Learning from User Behavior in Image Retrieval: Application of the Market Basket Analysis

Henning Müller, Thierry Pun
Computer Vision Group, University of Geneva
24 Rue du Général Dufour,
CH-1211 Genève 4, Switzerland
Henning.Mueller@cui.unige.ch

David Squire
Computer Science and Software Engineering
Monash University
Melbourne, Australia
David.Squire@csse.monash.edu.au

Abstract

This article describes an approach to learn feature weights for content-based image retrieval (CBIR) from user interaction log files. These usage log files are analyzed for images marked together by a user in the same query step. The problem is somewhat similar to one of the traditional data mining problems, the market basket analysis *problem, where items bought together in a supermarket are analyzed. This paper outlines similarities and differences between the two fields and explains how to use the interaction data for deriving a better feature weighting.*

Experiments with existing log files are done and a significant improvement in performance is reached with a feature weighting calculated from the information contained in the log files. Even with several steps of relevance feedback the results remain much better than without the learning, which means that not only information from feedback is taken into account earlier, but a better quality of retrieval is reached in all steps.

1. Introduction

Relevance feedback is regarded as one of the most powerful techniques to improve the results of content-based image retrieval systems (CBIRSs). Much has been written about different ways to implement relevance feedback [19, 24, 26, 29, 32, 35]. Most systems calculate the relevance feedback from one query step to the following query step. Like this, not the entire query session is used for calculating the next results but only the preceding step. In *FourEyes*, across session learning is proposed [17] and also, in [33], it is proposed to learn over several temporal scales. Some systems like the image browsers *PicHunter* [6] or *TrackingViper* [22] take several steps of user interaction into account to find a target image in a database. In [13], an approach to learn query concepts over several interaction steps without using seed images is proposed, using support vector machines.

In [12], a system is presented that groups images into clusters and changes these clusters when they are marked with contradicting relevances by another user. For this tool to be effective, all images of the database should have been selected by several users and should have been marked by at least one of these users. In [11], old user judgments are used to propose new images to users based on the items they have already marked. This *collaborative filtering* is applied to art images in a museum. A web demonstration is accessible at `http://abyss.eurecom.fr:1111/AWM/login.html`. Amazon (`http://www.amazon.com/`) also employs this collaborative filtering technique to propose books to potential customers.

In [14], a method to store correlations between images is proposed that promises good results when all images in the database are marked at least once. Large image databases will require extremely large storage capacities to implement this technique. The data to train the system is gained from automatically created usage log files which somewhat limits its expressive power.

The use of log files to discover knowledge is also very common in many other research areas in connection with the Internet. Log files are used to adapt web pages or web accessible systems to the users needs [36]. In [4], the behavior of users within web pages is analyzed to improve the page layout.

Experimental results in [18] have shown strong improvements in CBIRS performance when using feature weights that are calculated with the help of usage log files. This study uses images marked together in the same relevance feedback step for the calculation of a new feature weight. Therefore, it seems logical to take a more formal approach to exploit the usage log files of a CBIRS.

The stated problem has many similarities to the *market basket analysis* often described in the data mining literature. Supermarkets have large files of items purchased together by a customer at the same shopping. One would like to know which combinations of these items occur significantly often and how association rules can be derived from these data efficiently. In [1, 2, 9, 10, 28], efficient algorithms are described to solve the task. An explorative evaluation of all combinations is infeasible as there are thousands of items and several hundreds of them can be purchased together at the same shopping trip. Thus, we need algorithms to efficiently filter the data and follow promising groupings of items. Association rules can be derived from these data. A sort introduction of association rules is given in Section 3.1. without going into details.

Section 2. describes the *Viper* CBIRS we used to validate our approach. It will become clear that the specific architecture of the *Viper* CBIRS has been a major factor for the success of this study. Section 3. introduces association rules and compares the *market basket analysis* problem from traditional data mining with our problem of images that are marked together. The section also explains how the data is reduced to make it usable for our purpose. Section 4. shows how the actual feature weighting is calculated and how the learned information is integrated into the image feature weighting scheme. In Section 5.we include the calculated weights into our system and compare the results of the system before and after the use of the additional probabilistic feature weighting. The last section critically discusses the experimental outcome and gives ideas for future work.

2. The *Viper* system

For the evaluation of our approach we use the log files created by the web demonstration version of the *Viper* system `http://viper.unige.ch/`, which has been developed at the University of Geneva. A stable version of this program called *Gift* (GNU Image Finding Tool) can be downloaded free of charge at `http://www.gnu.org/software/gift/`. The system is described in more detail in [32].

2.1. System Architecture

The main difference of *Viper*/*GIFT* compared to other systems is the presence of a very large number of more than $85,000$ possible features. Most images contain between $1,000$ and $2,000$ of these features. The access method to the features is the *inverted file*, which is the most common access method used in text retrieval (TR), and which is efficient in very sparsely populated spaces. Thus, *Viper* allows for a fast and efficient access to this large number of features because the search space is limited to the features present in the query images and the efficiency of the inverted file.

The emphasis of the project is on user interaction. Hence, it embeds several interaction strategies using several steps of positive and negative feedback. Both online and offline learning are employed in the system. *Viper* offers a good flexibility for learning as it has a very large number of features for the creation of feature weights. Especially the extensive use of negative feedback has shown to be very effective [19] and is very important for the learning approach proposed in this paper.

2.2. Multimedia Retrieval Markup Language (*MRML*)

MRML (`http://www.mrml.net/`, [21]) was originally developed to separate the user interface from the actual search engine. It is an XML-based protocol and allows us to log all the communications between the user interface and the server in a human-readable and simple manner. It is therefore based on *MRML* logfiles of the user interaction that we develop this study.

An example of a simple query by example(s) (QBE) in *MRML* with positive and negative input can be seen here:

```
<mrml session-id="1" transaction-id="44">
<query-step session-id="1"
  resultsize="30"
  algorithm-id="algorithm-default"
  <user-relevance-list>
    <user-relevance-element
      image-location=
        "http://viper.unige.ch/images/1.jpg"
      user-relevance="1"
    <user-relevance-element
      image-location=
        "http://viper.unige.ch/images/2.jpg"
      user-relevance="-1"
  </user-relevance-list>
</query-step>
</mrml>
```

This example has one image marked relevant (”1”) and another one marked non-relevant (”-1”). For a query session with the web interface, images that are present on screen but not marked by the user, thus implicitly labeled neutral, are also transmitted. These images are not included in the evaluation of this article because they do not contain as much information as images being purposely marked by the user.

2.3. *Viper* Features

The system used for this study implements four different *groups* of image features for local and global color and texture measures that are described in more detail in [32]:

- A global color histogram based on the HSV color space which roughly corresponds to the human color vision [31]; The HSV space is quantized into 18 hues, 3 saturations, 3 values and four levels of grey which results in $18 \cdot 3 \cdot 3 + 4 = 166$ possible color histogram features;
- Local color blocks at different scales for fixed-size regions by using the mode color for each of the fixed blocks; the image is recursively partitioned into four equally sized blocks and each block is subsequently partitioned again four times; This results in $(4 + 16 + 64 + 256) \cdot 166 = 56,440$ possible color block features where each image contains a maximum of 340;
- Global texture characteristics represented by the histograms of the response to Gabor filters of three different scales and four directions that are quantized into 10 different bins with the lowest response being discarded; Gabor filters are known to be a good model for the human perception of edges [15]; This results in $4 \cdot 3 \cdot 9 = 108$ possible Gabor histogram features;
- Local Gabor features at different scales and in different regions by using the smallest blocks of the local color features and applying Gabor filters with four different directions, three scales quantized into 9 strength to these blocks; This results in $4 \cdot 3 \cdot 9 \cdot 256 = 27,648$ possible Gabor block features where an image can contain a maximum of 3072 of these features.

This results in a maximum of $84,362$ possible features where an image contains between 341 and 3,686 features, usually around 1,500. With respect to our learning algorithm every single of these features is regarded separately and can obtain a learned weight. There are two distinct sorts of features with similar characteristics. The color and Gabor *histograms* are relatively frequent in the collection of images. Their collection frequency (cf) is normally high. Their frequency in the document (tf – term frequency) is the percentage of the color our a particular Gabor response with respect to the entire image. The color and Gabor *block* features are local features and they are usually rather rare, meaning that the cf is low, whereas their tf is always 1 in a single image as they either occur or not.

These features are only low level features, but because of their high number, we believe that complex queries can be constructed with them. Higher level features such as image regions may provide even better results with the learning, but even in this case we would still suffer from the semantic gap between the semantics the user is looking for and the visual content the system can offer.

2.4. Weighting schemes

We have implemented several weighting schemes known from the TR literature [25]. They are all based on the collection and document frequencies of the features. For the experiments in this paper, we use the *inverse document frequency* weighting, which weights the features in the following way (see [32] for more details):

$$\textit{feature weight}_j = \frac{1}{N}\sum_{i=1}^{N}\left(tf_{ij}\cdot R_i\right)\cdot log^2\left(\frac{1}{cf_j}\right), \tag{1}$$

where *tf* is the *term frequency* of a feature, *cf* the *collection frequency* of a feature, j a *feature number*, q corresponds to a query with $i = 1..N$ input images, k is one *result image* and R_i is the *relevance* of an input image i within the range $[-1;1]$.

Then a $score$ is assigned to a possible result image k with query q containing features $1..j$:

$$\textit{score}_{kq} = \sum_j\left(\textit{feature weight}_j\right), \tag{2}$$

We can see in Equation 1 that the final result mainly depends on the collection frequency of a feature. Rare features are weighted high, whereas features very common in the collection are weighted low because they contain less information. The term frequency of a feature in the query images has a minor influence, as well. We can see in Equation 2 that the final score for an image is the sum of importances of all the features present in this image.

3. Analysis of the log files

This section shortly describes association rules and then, how we can reduce the large amount of data to find images being marked together several times. The reduction process is analog to a simple reduction of data described as the *market basket analysis* in [1].

3.1. Association rules

An *association rule* is an expression or rather implication of the form $I_a \rightarrow I_b$ where both I_a and I_b are subsets of a set of items or images $\mathcal{I}$ and part of a set of transactions or item sets $\mathcal{T}$. The probability for such an association rule or implication between two sets of items is also called the *rule confidence* and can be calculated easily:

$$P(I_a \rightarrow I_b) = \frac{\varphi(I_a I_b)}{\varphi(I_a)}, \tag{3}$$

where φ is the frequency of the single item or item pair in the set of transactions $\mathcal{T}$, respectively. Another important factor for determining the quality of a rule is the *support* for a certain rule which is the fraction of all item sets in $\mathcal{T}$ that contain the elements of the sets I_a and I_b used in an association rule. In [1], item sets with a support above a defined threshold are called *large item sets*. There are also other factors for determining the quality of a rule such as the correlation [5] but this article cannot describe all these factors and algorithms in detail.

Articles for efficient algorithms to extract association rules are [1, 2, 7, 10, 16, 27, 37]. A comparison of various methods is given in [9] where run–times of classical algorithms are compared. In [5], an interesting approach comparing association rules with correlations is given. An algorithm to find spatial co–locations is explained in [28].

3.2. Comparison with the market basket analysis

As mentioned earlier, the *market basket analysis* aims at characterizing articles (images) that are often marked together. Although there are many similarities between finding images marked together in queries and items bought together in a supermarket, there are also some differences we have to consider. The main differences are:

1. Each image can only be marked once in a query whereas in a supermarket, an item can be bought several times.

2. An image can be marked in three different ways: relevant, non-relevant or neutral, whereas an item in a supermarket is either bought or not.

3. The sizes of the basket in a supermarket can vary in a much larger range compared to image retrieval , where users normally do not mark a very large number of images.

4. We have less queries with the system than we have items, whereas in a supermarket much larger datasets of items bought together are available than there are items in the shop. Thus, every item in a shop is bought every once in a while, whereas not every image is marked as regularly.

5. Items that most frequently imply other items are most important for a supermarket, whereas for us it is more important to find the images that are marked relevant after several steps of feedback. This is the case because items (images) very often marked together are frequently retrieved together anyways, whereas we also want to retrieve items that are not always retrieved in the first query step. These items are further away from the query in similarity space but can be discovered when using feedback.

The additional information described in 1.) is not used for the qualitative analysis described in [1] anyways. Therefore, we can discard the unavailability of this information. The solution we use for coping with 2.) is that in the first step we regard an image marked “relevant” and the same image marked “non-relevant” or “neutral” as different items. We finally did not evaluate the images marked “neutral”. These images were not actively marked by the user, so the information content can be interpreted in different ways. The user might have been too lazy although they were relevant for him, or the user might have thought that they are not relevant for him but feared that marking them negatively could worsen the results, or the user may simply not have looked at them. Clearly, images marked neutral are in between the relevant and non-relevant images. 3.) is not as important as we can theoretically have larger groups of images as well, and most shopping data sets will not be much larger than our image data sets. Remark 4.) is important because we cannot prune out as much information as can be done from very large log files. We have to rely more on the quality of the data, because otherwise we can not get much information from the log files. With the time, we will get larger log files as well, which can be evaluated in a better way. 5.) also limits the amount of information we can filter out for further processing. Some images often marked together with different images in a different context might still contain very valuable information. On the association rule level, we thus do not set any minimum requirement for the probability of an association rules.

3.3. Reduction of the log files

We have several steps to reduce the data of the log files for the final analysis, when applying an algorithm similar to [1]. We choose such a simple a–priori algorithm for the calculation of the association rules because it is easy to use and to understand and we are only using a subset of the association rules anyways. The comparison between different efficient algorithms in [9] shows that the differences between the algorithms with respect to efficiency are rather small.

1. Reduce the log files to image sets marked together because the rest of the communication descriptors in *MRML* is not being needed.

2. Find images and then image pairs that occur more than once, so–called *large images* or *large image sets*, respectively, as they are named in [1]. This corresponds to the fact to have support above a certain threshold. We use a rather low threshold to have a maximum of rules for the feature weight calculation.

3. Use only image pairs we are really interested in as explained in Table 1 and described below.

4. Calculate Probabilities of association rules for all *large image sets* we found in 2.), and only use those above a certain probability threshold (which corresponds to the confidence of a rule).

At this stage, we concentrate on association rules between pairs of images, that is sets of size 2 only. Association rules with more than two images are not evaluated for the moment.

For us, only images marked together positively in the same query step or image pairs where one image is marked positively and the other one negatively are of interest, because images can be marked negatively or left neutral in the query for completely different reasons. Table 1 shows the combinations of image pairs we are interested in. Only the combinations marked with “++” are used is this paper. We could also use the ones marked with “?”, but we decided not to as they were not actively marked by the user. The combinations marked with “0” cannot be used. Images marked together negatively, for example, can be marked negatively for completely different reasons.

The analysis of the log files is performed completely automatically with perl scripts. The log files are in *MRML* format which is based on XML, so a generic XML parser is used to extract information about images being marked together in one query step. We then calculate the frequencies of images with a specific marking (1=relevant, -1=non-relevant). Our limit for images being called *large images* is that they have to occur at least twice. With larger log files, this limit of 2 may be increased but so far our log files are rather small and we do not want to let any information being unused. Images marked only once with a certain relevance are then removed from the query groups of images marked together in a step. Now each

	relevant	neutral	non-relevant
relevant	++	?	++
neutral	?	0	0
non-relevant	++	0	0

Table 1. Possible combination of markings of image pairs. The ones we our using for our approach are marked with “++”.

query which still contains multiple images is read in and all the possible combinations of two images are created and stored with their frequency. Pairs that occur less than twice are deleted from the list.

There are further important differences between the market basket data problem and ours. We are not necessarily interested in the association rules that occur the most frequently. If images are marked together very frequently, it means that they already have a high similarity and show up together very often. We are much more interested in relevant images that show up after one or more steps of feedback. These image pairs might not have a very high frequency for the association rule but they are indeed more important than images in rules with a higher frequency, that appear each time on screen together. For this reason we do not set a limit for the probability for association rules but also evaluate rules with small probabilities.

4. New weighting for the features

The data reduction explained in section 3. leaves us with a list of images marked together more than once, as well as with the frequencies with which they are marked together and alone. This is used to create association rules for the connection of images. These association rules can be relevant/relevant or relevant/non-relevant combinations.

4.1. Using association rules

From each of the pairs consisting of images I_a and I_b we can construct two association rules: $I_a \rightarrow I_b$ and $I_b \rightarrow I_a$. The probabilities for these association rules can easily be calculated from the frequencies of the two images marked together and marked alone as in Equation 3.

Each image pair thus produces two association rules and their probabilities. “*Positive association rules*” are rules where the two images in the pair were marked as *relevant* and “*negative association rules*” are rules where one of the two images is marked *non-relevant* and the other one *relevant*. *Positive* and *negative association rules* are noted $(I_a \rightarrow I_b)_+$, $(I_a \rightarrow I_b)_-$, respectively.

4.2. From Images to features

With the association rules, we have a connection between images, but we want to learn on a feature basis to be more general and also learn for images not yet marked by any user. Thus, we have to make a connection between the association rules for images and the features that these images contain.

First, we have to think about what makes a feature a good or positive feature in our sense, see Equation 4:

$$P(f_j \textit{good}) \quad = \quad P(f_j \textit{predicts relevance}) \wedge P(f_j \textit{does not predict irrelevance}); \tag{4}$$

where f_j is a a feature and P the probability. This means that a feature is a good feature if it is good at predicting relevance and bad for predicting irrelevance. If a feature predicts relevance it means that relevant images are generally returned when using this feature for querying whereas a feature that predicts irrelevance returns generally irrelevant images when used for querying.

Now, Equation 5 calculates the probability P that a feature predicts relevance by using the frequencies of features being in both images of an association rule or only in the first one.

$$\begin{aligned}
P(f_j \textit{predicts relevance}) &= P((f_j \in I_a) \Rightarrow (f_j \in I_b) | I_b \textit{is relevant for} I_a) \\
&= P((f_j \in I_a) \Rightarrow (f_j \in I_b) | (I_a \rightarrow I_b)_+) \\
&= P(((I_a \rightarrow I_b)_+ \wedge f_j \in I_a) \Rightarrow (f_j \in I_b)) \\
&= P(((f_j \in I_b) | (I_a \rightarrow I_b)_+ \wedge f_j \in I_a)) \\
&= \frac{\varphi(((I_a \rightarrow I_b)_+ \wedge f_j \in I_a) \Rightarrow (f_j \in I_b))}{\varphi((I_a \rightarrow I_b)_+ \wedge f_j \in I_a)} \\
&= \frac{|(f_j \in I_a \wedge f_j \in I_b); (I_a \rightarrow I_b)_+|}{|(f_j \in I_a); (I_a \rightarrow I_b)_+|},
\end{aligned} \tag{5}$$

where φ is the frequency and $|A|$ is the cardinality of set A.

The same can be done for *negative association rules*. There are two possible ways for negative connections: $I_a^- \rightarrow I_b^+$ and $I_a^+ \rightarrow I_b^-$. These two cases are treated the same way. In a manner analog to *positive association rules*, the *negative association rules* are written $(I_a \rightarrow I_b)_-$. The probability for a feature predicting irrelevance is shown in Equation 6.

$$P(f_j \textit{predicts irrelevance}) = \frac{|(f_j \in I_a \wedge f_j \in I_b); (I_a \rightarrow I_b)_-|}{|(f_j \in I_a); (I_a \rightarrow I_b)_-|} \tag{6}$$

Combining the two values we get Equation 7, which is the first factor F_1 we calculate for each feature.

$$F_{1_j} = P(f_j \textit{good}) \quad = \quad P(f_j \textit{predicts relevance}) \cdot P(f_j \textit{does not predict irrelevance}). \tag{7}$$

When calculating this factor it becomes unfortunately clear that our low level features are very often in one of the images of an association rule, but only rarely in the two, thus creating many extremely low values and being not very specific about the feature quality. Another problem is how to obtain a probability for features not present in any of the image pairs which should be a value between the good and the bad features. As these features seem to be in almost no image, we can discard these features and let their probability become 0.

To solve the above problem, we limit our calculations to the times a feature occurs in both images of an association rule, either in a positive or negative one. This leads to Equation 8, which turns out to be much more discriminative between “good” and “bad” features.

$$F_{2_j} = P(f_j \textit{good}) = \frac{|(f_j \in I_a \wedge f_j \in I_b); (I_a \rightarrow I_b)_+|}{|(f_j \in I_a \wedge f_j \in I_b); (I_a \rightarrow I_b)_- \vee (I_a \rightarrow I_b)_+|} \tag{8}$$

It will be shown in the next section that this second factor F_2 as described in Equation 8 also leads to better results. If a feature does not occur in any of the association rules we give it a value of 0.5 as we do not have any information to give it another probability. Very frequent features will also get a weighting close to 0.5 when they appear in almost every *positive* and *negative association rule*.

4.3. Combining our probability factor with the *idf* weighting

To evaluate the performance of our system, in addition to the probability of the feature that we just calculated, we also want to include the probability into our normal *idf* weight described in Equation 1. Therefore, we include the probability into the old feature weight calculation as an additional *factor* as shown in Equation 9.

$$\textit{feature weight}_j = \frac{1}{N} \sum_{i=1}^{N} \left(\textit{factor} \cdot \textit{tf}_{ij} \cdot R_i \right) \cdot log^2 \left(\frac{1}{\textit{cf}_j} \right), \qquad (9)$$

To calculate the score for an image we again use equation 2. To compare the retrieval results for a number of different factors we use both factors F_1 and F_2, the squared (F_1^2, F_2^2) to check out a stronger influence of the factors, as well as F_2^3. Because the results in [18] suggested that it might be better to keep even the negative factors with a low value and not let them become zero, we calculated F_2^2 as well in a version where the lowest value is 0.05, so these features are not completely discarded for negative feedback.

Such a calculated factor expressing the quality of a feature can be calculated regularly based on the user feedback. So far, we do not have an automatic updating scheme but the factor calculation is started by hand. To have a completely automatic system with regular updates does not seem too hard to implement. The values for all association rules would need to be stored and updated with the new interaction data. This can avoid to reanalyze the entire interaction data each time. As short–term feedback algorithms are used in the retrieval system as well, an updating after every query step is not necessary.

5. Experimental results

For our experimental results, we analyzed 800 MB of user interaction data in *MRML* from the *Viper* CBIR system. In the first reduction step we filtered the data to get all the images that were present in at least one query step. This led us to $3,258$ queries with more than $80,000$ images being marked in total. These images are from ten different databases, but most queries were done with three of the databases.

We decided to use the database of the University of Washington [3] for this evaluation because the database is freely available and thus the results are reproducible. This database contains 922 images being separated in 13 image groups containing very different numbers of images (between 27 and 256 per group).

In the second step, we regarded only image pairs which occurred more than once together. Over all databases, we have $61,361$ such pairs with $21,727$ of these pairs being from the Washington database. $7,930$ of these pairs for the Washington database are positive connections (leading to *positive association rules*), $8,131$ are a connection between a positive and a negative image (leading to *negative association rules*) and $5,666$ are a connection between a positive and a neutrally marked image, which is not used for this evaluation. We finally ended up with $32,122$ association rules for image pairs to calculate similarities.

5.1. Performance measures used

For the evaluation of the system performance we use a set of measures that are well known from text retrieval and, for example used in the TREC (Text REtrieval Conference) performance comparison [8, 34]. These measures have been proposed and used for image retrieval as well [20, 30]:

- $Rank_1$ and $\widetilde{Rank}$: rank at which first relevant image is retrieved, normalized average rank of relevant images (see Equation 10).
- $P(20)$, $P(50)$ and $P(N_R)$: *precision* after 20, 50 and N_R images are retrieved.
- $R_P(.5)$ and $R(100)$: *recall* at *precision* .5 and after 100 images are retrieved.
- PR graph.

A simple average rank is difficult to interpret, since it depends on both the collection size N and the number of relevant images N_R for a given query. Consequently, we normalize by these numbers and propose the *normalized average rank*, $\widetilde{Rank}$:

$$\widetilde{Rank} = \frac{1}{NN_R}\left(\sum_{i=1}^{N_R} R_i - \frac{N_R(N_R+1)}{2}\right) \tag{10}$$

where R_i is the rank at which the ith relevant image is retrieved. This measure is 0 for perfect performance, and approaches 1 as performance worsens. For random retrieval the result would be 0.5.

To evaluate the performance of a system with relevance feedback we use an algorithm described in [19]. This algorithm assumes that a user would feed all the relevant images on screen back as positive feedback and all the non–relevant images back as negative feedback. Relevance feedback can thus be generated from the ground truth of the image database and the query result of the retrieval system. We take the first 20 images of the system response for the generation of positive and negative feedback images. Thus, system evaluation can be performed in a completely automatic way.

5.2. Evaluation on the database of the University of Washington

From the $32,122$ association rules, we calculated the different factors derived in the preceding section. The retrieval results for the factors are compared for a first query step and for two feedback steps with automatically generated feedback.

Figure 1 shows a *Precision/Recall graph* (*PR-graph*) for the first query step. We can clearly see that our first factor leads to a really bad performance in the beginning of the *PR-graph* whereas the others all lead to similar, improved results. In the middle part of the *PR-graph* especially the squared and cubed second factor give good results of up to 20% better than the original weighting. As the results for F_1^2 and F_1^2 where the features remain with a frequency of at least 0.05 are almost the same in the *PR-graph*, the results are not shown in the table.

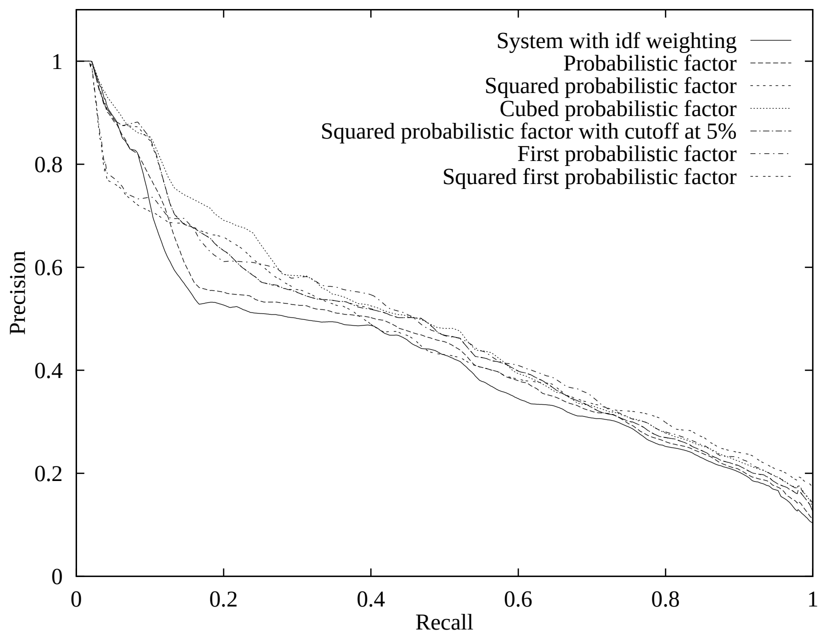

Figure 1. *PR-graph* **evaluation of the different factors without using feedback.**

In Table 2, we can see the other performance measures used for CBIR evaluation. In terms of rank measures, the first factor gives very poor results and the first relevant image is also found rather late compared to the second factor. Regarding the ranks, the normal *idf* system is the best, but only marginally better than the second factor. For a real user, the *precision* after 20 or 50 images, depending on how many images he looks at on screen, is normally the most important measure. With respect to these measures, the systems with the additional factor look much better than the normal *idf* system. The new results are up to 7% better, which is a significant improvement. With respect to the *precisions*, the second factor (especially when cubed), gives the best results.

Even more interesting are the results after the first step of relevance feedback. Because the system uses old relevance feedback to improve the results for further queries, we could assume that, in this case, part of the relevance feedback is already taken into account and the improvements could be less strong. Figure 2 shows that with relevance feedback, the results improve even more. Especially with the second factor (when cubed), we gain up to 15 % in the middle of the graph. We can also see that the results get worse in the beginning of the graph, the higher a power of F_2 we use, but that the curve gets much better in the middle and in the end of the graph for higher powers. Similarly, the first factor F_1 is much worse in the beginning, but gets better in the middle and end of the graph although by far not as good as the second factor.

Table 3 underlines the results already obtained from the *PR-graph*. The first factor does not give very good results for any of the measures. The *precision* values are still better than the original *idf* system, but the ranks are much worse, and the results of the squared factor get even worse than when using the *idf* weighting. F_2 though, seems to be much better, even for the early *precision* values which are up to 10% better. The rank values are also slightly better, and only $R(P(.5))$ drops below the original value for the squared and cubed version.

The second step of feedback in Figure 3 shows that the first factor F_1 gives bad results, but that the second factor F_2

	idf	F_2	F_2^2	F_2^3	F_1	F_1^2
N_{rel}	65.14	65.14	65.14	65.14	65.14	65.14
rel_1	1.5	1.5	1.79	4.79	7.93	11.43
$R(P(.5))$	.3799	.4066	.3653	.3945	.3791	.2899
$\overline{Rank}$	176.4	177.6	179.8	183.9	204.8	223.3
$\widetilde{Rank}$	.1583	.1597	.1620	.1664	.1895	.2098
$P(20)$	.5393	.5786	.6071	.6	.6071	.5857
$P(50)$	.4057	.4157	.4314	.4371	.4486	.4271
$P(N_r)$	.3883	.4067	.4214	.4337	.4349	.4151
$R(100)$	.4839	.4936	.5103	.5135	.5002	.4782

Table 2. Performance measures for different factors in the first query step.

	idf	F_2	F_2^2	F_2^3	F_1	F_1^2
N_{rel}	65.14	65.14	65.14	65.14	65.14	65.14
rel_1	1	1	1	1.43	2.79	11
$R(P(.5))$	.5157	.5799	.4483	.4965	.437	.4184
$\overline{Rank}$	162.3	159.9	158.5	156.7	189.6	205.3
$\widetilde{Rank}$	.1429	.1402	.1387	.1367	.1728	.1901
$P(20)$	.6857	.7607	.7642	.7857	.7071	.7071
$P(50)$	.5014	.5443	.5643	.5771	.5357	.5214
$P(N_r)$	.4957	.5504	.5680	.5887	.5303	.5129
$R(100)$	.5640	.5935	.6030	.63	.5737	.5519

Table 3. Performance measures for different factors with one step of feedback.

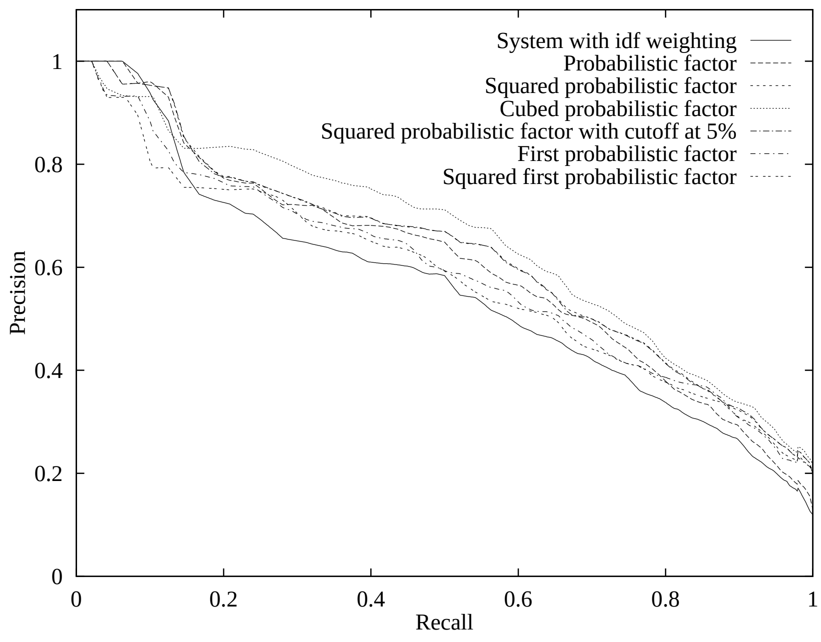

Figure 2. *PR-graph* **evaluation of the different factors with feedback.**

delivers better results from the beginning on than the *idf* systems. Only F_2^3 is slightly worse in the beginning part of the graph, but eventually becomes the best curve in the middle part, more than 10% better than the original curve of the *idf* weighting.

The measures in Table 4 again underline the results from the *PR-graph*. The first factor is not significantly worse for queries with multiple images whereas the second factor seems to be getting better, the more images there are in the query. The *precision* values are up to 10% better and also the ranks are better than in the *idf* system. The cubed system does not have a relevant image as top-ranked response for every query, but this can be explained with one rather bad query, whereas all other queries are processed significantly better.

We can see clearly that the first factor F_1 was unfortunately not suitable for queries with feedback although the results for one-image queries were quite good. The second factor we calculated leads to very good results. The fact that the results are even improving stronger for feedback queries is surprising as we thought that the feedback is partly already taken into account with the calculated factor. This does not seem to be the case and the results are very good.

As always, the results depend on the quality of relevance feedback we can gain from the log files. If people use much positive and negative feedback, the results can be very good and a factor in a higher power can be used, which promises the best results. Even if the quality of the feedback is not clear, a simple factor can be used, and the results in our test show that this significantly improves the performance of a system. As our feedback is taken from a web demonstration, we cannot necessarily trust the user.

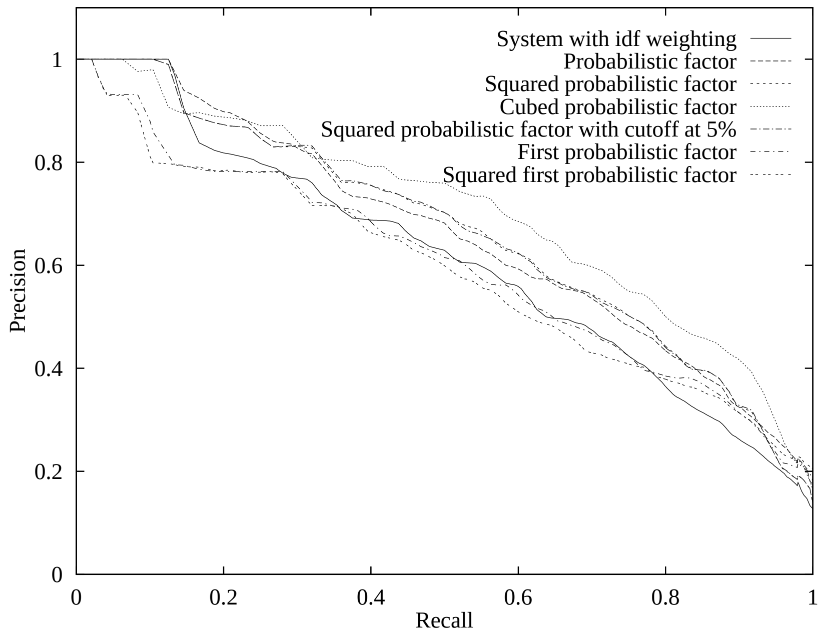

Figure 3. *PR-graph* **evaluation of the different factors with two steps of feedback.**

6. Conclusions and future work

This paper shows the clear connection between learning efforts in CBIR and classical data mining problems such as the *market basket analysis*. It also shows that significant performance improvements can be achieved by taking the relevance feedback of old queries into account when calculating the similarities for new queries. This gain in quality will be better the more relevance feedback exists for an image database. In this context, the quality of the relevance feedback definitely also plays a significant role.

We can also think of considering an entire query session as a "*market basket*" rather than every single query, because the final result might better correspond to what the user really wants. All the early query steps are then just repeating parts of the final results.

Here, we do not attempt to learn feature qualities from one database to use it for another database, because with our system using low level features, this is unlikely to lead to good results. However, with very specific or higher level features, we think that good results can be obtained with such a learning technique.

In the approach presented in this paper, we created a combination of a frequency-based and a probabilistic weighting, which leads to good results. Once we will have large amounts of feedback in our log files, the goal will be to get to a fully probabilistic weighting, possibly using the frequency-based weighting as a starting point to measure feature importance and then adapt the values based on the feedback. Another idea with a high chance of success is a hierarchic learning on different levels, i.e. on a user, database and overall basis. In this paper, we only describe the learning over a database, but the learning might even be more effective when the feedback of the user working with the system is taken into account in a stronger way than as a general database knowledge.

	idf	F_2	F_2^2	F_2^3	F_1	F_1^2
N_{rel}	65.14	65.14	65.14	65.14	65.14	65.14
rel_1	1	1	1	1	3.79	10
$R(P(.5))$	.58	.5558	.5775	.6284	.5185	.4375
$\overline{Rank}$	149.8	134.49	136	133.9	188.3	198.8
$\widetilde{Rank}$	.1292	.1123	.1140	.1116	.1713	.1830
$P(20)$	.775	.8143	.8107	.8464	.7393	.7286
$P(50)$	.5414	.5843	.5828	.6	.5271	.5214
$P(N_r)$	.5402	.5871	.5933	.6229	.5248	.5166
$R(100)$	.6153	.6594	.6539	.6838	.5794	.5692

Table 4. Performance measures for different factors with two steps of feedback.

7. Acknowledgments

This work was supported in part by the Swiss National Foundation for Scientific Research (grant no. 83R-064270).

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD Conference*, pages 207–216, Washington DC, USA, May 1993.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB Conference*, pages 487–499, Santiago, Chile, September 12–15 1994.

[3] Annotated groundtruth database. Department of Computer Science and Engineering, University of Washington, `http://www.cs.washington.edu/research/imagedatabase/groundtruth/`, 1999.

[4] B. Berendt and M. Spiliopoulou. Analysis of navigation behaviour in web sites integrating multiple information systems. *VLDB Journal: Special Issue on Databases and the Web*, 9(1):56–75, 2000.

[5] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations. In J. Peckham, editor, *Proceedings of the Annual International ACM SIGMOD Conference on Research and Development in Management of Data (SIGMOD'97)*, pages 255–264, Tuscon AR, USA, May 1997.

[6] I. J. Cox, M. L. Miller, S. M. Omohundro, and P. N. Yianilos. Target testing and the `PicHunter` Bayesian multimedia retrieval system. In *Advances in Digital Libraries (ADL'96)*, pages 66–75, Library of Congress, Washington, D. C., May 13–15 1996.

[7] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candicate generation. In W. Chen, J. F. Naughton, and P. A. Bernstein, editors, *Proceedings of the Annual International ACM SIGMOD Conference on Research and Development in Management of Data (SIGMOD2000)*, Dallas, TX, USA, May 2000.

[8] D. Harman. Overview of the first Text REtrieval Conference (TREC–1). In *Proceedings of the first Text REtrieval Conference (TREC–1)*, pages 1–20, Washington DC, USA, 1992.

[9] J. Hipp, U. Güntzer, and G. Nakhaeizadeh. Algorithms for association rule mining – a general survey and comparison. *SIGKDD Explorations*, 2(1):58–64, 2000.

[10] C. Jermain and R. J. Miller. Association mining without support thresholds. Technical report, Georgia Institute of Technology, 2001.

[11] A. Kohrs and B. Merialdo. Clustering for collaborative filtering applications. In *Proceedings of the International Conference on Computational Intelligence for Modelling Control and Automation*, pages 199–204, Vienna, Austria, February 1999. IOS Press.

[12] C. S. Lee, W.-Y. Ma, and H. Zhang. Information embedding based on user's relevance feedback in image retrieval. In Panchanathan et al. [23], pages 294–304. (SPIE Symposium on Voice, Video and Data Communications).

[13] B. Li, E. Chang, and C.-S. Li. Learning image query concepts via intelligent sampling. In *Proceedings of the second International Conference on Multimedia and Exposition (ICME'2001)*, pages 1168–1171, Tokyo, Japan, August 2001. IEEE Computer Society, IEEE Computer Society.

[14] M. Li, Z. Chen, L. Wenyin, and H.-J. Zhang. A statistical correlation model for image retrieval. In *Proceedings of the ACM Multimedia Workshop on Multimedia Information Retrieval (ACM MIR 2001)*, pages 42–45, Ottawa, Canada, October 2001. The Association for Computing Machinery.

[15] W. Y. Ma, Y. Deng, and B. S. Manjunath. Tools for texture- and color-based search of images. In B. E. Rogowitz and T. N. Pappas, editors, *Human Vision and Electronic Imaging II*, volume 3016 of *SPIE Proceedings*, pages 496–507, San Jose, CA, February 1997.

[16] H. Mannila and H. Toivonen. Discovering generalized episodes using minimal occurences. In *Proceedings of the 2nd Internationla Conference on Knowledge Discovery and Data Mining (SIGKDD1996)*, pages 146–151, Portland, OR, USA, August 1996.

[17] T. Minka. An image database browser that learns from user interaction. Master's thesis, MIT Media Laboratory, 20 Ames St., Cambridge, MA 02139, 1996.

[18] H. Müller, W. Müller, D. M. Squire, S. Marchand-Maillet, and T. Pun. Learning feature weights from user behavior in content–based image retrieval. In S. Simoff and O. Zaiane, editors, *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (Workshop on Multimedia Data Mining MDM/KDD2000)*, pages 67–72, Boston, MA, USA, August20-23 2000.

[19] H. Müller, W. Müller, D. M. Squire, S. Marchand-Maillet, and T. Pun. Strategies for positive and negative relevance feedback in image retrieval. In A. Sanfeliu, J. J. Villanueva, M. Vanrell, R. Alcézar, J.-O. Eklundh, and Y. Aloimonos, editors, *Proceedings of the 15th International Conference on Pattern Recognition (ICPR 2000)*, pages 1043–1046, Barcelona, Spain, September 2000. IEEE.

[20] H. Müller, W. Müller, D. M. Squire, S. Marchand-Maillet, and T. Pun. Performance evaluation in content–based image retrieval: Overview and proposals. *Pattern Recognition Letters*, 22(5):593–601, April 2001.

[21] W. Müller, Z. Pečenović, H. Müller, S. Marchand-Maillet, T. Pun, D. M. Squire, A. P. D. Vries, and C. Giess. MRML: An extensible communication protocol for interoperability and benchmarking of multimedia information retrieval systems. In *SPIE Photonics East - Voice, Video, and Data Communications*, pages 961–968, Boston, MA, USA, November 5–8 2000.

[22] W. Müller, D. M. Squire, H. Müller, and T. Pun. Hunting moving targets: an extension to Bayesian methods in multimedia databases. In Panchanathan et al. [23], pages 328–337. (SPIE Symposium on Voice, Video and Data Communications).

[23] S. Panchanathan, S.-F. Chang, and C.-C. J. Kuo, editors. *Multimedia Storage and Archiving Systems IV (VV02)*, volume 3846 of *SPIE Proceedings*, Boston, Massachusetts, USA, September 20–22 1999. (SPIE Symposium on Voice, Video and Data Communications).

[24] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool for interactive content–based image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5):644–655, September 1998. (Special Issue on Segmentation, Description, and Retrieval of Video Content).

[25] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[26] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–287, 1990.

[27] A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. In U. Dayal, P. M. D. Gray, and S. Nishio, editors, *Proceedings of the 22nd International Conference on Very Large Databases (VLDB'95)*, Zürich, Switzerland, September 1995.

[28] S. Shekhar and Y. Huang. Discovering spatial co–location patterns: A summary of results. In *Proceedings of the 7th International Symposium on Spatial and Temporal Databases*, Retondo Beach CA, USA, July 2001.

[29] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content–based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22 No 12:1349–1380, 2000.

[30] J. R. Smith. Image retrieval evaluation. In *IEEE Workshop on Content–based Access of Image and Video Libraries (CBAIVL'98)*, pages 112–113, Santa Barbara, CA, USA, June 21 1998.

[31] J. R. Smith and S.-F. Chang. VisualSEEk: a fully automated content–based image query system. In *The Fourth ACM International Multimedia Conference and Exhibition*, Boston, MA, USA, November 1996.

[32] D. M. Squire, W. Müller, H. Müller, and T. Pun. content–based query of image databases: inspirations from text retrieval. *Pattern Recognition Letters (Selected Papers from The 11th Scandinavian Conference on Image Analysis SCIA '99)*, 21(13-14):1193–1198, 2000. B.K. Ersboll, P. Johansen, Eds.

[33] N. Vasconcelos and A. Lippman. Learning over multiple temporal scales in image databases. In D. Vernon, editor, *6th European Conference on Computer Vision (ECCV2000)*, number 1842 in Lecture Notes in Computer Science, pages 33–47, Dublin, Ireland, June 26–30 2000. Springer–Verlag.

[34] E. M. Voorhees and D. Harmann. Overview of the seventh Text REtrieval Conference (TREC–7). In *The Seventh Text Retrieval Conference*, pages 1–23, Gaithersburg, MD, USA, November 1998.

[35] M. Worring, A. W. M. Smeulders, and S. Santini. Interaction in content–based image retrieval: An evaluation of the state of the art. In R. Laurini, editor, *Fourth International Conference On Visual Information Systems (VISUAL'2000)*, number 1929 in Lecture Notes in Computer Science, pages 26–36, Lyon, France, November 2000. Springer–Verlag.

[36] K.-L. Wu, P. S. Yu, and A. Ballman. Speedtracer: A web usage mining and analysis tool. *IBM Systems Journal on Internet Computing*, 37(1):89–105, 1998.

[37] M. J. Zaki, S. Parthgasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *Proceedings of the 3rd Internationla Conference on Knowledge Discovery and Data Mining (SIGKDD1997)*, Newport Beach, CA, USA, August 1997.