Manning, C., Witten, H., Smith, L., McNab, R.: Towards a digital library of popular music. In Fox, E., Rowe, N., eds.: Proc. ACM Conference on Digital Libraries (ACMDL'99), Berkeley, CA, ACM (1999) 161–169
57. Birmingham, W., Dannenberg, R., Wakefield, G., Bartsch, M., Bykowski, D., Mazzoni, D., Meek, C., Mellody, M., Rand, W.: MUSART: Music retrieval via aural queries. In: Proc. 2nd Ann. Symp. on Music Information Retrieval. (2001)
58. Liu, C., Tsai, P.: Content-based retrieval of mp3 music objects. In: Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM 2001), Atlanta, Georgia, ACM (2001) 506–511
59. Feiten, B., Günzel, S.: Automatic indexing of a sound database using self-organizing neural nets. Computer Music Journal **18**(3) (1994) 53–65
60. Zhang, H., Zhong, D.: A scheme for visual feature based image indexing. In: Proceedings of the IS&T/SPIE Conference on Storage and Retrieval for Image and Video Databases, San Jose, CA (1995) 36–46
61. Rauber, A., Pampalk, E., Merkl, D.: Using psycho-acoustic models and self-organizing maps to create a hierarchical structuring of music by musical styles. In: Proceedings of the 3rd International Symposium on Music Information Retrieval, Paris, France (2002) 71–80
62. Kohonen, T.: Self-Organizing Maps. 3rd edn. Volume 30 of Springer Series in Information Sciences. Springer, Berlin (2001)
63. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics **43** (1982) 59–69
64. Lidy, T., Rauber, A.: Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In: Proceedings of the Sixth International Conference on Music Information Retrieval (ISMIR 2005), London, UK (2005) 34–41
65. Vignoli, F., Pauws, S.: A music retrieval system based on user-driven similarity and its evaluation. In: Proceedings of the Sixth International Conference on Music Information Retrieval (ISMIR 2005), London, UK (2005) 3272 – 279

66. Tzanetakis, G., , Cook, P.: Marsyas: A framework for audio analysis. Organized Sound **4**(30) (2000)
67. Tzanetakis, G., , Cook, P.: Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing **10**(5) (2002) 293–302
68. Rauber, A., Frühwirth, M.: Automatically analyzing and organizing music archives. In: Proc. 5th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2001). LNCS, Darmstadt, Germany, Springer (2001)
69. Logan, B.: Content-based playlist generation: Exploratory experiments. In: Proc. 3rd Ann. Symp. on Music Information Retrieval (ISMIR 2002), France (2002)
70. Pampalk, E., Rauber, A., Merkl, D.: Content-based Organization and Visualization of Music Archives. In: Proceedings of the ACM Multimedia, Juan les Pins, France, ACM (2002) 570–579
71. Pampalk, E.: Islands of Music - Analysis, organization, and visualization of music archives. Journal of the Austrian Soc. for Artificial Intelligence **22**(4) (2003) 20–23
72. Rauber, A., Pampalk, E., Merkl, D.: The SOM-enhanced JukeBox: Organization and visualization of music collections based on perceptual models. Journal of New Music Research **32**(2) (2003) 193–210
73. Spevak, C., Favreau, E.: Soundspotter - a prototype system for content-based audio retrieval. In: Proceedings of the 5. International Conference on Digital Audio Effects (DAFx-02(), Hamburg, Germany (2002)
74. Ultsch, A., Siemon, H.: Kohonen's self-organizing feature maps for exploratory data analysis. In: Proceedings of the International Neural Network Conference (INNC'90), Dordrecht, Netherlands, Kluwer (1990) 305–308
75. Welsh, M., Borisov, N., Hill, J., von Behren, R., Woo, A.: Querying large collections of music for similarity. Technical Report UCB/CSD00 -1096, U.C. Berkeley Computer Science Division (1999)
76. Wold, E., Blum, T., Keislar, D., Wheaton, J.: Content-based classification search and retrieval of audio. IEEE Multimedia **3**(3) (1996) 27–36
77. Zwicker, E., Fastl, H.: Psychoacoustics, Facts and Models. 2 edn. Volume 22 of Series of Information Sciences. Springer, Berlin (1999)