A scalable and efficient content-based multimedia retrieval system

Lioudmila Boldareva, Djoerd Hiemstra, Willem Jonker
Database Group, University of Twente, The Netherlands,
{*L.V.Boldareva, D.Hiemstra W.Jonker*}@*cs.utwente.nl*

Abstract

In this work the problem of content-based information retrieval is approached from a new perspective. We look at a probabilistic approach in CBIR from the angle of Bayesian networks. Our data structure serves to break two bottlenecks of retrieval performance: (1) high dimensionality of feature vectors and (2) poor mapping of raw features into high-level content that a human understands (the semantic gap). We use the network structure instead of the feature space, and propose updating the higher-level content description by utilising the relevance feedback obtained from the user. Strategies for display update for the next iteration are studied. A new approach for selecting the next display set is tied to our data structure.

1 Introduction

In content-based information retrieval, there is a problem of the gap between human perception of the document, which is often referred to as high-level content and its actual representation at the lowest level, in the data storage.

This problem is addressed by indexing documents in the collection. Text documents are indexed based on the words they contain; for images and video, pictorial features such as colours, shapes, textures, motion detection are used. Careful selection of the feature set allows capturing semantics of the documents, especially in limited domains, where the range of possible values is pre-determined. Examples of such limited domains are texture catalogues, medical image databases, video archives of known context. Often the number of features automatically extracted from raw data is large, with the hope that it helps to capture the semantics better. Modern retrieval systems have rich feature space: in MARS (Rui et al., 1997b) the vector space is of at least several dozens dimensions, in PicHunter (Cox et al., 2000) the number of features is about 128, Viper (Müller et al., 2000b) boasts $O(80\ 000)$ values. The objects are represented as points in metric space, with the dimensionality corresponding to the number of extracted features. The similarity between them is determined by means of an appropriate metric, such as Euclidean distance.

A lot of work is done to determine "the best feature set", i.e. such representation of multimedia data that would match the human perception of it. This is not a trivial task for images and video, since visual data carries a lot of information which is hard to decode automatically. However, both automatic and manual feature selection might still not solve the problem. The user is not always certain what pictorial characteristics are important for his/her target, and the semantics of an object may be ambiguous.

At the same time, due to the large number of dimensions, the pictorial features are subject to "the curse of dimensionality", when the performance of indices drops dramatically as the number of dimensions grows. Effective multidimensional indexing and approximate retrieval based on indexing are active research topics, as well as the problem of (weighted) nearest neighbour search in the indexed space (Faloutsos and Lin, 1995; Wu and Manjunath, 2001; de Vries et al., 2002). The situation turns tricky: on one hand we have large number of features that are hard to index; on the other hand, the importance of a certain feature is unknown.

A significant improvement of the performance of content-based retrieval systems can be achieved by using *relevance feedback*, a technique that allows the user to rate the (intermediary) search results. Further ranking and retrieval of documents in the collection is based on the feedback received from the user. In the domain of image retrieval, where the semantic gap is especially large, relevance feedback is often used not only for ranking the output documents, but also for fine-tuning the whole system, adapting such parameters as similarity function (Rui et al., 1997a; Wu and Manjunath, 2001; Aksoy and Haralick, 2000; Geman and Moquet, 1999), and/or the feature set that is used. In the 2-layer retrieval model in MARS (Rui et al., 1997a), the useful feature subset is determined based on the user response, when features are examined across iterations and most distinguishing ones get more weight in subsequent iterations. In the Qbic image retrieval system

(Flickner et al., 1995) the user can manually emphasize the importance of a certain primitive feature by using "control knobs". At a higher level of representation, dealing with identified colour-texture-shape objects, or "Blobs" in Blobworld (Carson et al., 1999), the user can explicitly point the region of interest, modelled by pre-computed Gaussian mixtures, that incorporate colour and texture information.

In the present paper we approach the problem of content-based image retrieval and indexing from another, new perspective. We look at a *probabilistic approach* to document indexing and retrieval, from the angle of Bayesian networks. Our data structure serves to break two bottlenecks of content-based multimedia retrieval performance: (1) high dimensionality of feature vectors, preventing efficient indexing and (2) ineffective mapping of raw features into higher-level concepts reflecting the actual content. We propose using the network structure for the data instead of multidimensional feature space. The network encodes higher-level context and makes use of relevance feedback. Primitive pictorial features are not addressed at run-time to determine the similarity of objects, but instead, a probabilistic method is used at indexing time to construct the meta-data. We will also study various approaches to retrieval with relevance feedback in the light of our data structure.

The rest of the paper is organised as follows: In section 2 we briefly describe our novel approach, section 3 discusses some issues that arise in the retrieval process, the implementation of the system is described in section 4. Finally, sections 5-6 report preliminary experimental results and future research directions.

2 Bayesian retrieval framework

2.1 Data organization

Consider a collection $\Im$ of objects i among which there is an object that the user is looking for — the target T. In the search session the user retrieves a set of candidate objects on the screen and feeds back to the system his/her opinion about their relevance to the target. Each object might look like the target the user has in mind, and then it is *selected* by the user, or it is *de-selected*, if it doesn't resemble the target. For the selected candidate object i we denote the event as $(\delta_i = 1)$, and for de-selected ones as $(\delta_i = 0)$. The feedback obtained from the user allows the system to make some inference and compile a new display set of n elements, *the display set*, to show in the next iteration. There may be several rounds of feedback during one search session.

To perform comparison of relevant and non-relevant objects, it is necessary to organize the collection by introducing relations between the objects. As mentioned in the introduction, the relations are often defined by a vector space derived from primitive features and a corresponding similarity function(s). In our work we suggest that the user feedback can be used to build up and update the objects relations.

As in (Maron and Kuhns, 1960), we introduce a "measure of closeness" of an object i to an object j as a conditional probability and denote it as $P(\delta_i|T = j)$.

Def. 1 $P(\delta_i|T = j)$ is the probability of an object i being selected by the user given that another object j is the target of the search.

In other words, the user's judgement about the relevance of objects is a necessary component of our system. It is reasonable to assume that $P(\delta_i|T = i) \equiv 1$, i.e. the user always identifies the target as relevant. We also put a constraint that the target exists in the collection and is unique: $P(T = j|T = i) \equiv 0$, $i \neq j$, $\sum_{i \in \Im} P(T = i) = 1$.

As an example take a user looking for an image of a cherry tree. He/she may find an image of an apple tree relevant to his/her query, and the measure of this relevance is denoted as $P(\delta_{apple\,tree}|T = cherry\,tree)$. An image of a cherry fruit may be found relevant too, having another value of $P(\delta_{cherry}|T = cherry\,tree)$.

Each $P(\delta_i|T = j)$ can be seen as a *weighted arc*, or an oriented path of weight P from j to i that is traversed during the search session, utilising the user's feedback. The graph containing these arc values can be seen as a map of oriented paths between elements in the collection. We call its matrix representation *topographic*. The topographic matix need not be symmetric. Imagine a road map, where two cities are connected by roads with different number of lanes going in each direction. It is important to notice, that we deliberately do not construct the complete graph, with all existing connections between the nodes, but choose only the most significant ones.

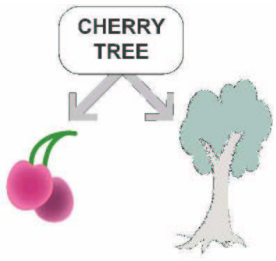

Figure 1: Graphic representation of cherry tree example

Looking at the structure of the graph, each element in the collection can be *described* by a number of other elements pointed by it, which, in

turn, are pointed by third elements. The cherry tree in our small example is described by the set $\{apple\,tree,\ cherry\}$, which drives at a tree with a cherry – the cherry tree (see Fig. 1). These associations that come from users judgements and refer to the hidden semantics of objects, serve as meta-data for the collection. The collection describes itself by means of meaningful relations observed in earlier retrieval sessions.

To initiate the system, these relations are calculated as conditional probabilities $P(\delta_i|T=j)$ based on low-level primitive features, which are silently present in the system, but are addressed only once. The subsequent successful searches are used to accumulate the knowledge about objects relations and update the arc weights in the topographic matrix. In this way, instead of chasing "the right feature set" we leave this task to the users, believing that vox populi will give us this best feature set enclosed in a black box.

With the graph representation that refers to a multimedia object as a whole, when a primitive stand-alone feature does not explicitly play an important role, the nodes in the topographic matrix need not be images only. Other types of media, such as video, audio or speech transcripts can be plugged in as separate nodes in the graph. Note however, that integrating other types of media is not trivial. Our data structure relies on multiple feedback iterations. Dynamic media such as video or audio may not stand many feedback loops, because assessing a video clip or a music fragment requires from the user more efforts and time compared to still images. Nevertheless, such nodes may be potential targets or, conversely, the starting points in a search session. Textual nodes are of particular interest for a retrieval system, since querying in the form of text is very convenient for the user.

2.2 Retrieval during the search session

We assume that the user is consistent in his/her judgements, does not forget what the target is, and that the target object is unique and exists in the collection. The assumption of uniqueness is valid with queries like "find me an image of a Golden Retriever puppy". Queries like "find me all pictures of Britney Spears" are not handled by the model directly. However, there is a way to retrieve ranked lists of "most relevant" objects which may be considered targets. In our framework we use the following definition of the target:

Def. 2 *The target* as an object, after retrieval of which the user terminates the search successfully.

The goal of a retrieval system is to help the user find the target object (and possibly all similar objects) after few iterations, with a reasonably small amount of time spent on each round.

Probabilistic methods in information retrieval were initially used for text collections (Maron and Kuhns, 1960; Robertson, 1977; Hiemstra, 2000) and later the ideas were adapted to image retrieval (Vasconcelos and Lippman, 2000). In content-based retrieval systems the user's information need is unknown and should be guessed. In general, retrieval with the use of relevance feedback can be formulated as follows:

> In the current data structure, having observed the user judgements in the search process, what is the object that the user wants to find?

We construct the answer (that is, predict the user's target object) using Bayes' rule. Then the problem is reformulated as estimating the user's action of selecting/deselecting relevant objects, given the target that he/she has in mind:

$$P(T=i,U|\delta^1_{(\cdot)},\ldots,\delta^n_{(\cdot)}) = \frac{P(\delta^1_{(\cdot)},\ldots,\delta^n_{(\cdot)}|T=i)P(T=i|U)P(U)}{P(\delta^1_{(\cdot)},\ldots,\delta^n_{(\cdot)})} \quad (1)$$

where U denotes the current user. Since we assume that the state of the (unknown) user variable does not change during one search session, and U affects $\delta_{(\cdot)}$ through T, we may omit the user notation in further formulae, to keep the notation short (Gelman et al., 1995, Chapter 5). The upper index in $\delta^1_{(\cdot)}\ldots\delta^n_{(\cdot)}$ denotes n displayed objects, either selected by the user ($\delta^i_{(\cdot)}=1$) or not ($\delta^i_{(\cdot)}=0$); $P(T=i)$ is the probability that the object i is the target, and $P(\delta^k|T=i)$ is the probability of a k-the object on the screen to be selected by the user given that i is his/her target.

We distinguish between the objects that have been displayed to the user $\delta^{1\cdots n}_{(\cdot)}$ and the rest of the collection $\{\Im\}$. The index of the element displayed on the screen determines uniquely an element from the collection, and further we omit the subscript, if the upper index is used. Note that equation (1) is regarded as recursive, i.e. the posterior probability of being the target determined at step s as $P(T=i|\delta^1,\ldots,\delta^n)$ serves as the prior $P(T=i)$ at the next iteration $s+1$ (Gelman et al., 1995, Chapter 2).

In this way, in each round the observed user response is used to calculate probability $P(T=i)$. In the beginning, before any information from the user is received, each object has a certain prior probability to be the target[1]. The possible output of in-

[1] Often equal prior probabilities are assigned to all elements of the collection. The importance of selecting the "good" priors is studied in, e.g. (Kraaij et al., 2002).

corporated primary textual query or previous search sessions results may be used to define the prior value of $P(T = i)$ more accurately. Recall that by definition 2, $P(T = i)$ is the probability that the search will be completed successfully immediately after object i is shown to the user.

The meaning of the first term in the numerator of equation (1) is explained by the following definition:

Def. 3 $P(\delta^1, \ldots, \delta^n | T = i)$ is the probability that the user marks the displayed set of objects in a certain way, given that i is the target for him/her.

To determine this joint conditional probability we assume (for the time being) that given the target, the user picks each of n candidates independently of other objects present on the screen. This assumption is similar to term independence assumption used in text retrieval. Then equation (1) becomes

$$P(T = i|\delta^1, \ldots, \delta^n) = \frac{\prod_{s=1}^{n} P(\delta^s|T = i)P(T = i)}{P(\delta^1, \ldots, \delta^n)}. \quad (2)$$

The denominator serves as normalizing factor, and, for the purpose of ranking, it can be replaced with a constant.

2.3 Retrieval in terms of Bayesian network

The search algorithm together with the topographic matrix can be graphically represented as a Bayesian Network (Fig. 2). The conditional probabilities $P(\delta_i|T = j)$ indicate the influence (or *casual impact*) of the fact that a user regards a certain element i of the collection as "somewhat related" to the target j in a search session. The events observed during an iteration are the values of random variables of the network. These values, or *states* are as follows: the node representing a user U looking for an object T which can be any single element of the collection ($T \ni \{1, 2, \ldots, n\}$), and binary variables $\delta_i, i \in \Im$, representing the elements that are judged by the user. They may be marked as similar to the target T, ($\delta_i = 1$), or not ($\delta_i = 0$). Each state is associated with a certainty.

For each user U we may introduce a different prior $P(T|U)$. Every element, and only one in the collection might be the target for a certain user in a given search session. Thus the target can be identified by the maximum probability of $P(T = j|U)$. In our network we talk about *the user model*, since the concept of "target object" has meaning only with respect to a particular user who wants to find the object. However, the user him-/herself is not part of the data structure, and needs to be separated from it.

When the user selects a displayed object as relevant to his/her target ($\delta_2 = 1$ in Fig. 2), he/she is

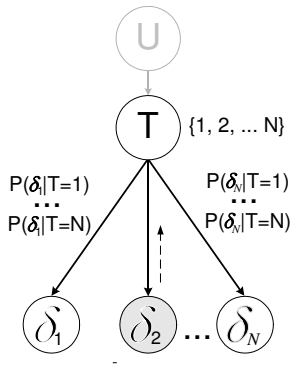

Figure 2: Bayesian Network representation.

doing reasoning in the direction opposite to the casual arrows (the dashed line). Thus the certainty of T changes, which in its turn creates new certainties of not-yet displayed elements δ_1 up to δ_N. Thus, when nothing is known about the state of T and evidence is received at δ_2, nodes $\delta_1 \ldots \delta_N$ are dependent, which means that information on either event affects the certainty of the other, in accordance with the tables attached to the casual arrows. These tables are in fact columns from the topographic matrix. However, when the state of T is known for certain, then its children are independent: information on one has no effect on another. When the retrieval system is at work we repeatedly get new cases, and we learn from these cases. It is a general practice in retrieval systems to discard the results of this learning after the search is concluded, since the user node remains unknown. Because the system is re-initiated for every new query, this method is called *short-term learning*.

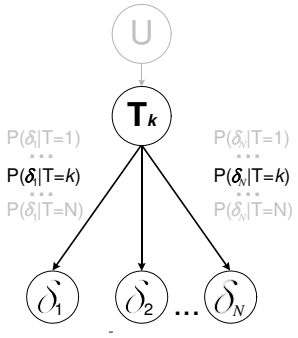

Figure 3: The network structure after the target is identified. The values from the topographic matrix subject to update are shown in black

After T is initiated, i.e. the target of the search is identified, some conditional probabilities, namely, the rows of the topographic matrix corresponding to the target object can be updated (Fig. 3). The

information obtained from a given retrieval session is used for *long-term learning*. The purpose of the update is to increase the conditional probability to be selected by the user of all objects that the user indeed selected. At the same time the connections to the objects that were marked by the user as non-relevant, may be punished.

In retrieval systems that try to learn from the user interaction, the following assumption is made explicitly or implicitly (Müller et al., 2000a; Ishikawa et al., 1998):

> The documents that the user marks as "relevant" are *similar to each other* with respect to a (hidden) feature. The documents that are discarded by the user do not necessarily have a common feature.

One should be careful about grouping the positively marked objects into a class of neighbours. The user who selects relevant objects, compares them to the target he/she is looking for, and not to each other!

In the future we would like to receive some evidence about the user model, which may affect the update strategy, and the prior distribution. However, a simple assumption about the user who wants to find the target and responses consistently, can serve as a generic user.

3 Display update schema

After the feedback is received from the user, $P(T = i|\delta^1, \ldots, \delta^n)$ is calculated according to the Bayes' rule. Then the new evidence should be received from the information variables δ_j that form the display set to present to the user in the next round.

3.1 What is *display update?*

The display update is an important part of the search process, since the speed and quality of the search depends on it. Each iteration should bring us closer to the target. "Closer to the target" may have various interpretations, such as *(a)* the posterior probability $P(T = i)$ of the element is increasing, or *(b)* the target element approaches the top of the ranked list or even *(c)* the expected number of remaining iterations decreases. Recall that the goal is not only to identify the target, but to do it in few iterations in a limited amount of time. It is however not clear which strategy is optimal with respect to both iteration number and calculation costs, as one can see below.

As an illustration to the various display schema, consider the following story:

> A person wanted to find a friend's house in a big city and got lost. The situation is as follows: the friend has a map of the area for pedestrians, the Lost person has a satellite mobile phone. The Lost can call her friend as often as she likes, paying $10 per minute, to ask for the way. The friend at home looks up her position on the map and gives to the Lost one directions.

The approach that we call *naïve* suggests that the friend at home directs the lost friend along the shortest way, but at each crossroads the latter has to make a costly call and ask where she should go next: straight, left, right or maybe back.

We can also advise the friend at home to look up the crossroads where there might be road signs on the way (say, major crossroads) and lead the Lost one towards them, so that from time to time she could see where to go further without calling, or walk in the direction of "through traffic" road sign. This will be called best-selected approach, discussed in Section (3.4). The *most informative* schema, which is also discussed briefly below, is to send the friend to the highway, where all the road signs are present (but it takes *a lot* of walking!).

3.2 Best target approach (*naïve*)

The probability that the target is an element i, $P(T = i)$, can be considered as the score that the element receives during the session. In the naïve model we show to the user the most-probable objects, hoping that the target is among them. This is a simple application of a standard technique used in information retrieval under the name *probability ranking principle* (Robertson, 1977). We note that the denominator in equation (2) does not depend on a particular element we want to rank, and can be replaced with a value that is a constant given object i, since this will not affect the ranking order. Thus, we replace the denominator, noting that in general $P(\delta^1, \ldots, \delta^n) \neq \prod_{s=1}^{n} P(\delta^s)$:

$$P(T = i|\delta^1, \ldots, \delta^n) \propto \frac{\prod_{s=1}^{n} P(\delta^s|T = i)P(T = i)}{\prod_{s=1}^{n} P(\delta^s)}. \quad (3)$$

It is interesting to note that by assuming that each object is selected/de-selected independently of others, we can treat the display set as a series of single objects shown one after another, where the order becomes unimportant.

Any monotonic transformation of the ranking function (3) will produce the same ordering of the objects. Instead of using the product of weights, the formula can be implemented by using the sum of logarithmic weights. All objects that have the ratio $P(\delta^s|T = i)/P(\delta^s)$ close to 1 give almost zero contribution to the score ($\log(1) = 0$). Their probability to be selected by the user does not depend on the user's target.

The obtained values are used to select n objects for the next display set. These are the objects that

have the highest score and have not been shown to the user yet. Strictly speaking, the objects that have been shown to the user have zero probability to be the target, since for them it is known for sure that they are not the target.

The probability ranking principle fits well in the framework of text retrieval, since texts require some time for reading and relevance judgment. Therefore only few feedback iterations are possible. When visual information is presented on the screen, a quick glance is sufficient to evaluate the results, and a new round of feedback can be started at once. Hence the strategy of the display update can be changed, allowing the user to evaluate as large data regions as possible, before the ranked list of results is produced.

3.3 Best information approach (costly)

We may choose the next display set in such a way, that it would maximally reduce the uncertainty of the system, based on the expected amount of on information that could be obtained (Cox et al., 2000; Zhang and Chen, 2002). Such an approach is used in machine learning and it is considered optimal with respect to the number of iterations. We use information theory (Guiasu, 1977) to find the optimal display set for the next iteration. The information I obtained from a display set $(\delta^1, \ldots, \delta^n)$ is

$$I(\delta^1, \ldots, \delta^n) = -\sum_{\mathcal{S}} P(\delta^1, \ldots, \delta^n)\ \log_2 P(\delta^1..\delta^n), \tag{4}$$

where

$$P(\delta^1, \ldots, \delta^n) = \sum_{i \in \Im} P(T = i) \prod_{s=1}^{n} P(\delta^s | T = i).$$

$\mathcal{S}$ is a set of all possible combinations of selected and de-selected objects in the display set $(\delta^1, \ldots, \delta^n)$ that may occur in the following iteration. Here as in (2) we assume that the user selects each object independently of other objects presented on the screen. Note that the denominator in equation (2) is the probability of only one possible feedback on the display set. There are 2^n possible ways of selecting/deselecting n objects. As one can see, the information-based display update schema is costly. Straightforward scanning the collection and selecting all possible combinations of n objects, and taking the best subset, is exponential with respect to the collection size. In PicHunter (Cox et al., 2000), Monte Carlo sampling was used to find a sub-optimal solution. Thus, although the best-information approach may be optimal with respect to the number of iterations, it is far from optimal when speaking about the computation intensity and total time spent on the search. However, the implication that information theory gives is the following: to reduce the number of communication rounds, each cycle should contain as much of (unknown) information as possible, and one should bear in mind that deviation from the optimal strategy may lead to the decrease of information gain and therefore result in declined quality of the search.

3.4 Best selected object approach (intuitive)

As we noticed, to minimize the number of iterations when selecting the new display set, we want to show such objects to the user that would result into best information about the "location" of the target in the database. If we take the case of text retrieval, in some situations the target can be, for instance, a list of articles about the subject. Such articles retrieved as the result set are likely to have the probability to be the target relatively high compared to other documents. However it is more likely that the user will prefer a document containing a (complete) list of references along with some most relevant documents from that list to several relevant documents, that are very similar to each other, but do not cover all the subject.

Intuitively, a group of similar elements in the collection is described best not with the element that only has the highest score $P(T = i)$, but by the one that, in the topographic matrix, is pointed by the largest number of elements that have high probability to be the target. It is easy to show that the maximum information can be expected from the user response to one displayed object when the probability of the object to be selected is exactly $1/2$. In this case the uncertainty about the following user action is largest. Let us have a closer look at $P(\delta_i)$. As mentioned in Section (2.1), this term denotes the probability of i being selected by the user in the current iteration, despite the target. We can evaluate it as follows:

$$\begin{aligned} P(\delta_i) = P(\delta_i | T = j) P(T = j) + \\ P(\delta_i | T = j) P(T \neq j). \end{aligned} \tag{5}$$

The probability of a single object *not* to be the target $P(T \neq j)$ is unknown, but, assuming that the target exists in the collection and is unique, we note that

$$P(i \neq T) = P\left(\bigcup_{\substack{j \in \mathcal{U} \\ j \neq i}} (T = j) \right), \tag{6}$$

i.e. if i is not the target then some other element in the collection can be the target.

From (5) and (6), the probability of an object to

be selected can be rewritten as

$$P(\delta_i) = P(T=i) + \left(\sum_{\substack{j \in \mathcal{U}, \\ j \neq i}} P(\delta_i | T=j) P(T=j) \right), \tag{7}$$

where the sum is greater than or equal to zero. If the candidate object is selected based on $P(T=i)$, then one has to specify what is the optimal value of this probability or what is the selection criteria. Note that the "largest value of $P(T=i)$ criterion" gives in most cases an arbitrary value of $P(\delta_i)$ and, thus, not optimal from the information gain point of view[2]. The drop in information gain may ultimately lead to unnecessary search iterations.

A similar statement holds for the display set consisting of more than one element. The optimal, with respect to information, selection consists of independent objects with equal probability for every outcome (see equation (4)). In other words, the optimal selection is such that for the display set under consideration $P(\delta^1, ..., \delta^n) = 1/2^n$ for every possible combination of candidate objects. This consideration is especially important in the begin phase, when all probabilities $P(T=i)$ are not quite distinguishing. Then the maximum information criteria can be, without too big loss of information, reduced to $\tilde{I} = max\left[P(\delta^1, ..., \delta^n)\right]$, which in turn can be approximated by maximum of the product of the corresponding $P(\delta^s)$. This maximum is delivered by the first n elements with the largest value of $P(\delta)$. Every selection based on $P(T=i)$ will be different from those based on $P(\delta_i)$ and thus systematically further from the approximation based on "best $P(\delta_i)$" selection.

This update strategy, however, has its limitations. Computation of $P(\delta_i)$ according to (7) requires scanning all the topographic matrix for each element. For a collection of useful size this will inevitably become a bottleneck. Our approach to the data organisation, when the topographic matrix contains quite many "holes", overcomes this problem. More detail about the construction of the initial topographic matrix is given in the next section.

4 Implementation

4.1 Topographic matrix

The topographic matrix containing conditional probabilities plays an important role in our retrieval model. We performed feature extraction and normalisation, to obtain $N \times N$ kick-off values for the collection of N elements. We store only a fraction of them, assuming that for a large number of objects the following holds:

$$P(\delta^s | T=i) = P(\delta^s), \tag{8}$$

i.e. the fact that the target is i does not affect the probability for s to be selected by the user.

By dropping useless connections between the elements that have weak or no influence on each other, the waste of time and disk space is avoided. In addition, we preserved not only the most similar objects, but also those that are the furthest from each other, assuming that they have negative impact, i.e. i is unlikely to be marked by the user as resembling the target when j is the target: $P(\delta^i | T=j) \to 0$. The question is, what are the assumed, or *default*, values of the dropouts? It is interesting to notice that in (8) the elements are the ones that do not affect the score in the naïve display update scheme. The missing conditional probabilities are simply ignored when updating the object scores according to equation (3). These scores, strictly speaking, are not probabilities, although they are derived with the help of probability theory. Setting $P(\delta^s | T=i)$ to zero would inevitably turn the whole equation (2) into zero, since we certainly leave most of the connections out. In similar situation in content-based text retrieval, (Hiemstra, 2000) used linear interpolation, introducing the notation of *importance* for the query term. We will simply use equation (8) to obtain the default value, retreating to $P(\delta^s)$ as a substitute for the missing $P(\delta^s | T=i)$. This is referred to as a "back-off model".

4.2 Available TREC data

As the data set we used the test set of TREC-2002 video collection, containing $N = 8869$ key frames extracted automatically from video data. To initiate the topographic matrix we used colour-based features, similar to those described in (Stricker and Orengo, 1995). For each image we took three central moments in HSV colour space: average value, variance and skewness in combination with weighted L_1 (Manhattan) distance. This compact feature set has quite good discriminating ability. The values of pair-wise distances varied from 0 to 7 and needed to be brought to probability range of $[0, 1]$. Simple division by the largest value (or the sum of all values) may distort the real distances, since an outlying value of 7 would cause squeezing the rest of similarities, most of which lied in $[1, 3]$, into a small interval, making the features undistinguishing. Usually as an approximation the "3-sigma" rule is applied (see e.g. (Rui et al., 1997b; Su et al., 2001)), because the features are processed online when the timing is constrained. Since we address the feature space offline, we could afford using the "full" normalisation that consisted of the following steps:

[2]Generally speaking, $P(\delta_i)$ is always greater than the corresponding $P(T=i)$, with two exceptions: when i is the target itself, and then $P(\delta_i) = P(T=i) \equiv 1$. This case is uninteresting in our framework.

1. Calculate pair-wise distances in the HSV metric feature space. Make sure that the pair-wise distance values have a distribution resembling the Gaussian.
2. Assuming normal distribution of pair-wise distances, for each pair of objects calculate the probability of their metric distance to be smaller or equal than its value using a cumulative normal distribution. This operation will re-scale the similarities with respect to their frequency of occurrence.
3. Subtract the obtained probability value from 1. This inversion assigns the largest probability for the objects that have the smallest distance in the selected feature space.

We plotted histograms of pair-wise distances in the selected feature space (Fig. 4) and found that logarithms of the distances are close to the shape of normal curve[3]. The log-igification of the similarities gives more distinguishing scale to the elements that have small distance from each other, which is a desired property, since we are more interested in differentiating between elements that affect each other rather than between those that seem independent.

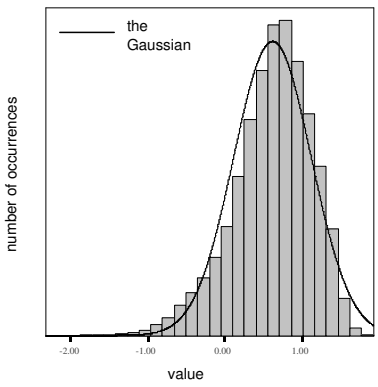

Figure 4: Example histogram of log-ifyed pair-wise distances of the collection, random sample of 3%.

The normalization is computationally quite intensive to perform for the whole dataset, since the major part of it would be discarded. We calculated iteratively the mean and standard deviation, and for each pair of images checked the metric distance before calculating the probability. Finally, we added neighbours for those elements that had too few close connections, disregarding the threshold. In this way we obtained an initial topographic matrix which can be updated based on the successful search sessions.

[3]Colour histograms and L_0 distance (*histogram intersection*) yielded similar distribution, with somewhat less steep right tail.

The optimal threshold, or the cut-off value for the dataset, is a topic for further investigation. The question is, how many connections from the topographic matrix can be sacrificed to performance so that the search quality would not drop dramatically. If we use a 9-element display set and leave 10% of connections, then, in the case when the elements on the screen are representing 9 non-overlapping regions from the collection, we have 90% of the collection affected by the user feedback. We realise that such a display update is unfeasible unless the user's information need is known. The upper bounds for the cut-off value is 1.5 of standard deviation for *neighbours* and 1.8 σ for *anti-neighbours*, which gives roughly 10% of the data. In any case we stored at least 1% of the elements as closest neighbours.

Ideally, if we leave only random $\bar{n} \approx 1.1\%$ connections, then in the case of display set containing 2 elements the expected number of objects that are connected to them would approximately be $8869 \cdot (0.011 * 0.011) = 1.073149$,. i.e. at lest one element. Since the remaining connections are not (completely) random, some meaningful results could already be expected with even smaller $\bar{n}$. However, when the display set contains $n > 2$ objects, it is unlikely that there will be an object connected to all n candidates, but the expected number of images connected to at least 2 of them is C_n^2, and a new display set can be selected among them.

4.3 "Toy" collection

In addition to the real dataset we used generated data containing geometric shapes: triangles, squares and circles of different colours, in various combinations. Some images were duplicated, and hand-drawn objects were added as noise. Part of the collection was duplicated once more and the colours were inversed. We used three feature sets with this collection: one addressed colours, and thus ignored the geometric shapes. Second contained a higher-level feature set, taking into account only type and the number of each kind of geometric shape, regardless the colour. Finally, the third feature representation was the combination of shape-based and colour-based features with weights, respectively, 0.2 and 0.8. The mapping of distances to probabilities was performed in the same way as in the TREC collection.

4.4 User interface

Before starting the search, the user is given a short introduction and instructions, how to give the feedback. The target image is then picked randomly from the collection. The user should find the target image, which for convenience is always present on the screen. The display set for the TREC data collection consisted of 9 images, and the feedback was ternary: the user was allowed to mark whether the

image was "good" or "bad", or set the pointer into neutral position ("don't know", default value). For the toy collection we showed only 4 images, because the collection size is small.

We started testing with the naïve update schema. The user was supposed to do one of the tree actions: (1) Select suitable images and press "Feedback" or (2) Select the target image if it is on the screen and press "Found" to retrieve the target and the ranked list of neighbours, or (3) Press 'Cancel" to terminate the unfinished search and get the best ranking images on the screen anyway. Only action (2) was considered as a successful search. All user actions are logged, and the distribution of $P(T = i)$ may be reconstructed for each moment of the search. In the best-selected display update, the user may only see the current ranked list by pressing "Found" and, if necessary, continue search by giving the feedback. The target is then indicated by the user as an extra step.

5 First conclusions

There are some conclusions that can be drawn already at the initial stage of the experiments:

1. Colour-based features alone are confusing for the images with contrasting colours. These features are far to low level, and large relevance feedback statistics should be collected before the topographic matrix is updated to the semantic level.
2. The naïve and best-selected display update schema both tend to select objects that are very similar to each other. This is especially obvious in the toy-collection, where some part of the data is identical to another part, with respect to geometric feature space. We expect that this effect would be decreased for the best-selected scheme by introducing a penalty function $\mathcal{P}$, which would improve approximation to the display update made in section 3.4.

   Since in the naïve scheme, the best targets in the current iteration are displayed to the user in the next round, the updated display consists of the closest neighbours of the good examples selected by the user, and, respectively, the furthest objects from the bad examples, and in this form mimics k-nearest neighbour retrieval technique. No wonder that k nearest neighbours are likely to be nearest neighbours of each other!
3. When binary feedback is used, the user's inconsistency is especially harmful, since images from the same class occasionally get different marks (one is selected, the other one is not). As mentioned in (Müller et al., 2000a), too much of negative feedback may damage the search session. However, the use of the "neutral" feedback button in the interface reduces the quality of the search, since it wastes screen space (and the user's attention). Alternatively, overwhelming influence of negative feedback can be eliminated by more careful selection of the display set and reducing (or even eliminating) the amount of anti-neighbours in the topographic matrix.

6 Future work

In the experiments we plan to clarify the following points:

1. What is the optimal strategy for the display update with respect to our data structure?
2. What is the optimal amount of connections in the topological matrix?
3. How does "neutral" feedback affect the search quality; What effect is brought by the presence of anti-neighbours in the database?
4. How to integrate transcripts of speech recognition of the video material into the system?

We need to perform a number of experiments, clarifying the questions listed above. In the experimental phase the target is always picked up from the dataset. To perform automatic simulation, we will use TREC-2002 topics as possible targets. The result sets provided by the TREC as answers to each query, will indicate positive feedback from the user. In this way we are able to clarify some research questions. The advantage of the automated system is that the exact same relevance judgement is done for different setups, and the same target may be retrieved many times by different versions of the system.

We plan two types of search: unbounded and limited. In the unbounded version the user is free to search as long as needed. In the limited conditions the user is allowed to make only a certain number of iterations, and then the resulting ranked list is studied. The checkpoints for the results are determined after 10 ($\log_2(8869/9) = 9.945$, expected number of rounds in case of optimal strategy) and 20 iterations.

We determine the quality of the search by the rank of the target object achieved at the checkpoints, and the highest rank achieved by the target element. In the unbounded version, the criteria are the number of iterations before seeing the target and the number of successful searches with a given number of iterations. Precision-recall graphs, a traditional method of evaluation the search quality, have a somewhat different meaning when the assumption that the target is unique and exists in the collection is made. Nevertheless, the subjective manual judgement of relevance of the result set is feasible. To evaluate any of the above strategies, we will also look at the

convergence of the rank of the target object during the search session.

7 Acknowledgement

The authors acknowledge *KPN Research, The Netherlands*, for financial support.

References

S. Aksoy and R. M. Haralick. 2000. Probabilistic vs. geometric similarity measure for image retrieval. In *IEEE Conf. Computer Vision and Pattern Recognition*, 06.

C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. 1999. Blobworld: A system for region-based image indexing and retrieval. In *Third International Conference on Visual Information Systems*. Springer.

I. J. Cox, M. L. Miller, T. P. Minka, T. V. Papathomas, and P. N. Yianilos. 2000. The bayesian image retrieval system, pichunter: Theory, implementation, and psychophysical experiments. *IEEE Tran. On Image Processing*, 9(1):20–37.

A. P. de Vries, N. Nes N. Mamoulis, and M. L. Kersten. 2002. Efficient k-NN search on vertically decomposed data. In *ACM SIGMOD International Conference on Management of Data*, Madison, WI, USA, June.

C. Faloutsos and K.-I. Lin. 1995. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In Michael J. Carey and Donovan A. Schneider, editors, *Proc. of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, San Jose, California, 22–25.

M. Flickner, H. S. Sawhney, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. 1995. Query by image content, the QBIC system. *IEEE Computer*, 28(9):23–31.

A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. 1995. *Bayesian Data Analysis*. Chapman & Hall.

D. Geman and R. Moquet. 1999. A stochastic feedback model for image retrieval. Technical report, Ecole Polytechnique, 91128 Palaiseau Cedex, France.

S. Guiasu. 1977. *Information theory with applications*. New York: McGraw-Hill.

D. Hiemstra. 2000. A probabilistic justification for using tf.idf term weighting in information retrieval. *International Journal on Digital Libraries*, 3(2):131–139.

Y. Ishikawa, R. Subramanya, and C. Faloutsos. 1998. MindReader: Querying databases through multiple examples. In *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, pages 218–227, 24–27 .

W. Kraaij, T. Westerveld, and D. Hiemstra. 2002. The importance of prior probabilities for entry page search. In *Proc. of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 27–34. ACM Press.

M. Maron and J. Kuhns. 1960. On relevance, probabilistic indexing, and information retrieval. *Journal of the ACM 7*, pages 216–244.

H. Müller, W. Müller, S. Marchand-Maillet, T. Pun, and et al. 2000a. Strategies for positive and negative relevance feedback in image retrieval. In *Proc. of International Conference on Pattern Recognition (ICPR'2000)*, Barselona, Spain, September.

H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and T. Pun. 2000b. Long-term learning from user behavior in content-based image retrieval. Technical Report 00.04, Computer Vision Group, Computing Centre, University of Geneva, rue Gnral Dufour, 24, CH-1211 Genve, Switzerland, March.

S. E. Robertson. 1977. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304.

Y. Rui, T. Huang, S. Mehrotra, and M. Ortega. 1997a. Automatic matching tool selection using relevance feedback in MARS. In *Int. Conf. on Visual Information Retrieval*.

Y. Rui, T. Huang, S. Mehrotra, and M. Ortega. 1997b. A relevance feedback architecture in content-based multimedia information retrieval systems. In *Proc. of IEEE Workshop on Content-based Access of Image and Video Libraries, in conjunction with IEEE CVPR*.

M. A. Stricker and M. Orengo. 1995. Similarity of color images. In *Proc. of the Storage and Retrieval for Image and Video Databases III*, pages 381–392, San Diego/La Jolla, California, USA, February.

Z. Su, S. Li, and H. Zhang. 2001. Extraction of feature subspaces for content-based retrieval using relevance feedback. In *ACM Multimedia*, pages 98–106.

N. M. Vasconcelos and A. B. Lippman. 2000. A probabilistic architecture for content-based image retrieval. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*.

P. Wu and B. S. Manjunath. 2001. Adaptive nearest neighbor search for relevance feedback in large image databases. In *Proc. of ACM International Multimedia Conference*, Ottawa, Canada, October.

C. Zhang and T. Chen. 2002. An active learning framework for content based information retrieval. Technical Report AMP 01-04, Advanced Multimedia Processing Lab, Electrical and Computer Engineering Carnegie Mellon University Pittsburgh, PA 15213, March.