Computer Vision and Image Understanding 116 (2012) 510–523

Contents lists available at SciVerse ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Incorporating multiple distance spaces in optimum-path forest classification to improve feedback-based learning ☆

André Tavares da Silva [a,*], Jefersson Alex dos Santos [b], Alexandre Xavier Falcão [b], Ricardo da S. Torres [b], Léo Pini Magalhães [a]

[a] Department of Computer Engineering and Industrial Automation, School of Electrical and Computer Engineering, University of Campinas (Unicamp), 400 Albert Einstein Avenue, 13083-852 Campinas, SP, Brazil
[b] Institute of Computing, University of Campinas (Unicamp), 1251 Albert Einstein Avenue, 13083-852 Campinas, SP, Brazil

ARTICLE INFO

Article history:
Received 21 December 2010
Accepted 6 December 2011
Available online 14 December 2011

Keywords:
Content-based image retrieval
Optimum-path forest classifiers
Composite descriptor
Genetic programming
Multi-scale parameter search
Image pattern analysis

ABSTRACT

In content-based image retrieval (CBIR) using feedback-based learning, the user marks the relevance of returned images and the system learns how to return more relevant images in a next iteration. In this learning process, image comparison may be based on distinct distance spaces due to multiple visual content representations. This work improves the retrieval process by incorporating multiple distance spaces in a recent method based on optimum-path forest (OPF) classification. For a given training set with relevant and irrelevant images, an optimization algorithm finds the best distance function to compare images as a combination of their distances according to different representations. Two optimization techniques are evaluated: a multi-scale parameter search (MSPS), never used before for CBIR, and a genetic programming (GP) algorithm. The combined distance function is used to project an OPF classifier and to rank images classified as relevant for the next iteration. The ranking process takes into account relevant and irrelevant representatives, previously found by the OPF classifier. Experiments show the advantages in effectiveness of the proposed approach with both optimization techniques over the same approach with single distance space and over another state-of-the-art method based on multiple distance spaces.

1. Introduction

Large image collections have increased the demand for efficient and effective information retrieval methods based on the visual content of the images. The simplest visual content representation is a *feature vector*, which encodes color, texture, and/or shape measures of an image. The similarity between images can be measured by the *distance* between their feature vectors. More complex representations are very specific for shape or texture or color, requiring special distance functions [1–5]. In this case, color, texture, and shape features are not simple measures that can be concatenated into a single feature vector. Each pair, feature extraction function and distance function, must be interpreted as an independent entity refereed to as a *simple descriptor*. Therefore, the use of multiple simple descriptors creates the problem of dealing with multiple distance spaces. Solutions to this problem find the best combination of distance values from simple descriptors, resulting in a *composite descriptor* [6].

For a given query image, a content-based image retrieval (CBIR) system can rank the most relevant images based on their distances to the query image. However, a *semantic gap* usually occurs between this result and the user's expectation, due to the absence of relevant information in those simple/composite descriptors. In order to reduce the semantic gap problem, feedback-based learning approaches have been investigated. In these methods, the user usually indicates which images are relevant (irrelevant) within a small set of returned images and the CBIR system learns how to better rank and return more relevant images in a next iteration. This search process can be repeated until the user is satisfied. The learning process is usually focused on the selection of a more effective simple descriptor [7,8], by changing features and/or distance function, on a better composite descriptor [9,10], on changing the query strategy [11,12], or on training a more accurate pattern classifier [13–16].

In this paper we propose a method that simultaneously improves the composite descriptor, the query strategy, and the pattern classifier during the feedback-based learning process. Our method extends our previous work [17,16] by incorporating ptimization techniques to compute composite descriptors. Two

☆ This paper has been recommended for acceptance by C.-S. Li.

* Corresponding author. Fax: +55 19 3521 3845.
E-mail addresses: atavares@dca.fee.unicamp.br (A.T. da Silva), jsantos@ic.unicamp.br (J.A. dos Santos), afalcao@ic.unicamp.br (A.X. Falcão), rtorres@ic.unicamp.br (Ricardo da S. Torres), leopini@fee.unicamp.br (L.P. Magalhães).

1077-3142/$ - see front matter Crown Copyright © 2011 Published by Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2011.12.001

techniques are evaluated in this context: a multi-scale parameter search (MSPS) [18], never used for CBIR, and a genetic programming (GP) algorithm [6]. The MSPS method is a recent optimization approach under development. This paper presents a slightly different implementation of the algorithm in [18], which was better suited for distance combination. The choice of GP is also motivated by the successful recent works [9,10,19]. Santos et al. [9,19], for example, showed that the GP-based method improves the retrieval effectiveness, outperforming methods based on genetic algorithm.

Support vector machines (SVMs) became a popular classifier in feedback-based learning [13,14,20,15]. Our method is based on the optimum-path forest (OPF) classifier [21], because we have recently demonstrated its gain in effectiveness and efficiency over SVMs for CBIR [17,16]. The OPF classifier uses the Image Foresting Transform algorithm [22], which is a generalization of Dijkstra's algorithm for multiple sources and more general *path-cost functions* in a graph. Its training set is interpreted as a complete graph weighted on the arcs by the combined distance values between nodes. Therefore, the composite descriptor should produce higher arc weights between relevant and irrelevant images, and lower arc weights within each class. The path-cost function assigns to any path its maximum arc weight, but the paths are constrained to start in a special set of representative images (called *prototypes*) from both classes. The prototypes compete among themselves and each prototype conquers its most closely connected images by paths of minimum cost, partitioning the graph into an optimum-path forest (classifier). The classification of database images based on OPF follows the same rule, by considering minimum-cost paths from the prototypes to the new image and assigning it to the class of the most closely connected prototype. The choice of prototypes is a key aspect in the OPF approach. First, they must "defend" their classes in order to avoid paths from prototypes of distinct classes. Since the method is choosing paths, whose maximum arc weight is minimum, the best prototypes will be the closest images between the relevant and irrelevant classes [21]. Second, the prototypes are used as new query points, changing the ranking process to sort images based on a normalized average distance to the relevant and irrelevant prototypes.

This paper is organized as follows. Section 2 provides a survey of related works and Section 3 reviews the original CBIR approach based on relevance feedback and OPF classification [17]. The method is initially described for a simple descriptor and then the optimization techniques that integrate multiple distance spaces are presented in Section 4 for composite descriptors. Exhaustive experiments involving several datasets, two state-of-the-art approaches [17,10], various simple descriptors, and the composite descriptors from the MSPS and the GP techniques are presented in Section 5. Finally, Section 6 states conclusion and discusses future work.

2. Related work

Over the past decades, several softwares for content-based image retrieval (CBIR) have been developed, such as IBM QBIC [23], UIUC MARS [24], PicHunter [25], TinEye [26], and Windsurf [27]. However, CBIR is still an open problem, as demonstrated by the ImageCLEF[1] and PASCAL VOC[2] challenges.

The research on CBIR initially received great influence from text retrieval [28], taking advantage of the experience reported in journals and conferences, such as TREC (Text REtrieval Conference). One of the main contributions was the Precision $\times$ Recall curve [29] that has been extensively used to evaluate the effectiveness of CBIR systems. Another contribution is also the idea of feedback-based learning, which considerably advanced the human–computer interaction process in CBIR systems [30,15,10,16].

We may divide the efforts in CBIR into two main goals of improvement: efficiency and effectiveness. Efficiency gain usually involves indexing structures [31–33], to access images in a more efficient way, and better scalability to huge image collections [34,35], by reducing the number of dimensions of the search space before indexing the data. Techniques such as PCA [36] and CSVD [37], for instance, can be used for this purpose. Effectiveness gains are essentially obtained by reducing the semantic gap problem.

Our method can take advantage of techniques that improve efficiency, but the focus of this work is on effectiveness gain. In this context, the initial studies proposed simple descriptors [38,1,39–41], with local [42,41] and/or global [43,44,1–3] image features, and strategies to change the query during the search, by moving the query point in the feature space [11] and by creating multiple query points [45,46,12]. However, multiple image representations may require distinct distance functions [1–3,47,4,5]. In this context, the simplest approach is to compute a composite descriptor as a fixed combination of the distance values from multiple representations [48].

Many recent works, however, have focused on feedback-based learning to either elaborate composite descriptors that improve ranking of relevant images [6,49–51,10] or to design pattern classifiers [15,52,17,16] that select images from the database as the best candidates to be relevant images. These methods take advantage of the user feedback about the relevance of the images in order to improve the composite descriptor or the pattern classifier at each iteration. Classifier-based methods can also be divided into those that return the most relevant images at each iteration [53,52,17] and those that return the most informative images (i.e., images that are difficult to be classified) during a few iterations [25,13–16], postponing the most relevant ones to the last iteration. While the strategy of the former approaches is *greedy*, aiming the most relevant images at each iteration, the strategy of the latter methods is *planned*, in the sense that the user has to decide in which iteration the system will return the most relevant images [16]. Planned approaches are widely known as *active learning* methods. However, active learning could have a more general interpretation, being accepted as a synonymous for feedback-based learning, given that the learner (the classifier) is active in the learning process, by trimming images from the database to create a set of candidate ones for ranking.

In this work we are exploiting the use of composite descriptor and pattern classifier at the same time. Methods based on composite descriptors tend to be more effective than approaches based on simple descriptors in most applications [49,50,10]. In addition, other works have demonstrated that the classification of the relevant candidates before ranking is a good strategy to increase the effectiveness of the CBIR system [15,17]. We have proposed in [17] a greedy approach based on optimum-path forest (OPF) classifier and demonstrated its gain in effectiveness with respect to the first active learning method based on support vector machines (SVMs) [13,14]. It is important to note that it is not possible to combine distance functions with SVMs, because this classifier is based on support (feature) vectors and they will not be the same for distinct data representations (samples using different features and distance functions). For this reason, SVM-based methods typically combine features by fusioning or concatenating the vectors [54,55,15,56]. Due to the large size of the resulting vectors, some studies also suggest the reduction of dimensionality by using techniques such as PCA [36,56]. Other works try to reduce the number of vectors by feature selection techniques [54,57,8,7].

Some works also allow the user to assign a finer gradation of relevancy, such as somewhat relevant, not sure, and somewhat irrelevant [58,59]. The OPF classifier is a multi-class classifier and can handle multiple gradation levels. Although we will present the method for the case of relevant/irrelevant images, its extension to multiple grades is straightforward.

[1] http://www.imageclef.org/ (As of 10/10/2011).
[2] http://pascallin.ecs.soton.ac.uk/challenges/VOC/ (As of 10/10/2011).

3. Relevance feedback with optimum-path forest

Fig. 1 shows the overview of our approach for feedback-based learning using the OPF classifier and a simple descriptor $D = (v, d)$, being v the feature extraction function and $d(s,t)$ the distance function between two image representations (e.g., $d(s,t) = \|\vec{v}(t) - \vec{v}(s)\|$) [60]. Its extension to composite descriptor is explained in Section 4. At the first iteration, we have a simple search by similarity where, for a given image database $\mathcal{Z}$, the user specifies a query image q and the system simply ranks the N closest images $t \in \mathcal{Z}$ in the increasing order of $d(q,t)$. We aim to return a list $\mathcal{X}$ with the N most relevant images in $\mathcal{Z}$ with respect to that query image q. However, due to the limitation of the descriptor (v, d) in representing the user's expectation (*semantic gap*), the list $\mathcal{X}$ contains relevant and irrelevant images according to the user's opinion. To reduce the semantic gap problem, the system asks for the user's feedback about the relevance of the returned images during a few iterations. The user indicates which images are relevant (or irrelevant) in $\mathcal{X}$, forming a *labeled training set* $\mathcal{T}$ which gains new elements at each iteration of relevance feedback by $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{X}$.

Our feedback-based learning process, named OPF_{RF}, uses the set $\mathcal{T}$ to project a new OPF classifier at each iteration of the relevance feedback loop. This process consists of first estimating prototypes as the closest images from distinct classes in $\mathcal{T}$, forming the sets $\mathcal{S}_R \subset \mathcal{T}$ and $\mathcal{S}_I \subset \mathcal{T}$ of relevant and irrelevant prototypes, respectively. The training process considers a complete graph, whose nodes are all elements in $\mathcal{T}$ and arcs (s,t) between images s and t are weighted by $d(s,t)$. Every path in the graph has a cost and minimum-cost paths are computed from $\mathcal{S}_R \cup \mathcal{S}_I$ to each node $t \in \mathcal{T}$, such that the classifier is an optimum-path forest rooted in $\mathcal{S}_R \cup \mathcal{S}_I$ (Section 3.1). In this forest, the nodes $t \in \mathcal{T} \setminus \mathcal{S}_R \cup \mathcal{S}_I$ are conquered and labeled (in the same class as relevant/irrelevant) by the prototype in $\mathcal{S}_R \cup \mathcal{S}_I$ which offers the minimum-cost path with terminus t. Afterward, this classifier evaluates the images in $\mathcal{Z} \setminus \mathcal{T}$, by computing the cost of the optimum path from $\mathcal{S}_R \cup \mathcal{S}_I$ to each node $t \in \mathcal{Z} \setminus \mathcal{T}$ in an incremental way, and inserts t in a reduced set $\mathcal{Y} \subset \mathcal{Z} \setminus \mathcal{T}$ of *relevant candidates* whenever the optimum path is rooted in $\mathcal{S}_R$.

The system then returns a new list $\mathcal{X} \subset \mathcal{Y}$ with the N closest images in the increasing order of a normalized mean distance $\bar{d}(t, \mathcal{S}_R, \mathcal{S}_I)$ between t and the two sets of prototypes (Eq. (1)) [17,16]. Therefore, each iteration may produce distinct sets of prototypes, changing the query/ranking strategy.

$$\bar{d}(t, \mathcal{S}_R, \mathcal{S}_I) = \frac{\bar{d}(t, \mathcal{S}_R)}{\bar{d}(t, \mathcal{S}_R) + \bar{d}(t, \mathcal{S}_I)}, \tag{1}$$

$$\bar{d}(t, \mathcal{S}_R) = \frac{1}{|\mathcal{S}_R|} \sum_{\forall s \in \mathcal{S}_R} d(s,t), \tag{2}$$

$$\bar{d}(t, \mathcal{S}_I) = \frac{1}{|\mathcal{S}_I|} \sum_{\forall s \in \mathcal{S}_I} d(s,t). \tag{3}$$

Again, the user may indicate relevant and irrelevant images in $\mathcal{X}$ and a new training set is created by $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{X}$ to redesign an improved classifier. This process is repeated until the user is satisfied.

Next sections detail the optimum-path forest training and classification processes and the feedback-based learning algorithm, respectively.

3.1. Training and classification by optimum-path forest

The OPF classifier aims at exploiting possible connectivities between samples and representative elements (prototypes) of each class in a given distance space to improve classification. Its decisions are not usually taken based on direct distances between samples, but on optimal sequences of distance values between samples that are most closely connected to some prototype. For a given training set $\mathcal{T}$, samples may be connected based on different adjacency rules. In this work we adopted a complete graph, as shown in Fig. 2a, whose nodes are all images in $\mathcal{T}$. In order to decide the class of any sample t, we are interested in finding the minimum-cost path from a set of prototypes. Therefore, a path π_t in this graph with terminus t is a sequence $\langle t_1, t_2, \ldots, t_n = t \rangle$ of distinct nodes, such that the root $R(\pi_t) = t_1$ is a prototype. The value $c(\pi_t)$ of a path is defined as the maximum arc weight $d(t_i, t_{i+1})$ along it:

$$c(\pi_t) = \max_{i=1,2,\ldots,n-1} \{d(t_i, t_{i+1})\}. \tag{4}$$

By definition, any other path whose root is not a prototype will be assigned to infinity cost and a *trivial* path formed by a single node (prototype) will have cost equal to zero. A path π_t will be optimum whenever $c(\pi_t) \leqslant c(\tau_t)$ for any other path τ_t with the same terminus t, irrespective to their root nodes. Thus, the prototypes compete among themselves and the OPF classifier always decides the class of t as being the one of its most closely connected prototype, according to Eq. (4). For given sets $\mathcal{S}_R$ and $\mathcal{S}_I$ of relevant and irrelevant prototypes, this competition creates a minimum-cost path forest (optimum partition) rooted at $\mathcal{S}_R \cup \mathcal{S}_I$ (Fig. 2b).

Therefore, training consists of finding the prototypes from all classes and computing an optimum-path forest (classifier) in the training set $\mathcal{T}$. The minimization of the path costs $c(\pi_t)$ to each training node t tends to choose the closest nodes to form the links along the optimum paths. By choosing prototypes as the closest samples between distinct classes (images with black borders in Fig. 2b), they will protect other samples of the same class to be conquered by prototypes from other classes. Let $\lambda(t)$ be the class of image $t \in \mathcal{T}$, which may be relevant or irrelevant. The prototypes are obtained by computing a *minimum spanning tree* (MST) in the complete graph with nodes in $\mathcal{T}$ [61]. For each arc (s,t) in the MST, if $\lambda(s) \neq \lambda(t)$ then s and t are marked as prototypes. If $\lambda(s)$ is relevant and $\lambda(t)$ is irrelevant, then s is inserted in $\mathcal{S}_R$ and t is inserted in $\mathcal{S}_I$. Similarly, it is done

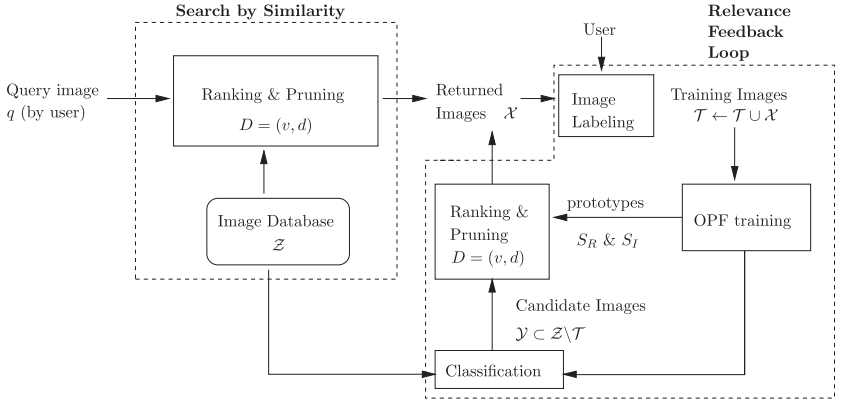

Fig. 1. Feedback-based learning process.

in the other way around. The optimum-path forest P in Fig. 2b is then an acyclic graph which stores all optimum paths in a predecessor map, i.e., a function which assigns to every node $t \in \mathcal{T} \setminus \mathcal{S}_R \cup \mathcal{S}_I$ its predecessor node $P(t)$ in the optimum path from $\mathcal{S}_R \cup \mathcal{S}_I$, or a mark $P(t) = nil$ when $t \in \mathcal{S}_R \cup \mathcal{S}_I$. This forest can be computed by Algorithm 1, customized for path-cost function c [22,21]. This algorithm outputs for each node $t \in \mathcal{T}$, the minimum-path cost $C(t)$ of the optimum path with terminus t, its root node $R(t) \in \mathcal{S}_R \cup \mathcal{S}_I$, and the predecessor $P(t)$ in that optimum path. A list $\mathcal{T}'$ of the training nodes in the increasing order of cost $C(t)$ is also output to speedup classification.

Algorithm 1. OPF algorithm

Input: Training set $\mathcal{T}$, the sets of prototypes $\mathcal{S}_R \subset \mathcal{T}$ and $\mathcal{S}_I \subset \mathcal{T}$, and a descriptor (v,d).
Output: Optimum-path forest P (predecessor map), path-cost map C, root map R, and the ordered list $\mathcal{T}'$ of training nodes.
Auxiliary: Priority queue Q and cost variable cst.

```
1  for each s ∈ T \ S_R ∪ S_I do
2    Set C(s) ← +∞
3  end
4  for each s ∈ S_R ∪ S_I do
5    Set C(s) ← 0, P(s) ← nil,
6    R(s) ← s, and insert s into Q.
7  end
8  While Q is not empty do
9    Remove from Q a node s with minimum C(s) and insert s
     in T'.
10   for each t ∈ T such that C(t) > C(s) do
11       Compute cst ← max{C(s),d(s,t)}.
12       if cst < C(t) then
13         P(t) ← s, R(t) ← R(s) and C(t) ← cst.
14         if C(t) = +∞
15           Insert t in Q.
16         else
17           Update the position of t in Q.
18         end
19       end
20     end
21 end
```

The algorithm follows the dynamic programming scheme [22]. Lines 1–7 initialize the cost, predecessor, and root maps, forcing the optimum paths to start in $\mathcal{S}_R \cup \mathcal{S}_I$, and insert the roots in Q. The main loop computes an optimum path from $\mathcal{S}_R \cup \mathcal{S}_I$ to every node $s \in \mathcal{T}$ in a non-decreasing order of cost (Lines 8–21). At each iteration, a path of minimum cost $C(s)$ is obtained in P when we remove its last node s from Q, preserving its order in $\mathcal{T}'$ (Line 9). The rest of the lines evaluate if the path that reaches an adjacent node $t \neq s$ through s is cheaper than the current path with terminus t and update Q, $C(t)$, $R(t)$, and $P(t)$ accordingly. The test $C(t) > C(s)$ in Line 10 avoids to visit nodes t that will never be updated from nodes s. The test $C(t) = +\infty$ in Line 14 identifies when a node t is being visited by the first time.

In the classification phase (Fig. 2c), for every sample $t \in \mathcal{Z} \setminus \mathcal{T}$, the optimum path with terminus t can be easily identified by finding which training node $s^* \in \mathcal{T}$ provides the minimum value in

$$C(t) = \min_{\forall s \in \mathcal{T}} \{\max\{C(s), d(s,t)\}\}. \quad (5)$$

Node s^* is the predecessor $P(t)$ in the optimum path with terminus t and the image t is classified (Fig. 2d) as $\lambda(R(s^*))$.

The role of $\mathcal{T}'$, sorted by the cost $C(s)$ computed by Algorithm 1, is to speed up the evaluation of Eq. (5). The search for the minimum value can be halted when the cost for a node p whose position in $\mathcal{T}'$ succeeds the position of s is greater then the cost computed previously ($C(p) > \max\{C(s), d(s,t)\}$) [62]. Thus, it is not necessary to compare each sample $t \in \mathcal{Z} \setminus \mathcal{T}$ with all elements of the set $\mathcal{T}$.

3.2. Feedback-based learning algorithm

Algorithm 2 presents the feedback-based learning using OPF classification [17] (as illustrated in Fig. 1). It returns a list $\mathcal{R} \subset \mathcal{Z}$ of images in the decreasing order of relevance.

In Line 1, the output list $\mathcal{R}$ and the training set $\mathcal{T}$ are initialized to empty sets. For a given query image q and image database $\mathcal{Z}$, the algorithm first returns a list $\mathcal{X}$ with the N closest images $t \in \mathcal{Z}$ in the increasing order of $d(q,t)$ (Line 2). The learning process is executed in the main loop (Lines 3–9). In Line 4, the user marks the relevant images in $\mathcal{X}$, and the remaining images are considered as irrelevant, or vice versa. This essentially creates a training set $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{X}$ where each image $t \in \mathcal{T}$ has a relevance label $\lambda(t)$. The relevant images of $\mathcal{X}$ are inserted in the output list $\mathcal{R}$ (Line 5). The OPF training is computed in Line 6 (Algorithm 1). It results sets $\mathcal{S}_R \subset \mathcal{T}$ and $\mathcal{S}_I \subset \mathcal{T}$ of relevant and irrelevant prototypes; the OPF attributes, predecessor $P(t)$, cost $C(t)$, and root $R(t)$ for each $t \in \mathcal{T}$; and set $\mathcal{T}'$ of training images in the increasing order of optimum cost $C(t)$ for faster classification. In Line 7, OPF is used to classify images in $\mathcal{Z} \setminus \mathcal{T}$ using Eq. (5), forming a set $\mathcal{Y}$ with the images classified as relevant. A new list $\mathcal{X}$ with the N closest images $t \in \mathcal{Y}$ is created in the increasing order of a *normalized mean distance* $\bar{d}(t, \mathcal{S}_R, \mathcal{S}_I)$ between t and the two sets of prototypes (Eq. (1)).

Algorithm 2. OPF_{RF} Algorithm

Input: A query image q, a descriptor (v,d), the number N of returned images per iteration, and an image database $\mathcal{Z}$.
Output: A list $\mathcal{R} \subset \mathcal{Z}$ of retrieved images in the decreasing order of relevance.
Auxiliary Set $\mathcal{T}$ of training images, maps $(R, P, C, \mathcal{T}')$ of the OPF classifier, sets $\mathcal{S}_R \subset \mathcal{T}$ and $\mathcal{S}_I \subset \mathcal{T}$ of prototypes, set $\mathcal{Y} \subset \mathcal{Z} \setminus \mathcal{T}$ of images classified as relevant, and ordered list $\mathcal{X} \subset \mathcal{Y}$ of N returned images per iteration.

```
1  Set R ← ∅ and T ← ∅.
2  Compute a list X with the N closest images t ∈ Z in the
   increasing order of d(q,t).
3  while the user is not satisfied do
4      The user marks relevant (irrelevant) images, forming
   a training set T ← T ∪ X where each image has a
   relevance label λ(t).
5      Insert in R the relevant images of X.
6      Compute sets S_R and S_I of prototypes in T (Section
   3.1) and execute (R, P, C, T') ← OPF(T, S_R, S_I, v, d)
   (Algorithm 1).
7      Classify images in Z \ T using (R, P, C, T') and λ(t) for
   t ∈ S_R ∪ S_I (Eq. (5)), and create set Y with the relevant
   candidates.
8      Compute a list X with the N closest images t ∈ Y in
   the increasing order of d̄(t, S_R, S_I). (Eq. (1))
9  end
10 Return R ← R ∪ X.
```

The main loop repeats until the user is satisfied and the retrieved images in $\mathcal{R}$ are all relevant images together with the last set $\mathcal{X}$ in the increasing order of $\bar{d}(t, \mathcal{S}_R, \mathcal{S}_I)$.

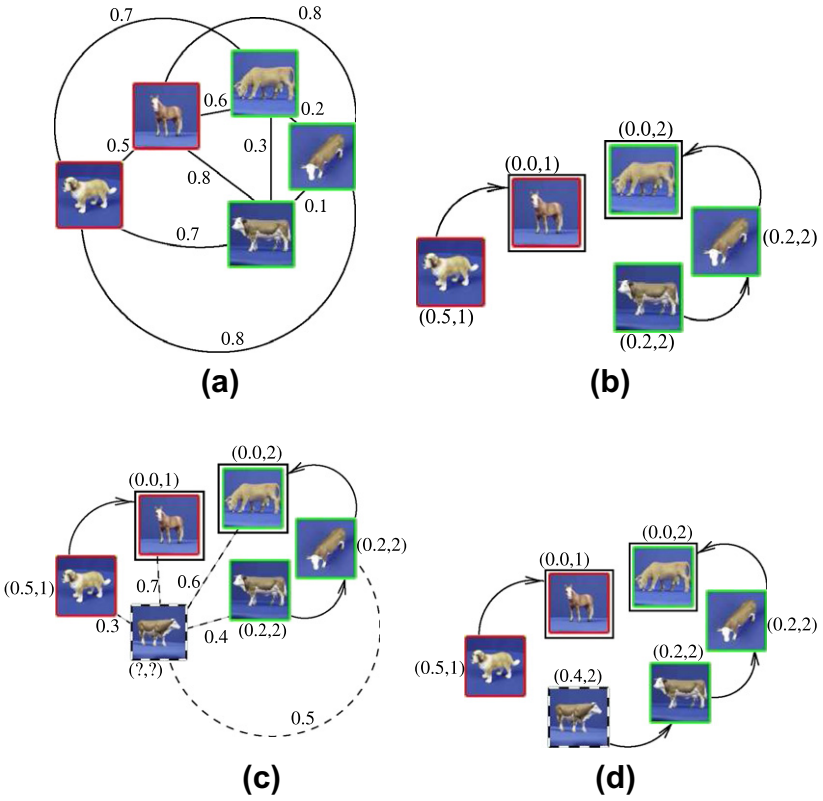

Fig. 2. (a) Complete weighted graph for a simple training set. The user marked the relevant images using a green border and the irrelevant ones with a red border. (b) Resulting optimum-path forest with two prototypes (marked nodes by black borders). The entries (x, y) over the nodes are, respectively, the cost and the label of the samples. The directed arcs indicate the predecessor nodes in the optimum paths. (c) Test sample (dashed square) and its connections (dashed lines) with the training nodes. (d) The optimum path from the most closely connected prototype, its candidate label 2, and classification cost 0.4 are assigned to the test sample. The test sample is then classified as relevant, although its nearest training sample is from the irrelevant class. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4. Extension to multiple descriptors

Now consider a set $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$ of simple descriptors such that a distance $d_i(s,t)$, $i = 1, 2, \ldots, n$, between any given pair of images $s, t \in \mathcal{Z}$ can be computed as a function of their respective representations $\vec{v}_i(s)$ and $\vec{v}_i(t)$.

We wish to find the best combination of simple descriptors in order to retrieve relevant images from $\mathcal{Z}$ with respect to a given query image and user. This combination is found through a *composite descriptor* [60] that integrates the respective distance spaces. That is, a composite descriptor D^* is a pair $(\mathcal{D}, \delta D^*)$, where δD^* is a function that best combines the distance values computed by each descriptor into a final distance value $d^*(s,t)$ (Fig. 3).

Observe that the distance values from different descriptors may diverge significantly in scale. Thus, it is important that all values be normalized between 0 and 1. A Gaussian normalization [38] can be employed for that.

At each iteration of the feedback-based learning process, a training set $\mathcal{T}$ with relevant and irrelevant images is used to choose the best combination function using some optimization algorithm. For a given $\mathcal{T}$ and set $\mathcal{D}$ of descriptors, the optimization process essentially evaluates candidates δD according to some criterion function. The criterion function should measure the ability of the system in ranking the most relevant images first. We measure this property by averaging the FFP4 [63] values (Eq. (6)) with respect to each relevant image in $\mathcal{T}$, rather than using only the query image. As introduced in a previous work [6], this approach has proven to be more effective than the simplest method based on the query image only. Let $\mathcal{T}_R$ be the subset of relevant images in $\mathcal{T}$ and $\mathcal{T}_u$ be a set of training images $t \in \mathcal{T}$ in their increasing order of distance $\delta D(t,u)$ with respect to a given relevant image $u \in \mathcal{T}_R$, our criterion function $F(\mathcal{T}, \mathcal{D}, \delta D)$ is defined as

$$F(\mathcal{T}, \mathcal{D}, \delta D) = \frac{1}{|\mathcal{T}_R|} \sum_{\forall u \in \mathcal{T}_R} \sum_{k=1}^{|\mathcal{T}_u|} 7\lambda_k 0.982^k, \tag{6}$$

where $\lambda_k \in \{0, 1\}$ indicates the user's opinion about the relevance of each image at every position k in $\mathcal{T}_u$, as either relevant ($\lambda_k = 1$) or irrelevant ($\lambda_k = 0$). The constant 0.982 was experimentally determined in [63].

In Algorithm 2, the complete graph used to compute the OPF classifier has its arcs (s,t) weighted by the best combined distance $d^*(s,t)$ that results from the optimization process (Fig. 3). The best

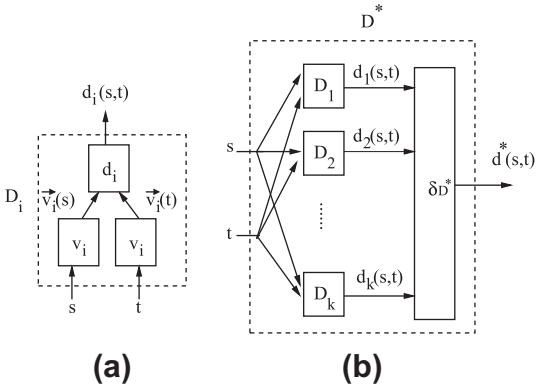

Fig. 3. (a) Simple and (b) composite descriptor $D^* = (\mathcal{D}, \delta D^*)$ [60].

combination function δD^* is also used to compute the normalized distance $\bar{d}(t, \mathcal{S}_R, \mathcal{S}_I)$ in that algorithm. Fig. 4 details the learning and retrieval process from our feedback-based learning approach. First a composite descriptor is defined using combination techniques such as MSPS or GP that will be used by the OPF classifier.

In the following sections we present two optimization techniques to compute composite descriptors: the multi-scale parameter search (MSPS) and a genetic programming algorithm (GP).

4.1. Composite descriptors using multi-scale parameter search

In the multi-scale parameter search (MSPS), the combination function δD is represented by a same equation and the parameters of this function must be optimized [18]. In this paper, this equation is defined by

$$\delta D(s,t) = \sum_{i=1}^{n} d_i^{\theta_i}(s,t) \tag{7}$$

where $\theta = (\theta_1, \theta_2, \ldots, \theta_n)$ is its parameter vector, being $0 \leqslant \theta_i \leqslant 2$. The parameters may give distinct non-linear emphases to their corresponding descriptors. Given that the values of $F(\mathcal{T}, \mathcal{D}, \delta D)$ during optimization will depend only on the choice of θ, we will denote $F(\mathcal{T}, \mathcal{D}, \delta D)$ by $F(\mathcal{T}, \mathcal{D}, \theta)$ for clarity. From any given state $\theta = (\theta_1, \theta_2, \ldots, \theta_n)$, the idea is to find the best displacement vector $\mathbf{\Delta}^*$, by exploiting multiple scales of the parameter space, and update the parameter vector into a next value $\theta \leftarrow \theta + \mathbf{\Delta}^*$, repeating this process until a maximum $F(\mathcal{T}, \mathcal{D}, \theta^*)$. In order to avoid local maximum, the method perturbs θ on each parameter axis $i = 1, 2, \ldots, n$ and with m displacement scales $j = 1, 2, \ldots, m$ along each axis. At each iteration, it evaluates $F(\mathcal{T}, \mathcal{D}, \theta + \mathbf{\Delta})$ for the displacement vectors $\mathbf{\Delta}$ that result from all perturbations on each axis, separately, all scales, and the resulting vectors from each scale. This method works as follows.

Let $0 \leqslant \Delta_{ij} \leqslant 1$ be a positive displacement along the parameter axis i for a scale j. The method takes into account the following displacements:

- the best perturbation along each parameter axis i, $\mathbf{\Delta}^*_{ij} = (0, \ldots, \Delta^*_{ij}, \ldots, 0)$ for $\Delta^*_{ij} \in \{\Delta_{ij}, 0, -\Delta_{ij}\}$, such that

$$F\left(\mathcal{T}, \mathcal{D}, \theta + \mathbf{\Delta}^*_{ij}\right) = \max \left\{ \begin{array}{l} F(\mathcal{T}, \mathcal{D}, \theta + \mathbf{\Delta}_{ij}), \\ F(\mathcal{T}, \mathcal{D}, \theta), \\ F(\mathcal{T}, \mathcal{D}, \theta - \mathbf{\Delta}_{ij}) \end{array} \right\} \tag{8}$$

- and the resulting vectors $\mathbf{\Delta s}_j = \sum_{i=1}^{n} \mathbf{\Delta}^*_{ij}$, $j = 1, 2, \ldots, m$.

Hence, the choice of $\mathbf{\Delta}^*$ can be expressed by:

$$F(\mathcal{T}, \mathcal{D}, \theta + \mathbf{\Delta}^*) = \max \left\{ \begin{array}{ll} F\left(\mathcal{T}, \mathcal{D}, \theta + \mathbf{\Delta}^*_{ij}\right) & \text{for } i = 1, 2, \ldots, n \text{ and} \\ & j = 1, 2, \ldots, m. \\ F(\mathcal{T}, \mathcal{D}, \theta + \mathbf{\Delta s}_j) & \text{for } j = 1, 2, \ldots, m. \end{array} \right\} \tag{9}$$

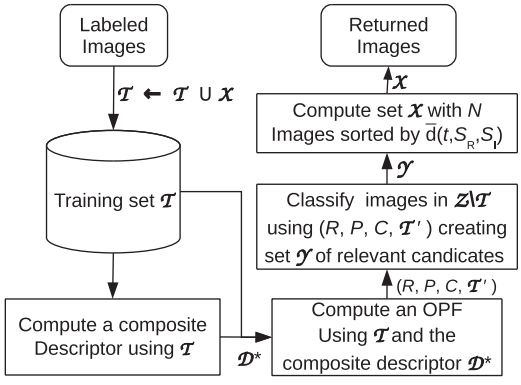

Fig. 4. Learning, ranking, and pruning exploiting composite descriptor.

Algorithm 3. MSPS Algorithm

Input:Training set $\mathcal{T}$, set $\mathcal{D}$ of simple descriptors, and displacements $\Delta_{i,j}$, for $i = 1,2,\ldots,n$ and $j = 1,2,\ldots,m$.
Output: The best parameter vector θ^*.
Auxiliary: Displacement vectors $\mathbf{\Delta}$, $\mathbf{\Delta}^*$ and $\mathbf{\Delta s}$, parameter vectors θ^* and θ, and variables i, j, V_0, V^-, V^+, and V^*.

```
1  θ ← (1,...,1);
2  V* ← F(T,D,θ) and θ* ← θ;
3  repeat
4      V0 ← V* and θ ← θ*;
5      for j = 1 to m do
6        Δs ← (0,...,0);
7        for i = 1 to n do
8          Δ ← (0,...,Δ_ij,...,0), V ← V0, and Δ* ←
   (0,...,0);
9          V+ ← F(T,D,θ+Δ) and V− ← F(T,D,θ−Δ);
10         If V+ > V then V ← V+ and Δ* ← Δ;
11         If V− > V then V ← V− and Δ* ← −Δ;
12         If V > V* then θ* ← θ + Δ* and V* ← V;
13         Δs ← Δs + Δ*;
14       end
15     V ← F(T,D,θ+Δs);
16     If V > V* then V* ← V and θ* ← θ + Δs;
17     end
18 until V* > V0
19 Return θ*.
```

Algorithm 3 illustrates the computation of the best parameter vector θ^* in Eq. (7). In this work, we fixed the displacements Δ_{ij} as 0.001, 0.01, 0.12, 0.45, and 1.0 along $m = 5$ scales for every parameter $i = 1,2,\ldots,n$. These displacements could have also be created by some increasing function. In Line 1, vector θ is initialized with the same weight to all descriptors. Lines 4–16 search for the θ^* that maximizes Eq. (6). If there is a $\mathbf{\Delta}$ (Line 8), tested for positive and negative displacements, that increases the current criterion value, it is stored in $\mathbf{\Delta}^*$ (Lines 10–11). Lines 10 and 11 aim at comparing V_0 with V^+ and V^-. If both V^+ and V^- are greater than V, Line 10 will substitute V by V^+ and so Line 11 will be comparing V^+ with V^- in order to choose the highest one. The best displacement vector, $\mathbf{\Delta}^*$, is used in Lines 12 and 13 to update the parameter vector and to compose the combination of the best displacements for each scale, respectively. Lines 15 and 16 test if $\mathbf{\Delta s}$ generates a better combination function and update the parameter vector. The algorithm stops when no combination better than θ is found, returning in Line 19 the last parameter vector θ^*, which is then used in Eq. (7) to define the final composite descriptor for feedback-based learning using the OPF classifier.

4.2. Composite descriptors using genetic programming

In the genetic programming (GP) approach [64], the combination function δD is a tree of mathematical operations applied to the distance values that result from the simple descriptors (Fig. 5). Leaf nodes are represented by these distance values, while the mathematical operatiors are defined in the internal nodes. Fig. 5 exemplifies an individual with three distinct descriptors using the set {+, −, /, *sqrt*} of operators as internal nodes. The best

combination function $\delta D^*(s,t)$ is $\frac{d_1(s,t)+d_3(s,t)}{d_2(s,t)} - \sqrt{d_2(s,t)+d_3(s,t)}$ in this example.

The GP approach [6] searches for the best combination function by evolving a population $\mathcal{P}$ of n_p candidate individuals per iteration during n_g generations. At each iteration, all individuals $\delta D_i, i = 1,2,\ldots,n_p$, are evaluated by function $F(\mathcal{T},\mathcal{D},\delta D_i)$ (Eq. (6)). The best individuals are identified and modified by applying genetic transformations, such as reproduction, mutation, and crossover. The reproduction operator copies them to the next generation. Mutation randomly selects a point in the GP tree of an individual and replaces the subtree of that point with a new randomly generated subtree. The crossover operator swaps a subtree of one parent with a subtree of another [64].

Algorithm 4 describes the search for the best individual δD^* during n_g generations, starting from an initial population $\mathcal{P}$ with n_p individuals. The population starts with individuals created randomly (Line 1). This population evolves generation by generation through genetic operations (Lines 2–9). The criterion function $F(\mathcal{T},\mathcal{D},\delta D_i)$ (Eq. (6)) is used to assign the fitness value for each individual (Line 5). This value is used to select the best individuals (Line 7). Next, genetic operators are applied to this population in order to create more diverse and better performing individuals (Line 8). The last step consists of determining the best individual (Line 10). The commonest choice is the individual with the best performance in the last generation of $\mathcal{P}$.

Algorithm 4. GP Algorithm

Input: A training set $\mathcal{T}$, a set $\mathcal{D}$ of descriptors, the size n_p of the population, the number n_g of generations, and percentage of reproduction, mutation and crossover.
Output: The best individual δD^* (a tree structure).
Auxiliary: Set $\mathcal{P}$ (population) of n_p individuals δD_i, a set $\mathcal{A}$ of pairs $(\delta D_i, F(\mathcal{T},\mathcal{D},\delta D_i))$, and variables i and g.

1 $\mathcal{P} \leftarrow$ Initial random population of n_p individuals;
2 $\mathcal{A} \leftarrow \emptyset$;
3 **for** *each generation* $g = 1,2,\ldots,n_g$ **do**
4 **for** *each individual* $\delta D_i \in \mathcal{P},\ i = 1,2,\ldots,n_p$ **do**
5 Insert $\mathcal{A} \leftarrow \mathcal{A} \cup (\delta D_i, F(\mathcal{T},\mathcal{D},\delta D_i))$;
6 **end**
7 Sort $\mathcal{A}$ in the decreasing order of $F(\mathcal{T},\mathcal{D},\delta D_i)$;
8 Create a new population $\mathcal{P}$ of size n_p by reproduction, crossover and mutation among the best individuals in $\mathcal{A}$;
9 **end**
10 Return the best individual δD^* in $\mathcal{A}$ with the highest value of $F(\mathcal{T},\mathcal{D},\delta D^*)$.

5. Experiments and results

We call OPF_{MSPS} and OPF_{GP} the feedback-based learning process using OPF classification and the optimization techniques MSPS and GP, respectively.

Table 1 shows the values used for GP parameters [64] used in the OPF_{GP} feedback-based learning method. Table 2 presents the population size and the number of generations of GP in the OPF_{GP} for each dataset.

For each image database, we simulate the user behavior by using each image as initial query point and marking the relevant points (images from the same class of the query) among 30 returned images per iteration.

We compare the effectiveness of OPF_{MSPS} and OPF_{GP} against OPF_{RF}, as proposed in [17] and described in Section 3 for simple descriptors (the best one for each dataset), and another approach, named GP^+ [10]. GP^+ is a feedback-based learning method which uses genetic programming to compute the best composite descriptor given a training set composed only by relevant (positive)

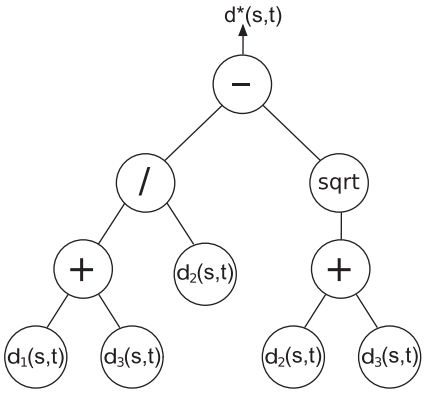

Fig. 5. A tree representing the best combination function of a composite descriptor.

Table 1
Parameter values for GP in the OPF_{GP} method.

Initial population	Half-and-half
Initial tree depth	2–5
Maximum tree depth	5
Selection method	Tournament (size 2)
Crossover prob.	0.8
Mutation prob.	0.9
Reproduction prob.	0.05
Functions set	$+, *, \sqrt{}$

Table 2
Population size and number of generations of GP in the OPF_{GP} method for each dataset.

	n_p	n_g
Coil-100	40	8
Corel-3906	100	10
ETH-80	100	10
MPEG7	100	10
MSRCORID	50	10
PASCAL	60	10

Table 3
Descriptors to be combined in each database.

Database	Descriptors
Coil-100	SID, LBP and Color Bitmap
Corel	ACC, BIC, JAC, LAS and SASI
ETH-80	ACC, BIC, LAS, MSF and TSDIZ
MPEG7	Fourier, MSF and TSDIZ
MSRCORID	ACC, BIC, JAC, LAS and SASI
PASCAL	ACC, BIC, JAC, LAS and SASI

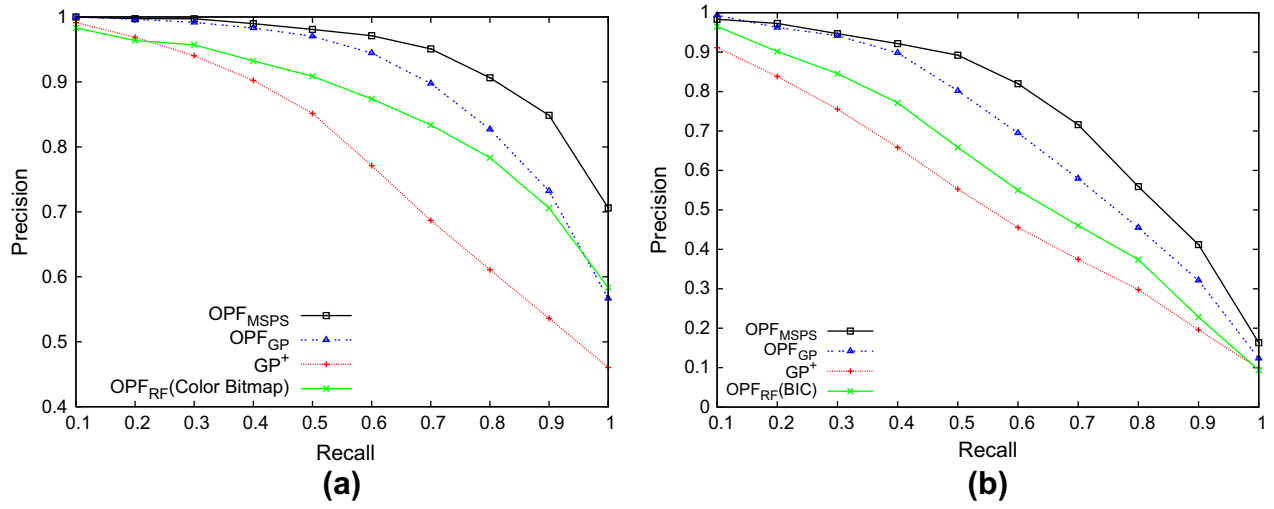

Fig. 6. Mean $P \times R$ curves in (a) Coil-100 and (b) Corel databases after 3rd iteration.

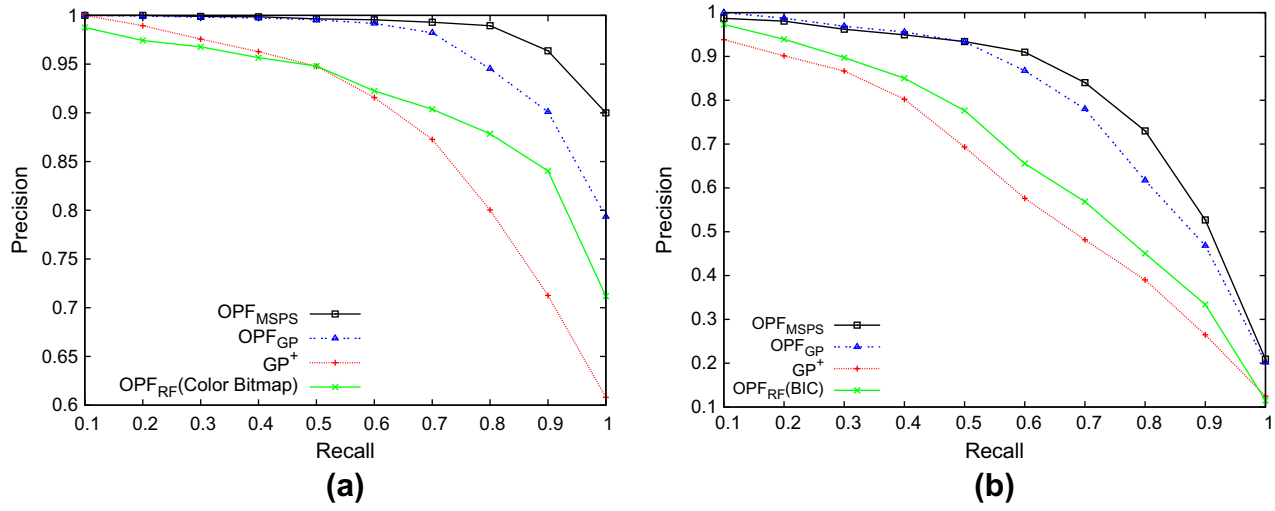

Fig. 7. Mean $P \times R$ curves in (a) Coil-100 and (b) Corel databases 5th iteration.

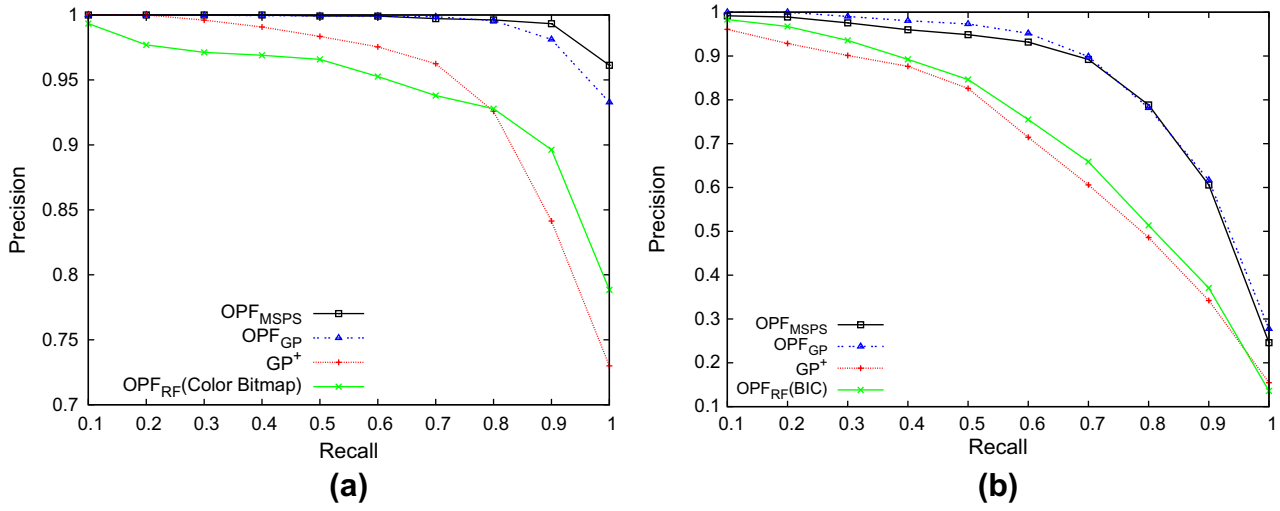

Fig. 8. Mean $P \times R$ curves in (a) Coil-100 and (b) Corel databases after 8th iteration.

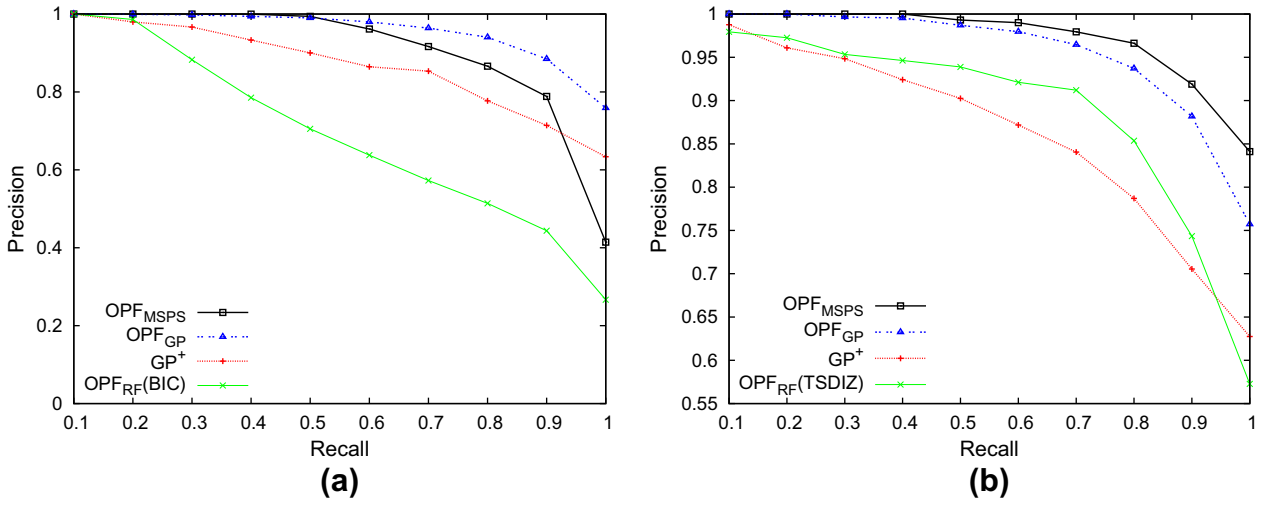

Fig. 9. Mean $P \times R$ curves in (a) ETH-80 and (b) MPEG7 databases after 3rd iteration.

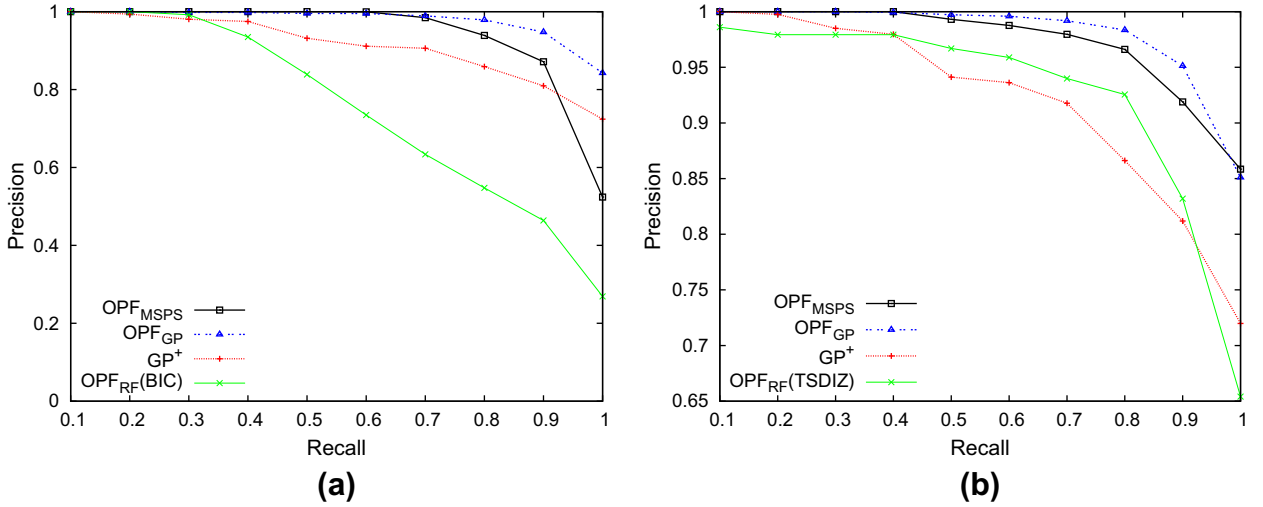

Fig. 10. Mean $P \times R$ curves in (a) ETH-80 and (b) MPEG7 databases after 5th iteration.

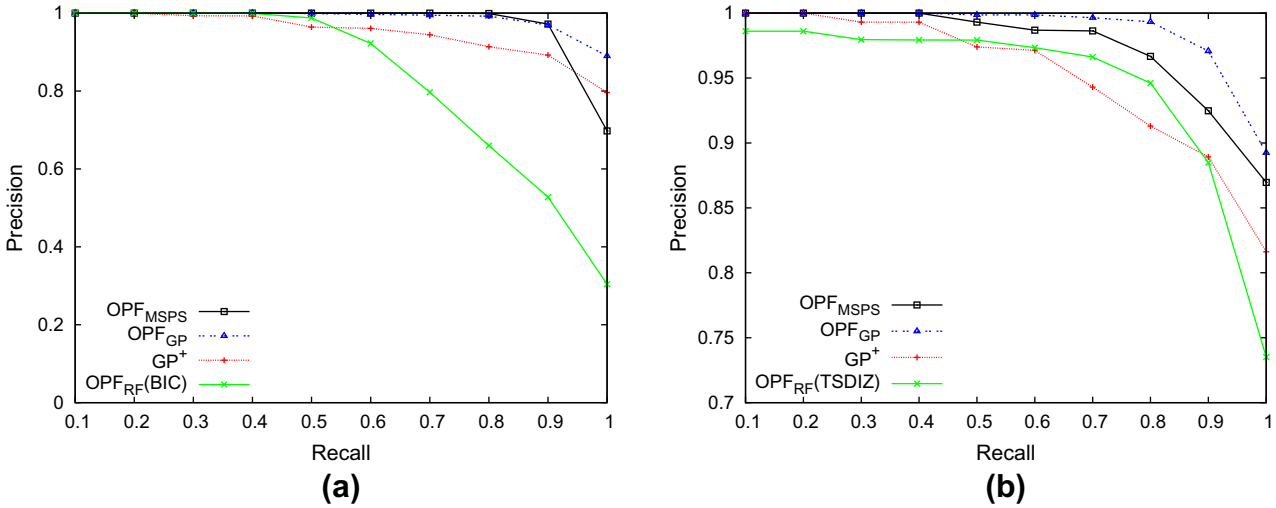

Fig. 11. Mean $P \times R$ curves in (a) ETH-80 and (b) MPEG7 databases after 8th iteration.

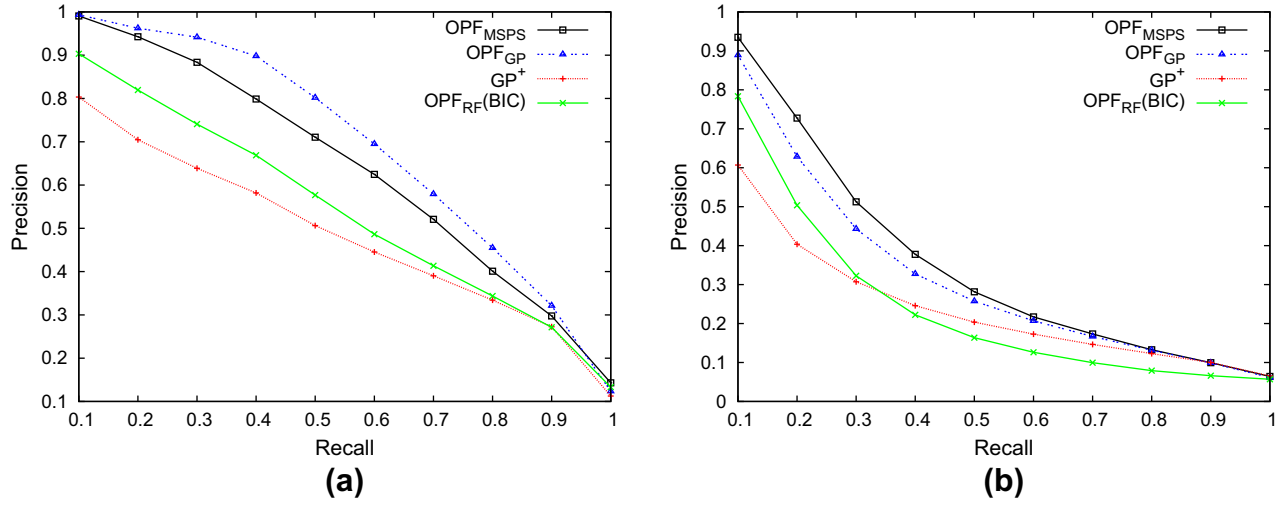

Fig. 12. Mean $P \times R$ curves in (a) MSRCORID and (b) PASCAL databases after 3rd iteration.

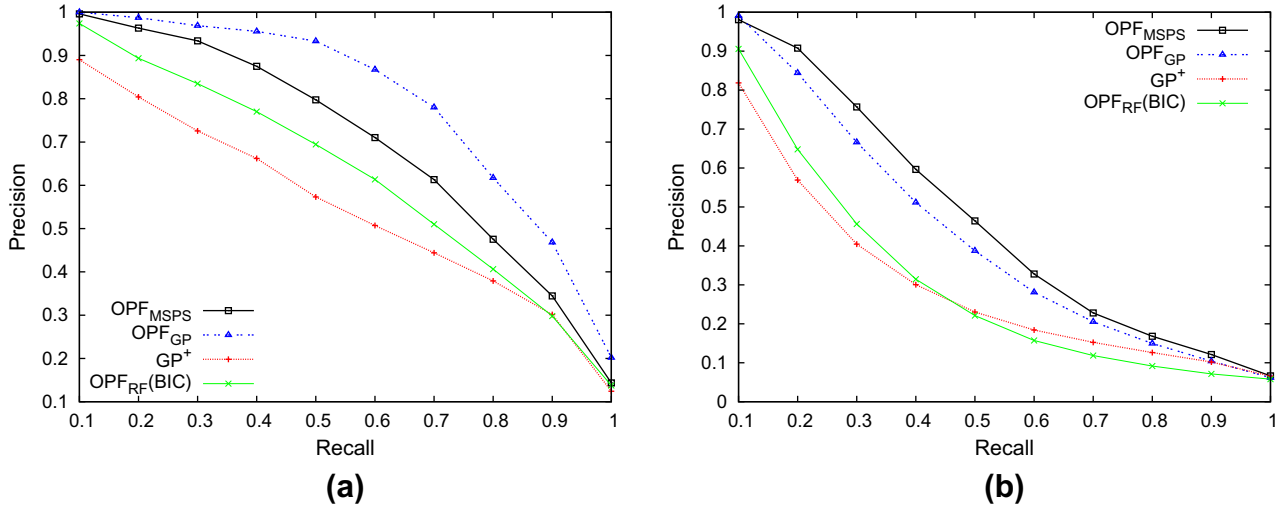

Fig. 13. Mean $P \times R$ curves in (a) MSRCORID and (b) PASCAL databases after 5th iteration.

images. GP^+ ranks the database images by a voting scheme among the best individuals of the last population. In this voting scheme, selected individuals vote for N candidate images. The most voted images are showed to the user. GP^+ outperforms several other feedback-based learning techniques [38,11,13,65,66] as well as the ranking process based on the best individual d^* obtained by Algorithm 4 with no OPF classification. In our experiments, the values for the GP parameters are the same as those used for the Corel Collection in [10].

We use the precision-recall ($P \times R$) curve for 3, 5, and 8 iterations to measure effectiveness. We use the $P \times R$ curve considering the results obtained at the last RF iteration. The higher the curve, the better the method. The first comparison evaluates the importance of having a composite descriptor while the second comparison evaluates the importance of having a pattern classifier.

We also present the percentage of relevant images retrieved to the user given a number of relevance feedback iterations of OPF_{MSPS}, OPF_{GP}, and GP^+. This curve allows us to evaluate how well the number of retrieved relevant images grows over iterations. Both curves show the average result considering all database images computed using the leave-one-out cross-validation. Since the curves use the entire image database $\mathcal{Z}$, Line 10 of Algorithm 2 is replaced by: *Return* $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{Y}$ *in their increasing order of* $\bar{d}(t, \mathcal{S}_R, \mathcal{S}_I)$.

The experiments used six heterogeneous image databases, representing different challenges for CBIR.

- Coil-100 [67]: It is an image database of 100 objects. Pictures of each object were taken in 72 different poses (total of 7,200 images).
- Corel [68]: This database is a collection with 200,000 images from the Corel Gallery Magic–Stock Photo Library 2. We use a subset of 3,906 samples, pre-classified into 85 classes. These classes have different number of images varying from 7 to 98 images each.
- ETH-80 [69]: This database is available in the project COGVIS. The project includes images of objects from 8 basic-level categories performing a total of 2384 images, distributed uniformly among the classes.
- MPEG7 (MPEG7 CE Shape-1 Part B) [70]: It is a database of 1,400 binary images of 70 shape categories, being 20 objects per category.
- MSRCORID [71]: It contains a set of 4,320 images grouped into 20 categories – about 36–652 per category. Most categories have about 200 images.

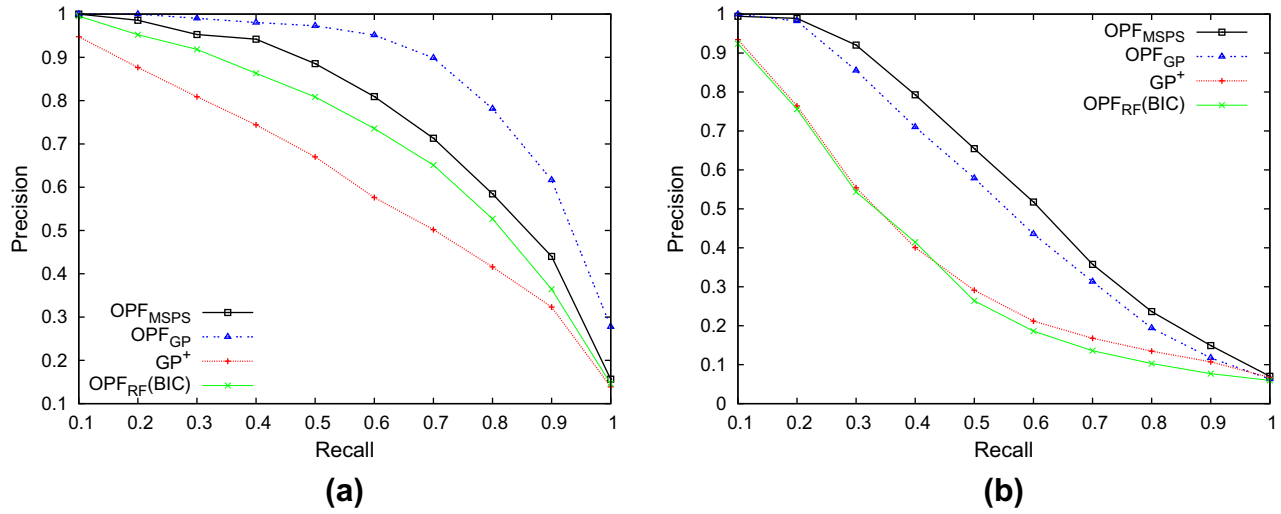

Fig. 14. Mean $P \times R$ curves in (a) MSRCORID and (b) PASCAL databases after 8th iteration.

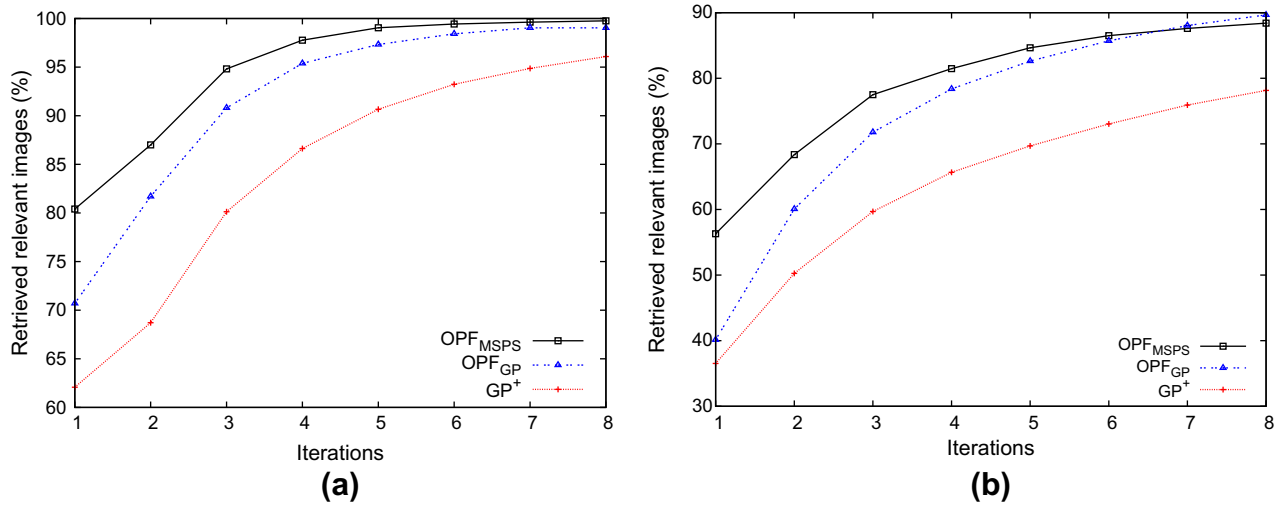

Fig. 15. *Rel* $\times$ *It* to (a) Coil-100 and (b) Corel databases from 1st to 8th iteration.

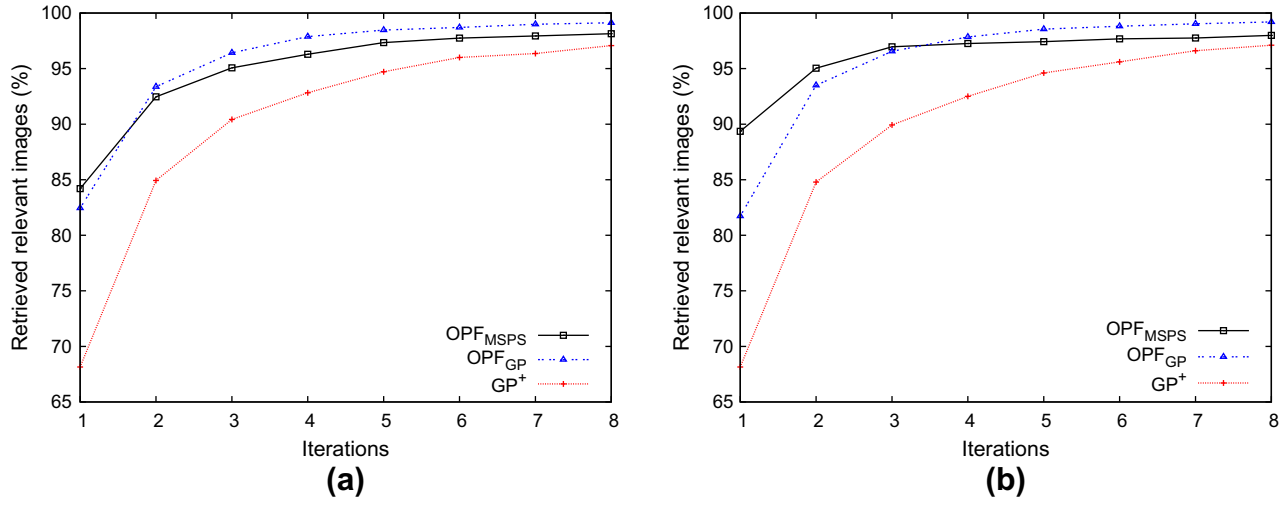

Fig. 16. *Rel* $\times$ *It* to (a) ETH-80 and (b) MPEG7 databases from 1st to 8th iteration.

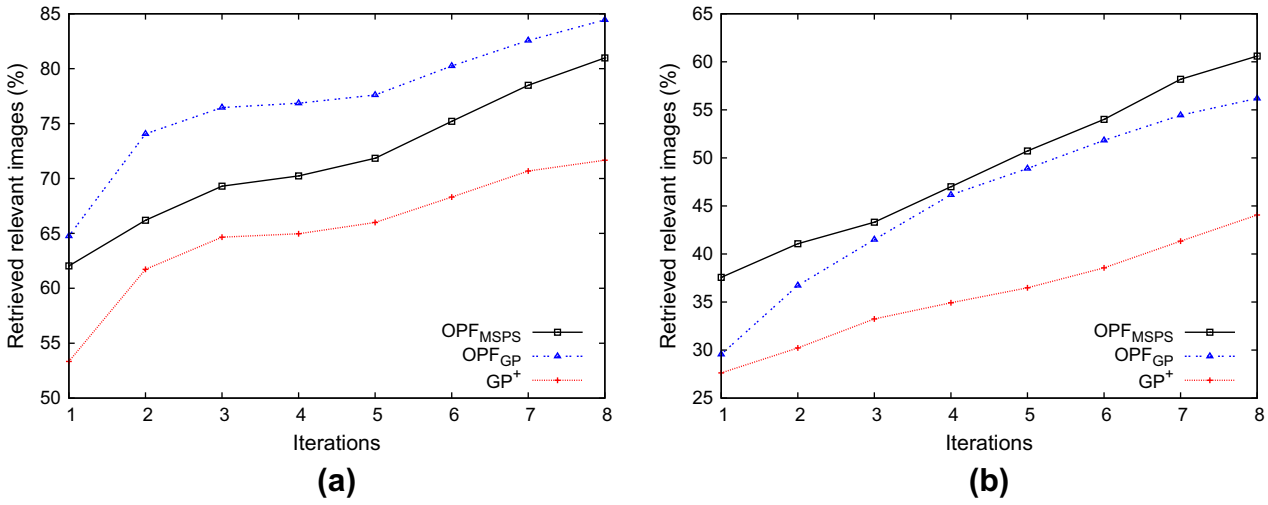

Fig. 17. *Rel* × *It* to (a) MSRCORID and (b) PASCAL databases from first to eighth iteration.

- PASCAL [72]: This database consists of images from Flickr.[3] We use a subset of 3448 grouped into 23 classes with different number of images, varying from 72 to 446 subimages each.

We use several image descriptors to test the methods. The results presented in this paper make use of the following descriptors: ACC [43], BIC [1], Color Bitmap [73], Fourier [74], JAC [3], LAS [44], LBP [75], MSF (Multi-Scale Fractal) [47], SASI [2], SID [4] and TSDIZ [5]. ACC, BIC, Color Bitmap and JAC are *color* descriptors, while LAS, LBP, SASI and SID are *texture* descriptors. Fourier, Multi-Scale Fractal and TSDIZ are used for databases with *shape* information.

For each image database, we chose the best simple descriptors for combination, as shown in Table 3 (except for the case of Coil-100, we chose the worst ones, because the problem was too easy with the best one – BIC [1]). For Corel, MSRCORID and PASCAL databases we selected three color descriptors and two texture descriptors. In ETH-80 we used shape, color, and texture descriptors. In MPEG7 we used all shape descriptors above.

Figs. 6–14 show the mean precision-recall curves of each method in all image databases for 3, 5 and 8 iterations of relevance feedback. These images show the evolution in the performance of each method. One may observe that both OPF_{MSPS} and OPF_{GP} outperformed GP^+ in all tested databases. OPF_{MSPS} outperformed OPF_{GP} in Coil-100 and PASCAL image databases while OPF_{GP} was more effective than OPF_{MSPS} in ETH-80 and MSRCORID databases. In Corel database, OPF_{MSPS} had better result up to the fifth iteration and OPF_{GP} beat OPF_{MSPS} after eight iterations. In MPEG7 shape database, OPF_{MSPS} had a good result for three iterations while OPF_{GP} outperforms OPF_{MSPS} from five iterations. In addition to that, OPF_{RF} using the best descriptor (TSDIZ for the MPEG7 database and BIC for the others) was more effective than GP^+ in most results, proving that the classification step is very important in the feedback-based learning process. Figs. 15–17 present the mean curves of relevant returned images per iteration (*Rel* × *It*) for all databases from the first to the eighth iteration, confirming the previous results.

6. Conclusion

We extended a recent feedback-based learning approach for CBIR using OPF classification (OPF_{RF}) to handle composite descriptors. The new methods, OPF_{MSPS} and OPF_{GP}, use optimization techniques, such as multi-scale parameter search (MSPS) and genetic programming (GP), to find the best combination function for a given set of simple descriptors at each iteration of relevance feedback.

Recent works [10,16] compare the baseline methods with the most effective approaches according to the state-of-the-art. Experiments with several datasets and descriptors have demonstrated that both OPF_{MSPS} and OPF_{GP} are very effective methods, with better performance than other state-of-the-art approaches, GP^+ and OPF_{RF}. In [16], however, we have demonstrated that our planned approach outperforms our greedy approach used in this work for simple descriptors. The extension of our planned approach for composite descriptors is not direct, given that it returns the most informative images (being many of them irrelevant images) per iteration, while the composite descriptors are being found based on the rank of the relevant images in the training set. This would require to investigate an effective optimization criterion to find composite descriptors based on sets with more irrelevant images than relevant ones. We understand that this is a subject for a future work.

The experiments show the importance of using multiple descriptors and pattern classifiers for CBIR. The best choice between OPF_{MSPS} and OPF_{GP} will depend on the dataset (application). However, it is important to note that MSPS assumes a fixed equation for the combination function and searches for its best parameters, while GP is more flexible in the choice of the best combination function. On the other hand, MSPS is faster than GP, and this favors to the choice of OPF_{MSPS}. The execution time of MSPS varies according to the number of descriptors to be combined. The number of scales has some influence, but tests have indicated that lower number of scales usually imply a higher number of iterations to achieve the best combination function.

A drawback in GP is its sensitivity to the initial parametrization (values of n_p and n_g for instance). This requires a previous study of the impact of each parameter for each dataset. Its efficiency depends on the number of generations, population size, training set size, and the size of the individuals [63].

The results also indicate that OPF_{GP} and OPF_{MSPS} require a few iterations of relevance feedback to retrieve the most relevant images.

Our future work includes a study of indexing schemes to provide scalability in large image databases with fast image access, the use of irrelevant images in the training set used by GP in the OPF_{GP} method, and the extension of our planned method [16] to

[3] www.flickr.com (As of 10/10/2011).

use composite descriptors. The OPF classifier is a multiple-class approach, as mentioned in Section 2, so the extension of our methods for multiple levels of relevance is straightforward and the efficacy gain may be higher. We also intend to investigate that.

Acknowledgments

Authors thank CAPES, CNPq (grants 140968/2007-5, 481556/2009-5, 303673/2010-9, and 306587/2009-2), and FAPESP (grants 2007/52015-0, 2008/57428-4, 2008/58528-2, and 2009/18438-7) for financial support.

References

[1] R.O. Stehling, M.A. Nascimento, A.X. Falcão, A compact and efficient image retrieval approach based on border/interior pixel classification, in: International Conference on Information and Knowledge Management, ACM, New York, NY, USA, 2002, pp. 102–109.
[2] A. ÇarkacIoglu, F. Yarman-Vural, SASI: a generic texture descriptor for image retrieval, Pattern Recog. 36 (11) (2003) 2615–2633.
[3] A. Williams, P. Yoon, Content-based image retrieval using joint correlograms, Multimedia Tools Appl. 34 (2) (2007) 239–248.
[4] J.A. Montoya-Zegarra, N.J. Leite, R. da S.Torres, Rotation-invariant and scale-invariant steerable pyramid decomposition for texture image retrieval, in: Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI), IEEE Computer Society, Washington, DC, USA, 2007, pp. 121–128.
[5] F.A. Andaló, P.A.V. Miranda, R.S. Torres, A.X. Falcão, Shape feature extraction and description based on tensor scale, Pattern Recog. 43 (1) (2010) 26–36.
[6] R. Torres, A. Falcão, M. Gonçalves, J. Papa, B. Zhang, W. Fan, E. Fox, A genetic programming framework for content-based image retrieval, Pattern Recog. 42 (2) (2009) 283–292.
[7] R. Li, J. Lu, Y. Zhang, T. Zhao, Dynamic adaboost learning with feature selection based on parallel genetic algorithm for image annotation, Knowledge-Based Syst. 23 (3) (2010) 195–201.
[8] C. Chen, Y. Yang, F. Nie, J.-M. Odobez, 3D human pose recovery from image by efficient visual feature selection, Computer Vision Image Understanding 115 (2011) 290–299.
[9] J.A. Santos, C.D. Ferreira, R.S. Torres, A genetic programming approach for relevance feedback in region-based image retrieval systems, in: Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI), IEEE Computer Society, 2008, pp. 155–162.
[10] C. Ferreira, J. dos Santos, R.daS. Torres, M. Gonçalves, R. Rezende, W. Fan, Relevance feedback based on genetic programming for image retrieval, Pattern Recog. Lett. 32 (1) (2011) 27–37.
[11] Y. Rui, T. Huang, Optimizing learning in image retrieval, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, 2000, pp. 236–243.
[12] D. Liu, K.A. Hua, K. Vu, N. Yu, Fast query point movement techniques for large CBIR systems, IEEE Trans. Knowl. Data Eng. 21 (5) (2009) 729–743.
[13] S. Tong, E. Chang, Support vector machine active learning for image retrieval, in: ACM international conference on Multimedia, ACM, New York, NY, USA, 2001, pp. 107–118.
[14] X. Wang, L. Yang, Application of SVM relevance feedback algorithms in image retrieval, in: International Symposium on Information Science and Engineering, IEEE Computer Society, Washington, DC, USA, 2008, pp. 210–213.
[15] X.-Y. Wang, J.-W. Chen, H.-Y. Yang, A new integrated SVM classifiers for relevance feedback content-based image retrieval using EM parameter estimation, Appl. Soft Comput. 11 (2) (2011) 2787–2804.
[16] A.T. Silva, A.X. Falcão, L.P. Magalhães, Active learning paradigms for CBIR systems based on optimum-path forest classification, Pattern Recog. 44 (12) (2011) 2971–2978.
[17] A.T. Silva, A.X. Falcão, L.P. Magalhães, A new CBIR approach based on relevance feedback and optimumpath forest classification, J. WSCG 18 (1–3) (2010) 73–80.
[18] G. Ruppert, F. Favretto, A. Falcão, C. Yassuda, F. Bergo, Fast and accurate image registration using the multiscale parametric space and grayscale watershed transform, in: Systems, Signals and Image Processing, IEEE Computer Society, Rio de Janeiro, 2010, pp. 17–19.
[19] J. Santos, C. Ferreira, R. Torres, M. Gonçalves, R. Lamparelli, A relevance feedback method based on genetic programming for classification of remote sensing images, Inform. Sci. 181 (13) (2011) 2671–2684.
[20] S. Philipp-Foliguet, J. Gony, P.H. Gosselin, FReBIR: an image retrieval system based on fuzzy region matching, Computer Vision Image Understanding 113 (6) (2009) 693–707.
[21] J.P. Papa, A.X. Falcão, C.T.N. Suzuki, Supervised pattern classification based on optimum-path forest, Int. J. Imaging Syst. Technol. 19 (2) (2009) 120–131.
[22] A. Falcão, J. Stolfi, R. Lotufo, The image foresting transform: theory, algorithms, and applications, IEEE Trans. Pattern Anal. Mach. Intel. 26 (1) (2004) 19–29.
[23] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, W. Equitz, Efficient and effective querying by image content, J. Intel. Inform. Syst. 3 (1994) 231–262.
[24] S. Mehrotra, Y. Rui, M. Ortega-Binderberger, T.S. Huang, Supporting content-based queries over images in MARS, in: IEEE International Conference on Multimedia Computing and Systems, 1997, pp. 632–633.
[25] I.J. Cox, M.L. Miller, T.P. Minka, T.V. Papathomas, P.N. Yianilos, The bayesian image retrieval system, PicHunter: theory, implementation, and psychophysical experiments, IEEE Trans. Image Process. 1 (9) (2000) 20–37.
[26] L. Boujnane, P. Bloore, TinEye, PixID, Piximilar: Advanced Image Software by Idée Inc., 2009. <http://www.tineye.com/>.
[27] I. Bartolini, P. Ciaccia, M. Patella, Query processing issues in region-based image databases, Knowl. Inform. Syst. 25 (2010) 389–420.
[28] J.J. Rocchio, Relevance Feedback in Information Retrieval, Prentice-Hall, Englewood Cliffs, NJ, USA, 1971. pp. 313–323.
[29] V.V. Raghavan, G.S. Jung, P. Bollmann, A critical investigation of recall and precision as measures of retrieval system performance, ACM Trans. Inform. Syst. 7 (1989) 205–229.
[30] R. Datta, D. Joshi, J. Li, J.Z. Wang, Image retrieval: ideas, influences, and trends of the new age, ACM Comput. Surveys 40 (2) (2008) 1–60.
[31] J. Traina, C.A. Traina, C. Faloutsos, B. Seeger, Fast indexing and visualization of metric data sets using slim-trees, IEEE Trans. Knowl. Data Eng. 14 (2) (2002) 244–260.
[32] X. Qian, H.D. Tagare, Adapting indexing trees to data distribution in feature spaces, Computer Vision Image Understanding 114 (2010) 111–124.
[33] M.R. Vieira, C. Traina Jr., F.J.T. Chino, A.J.M. Traina, DBM-Tree: a dynamic metric access method sensitive to local density data, J. Inform. Data Manage. 1 (1) (2010) 111–128.
[34] H. Lejsek, F. Ásmundsson, B. Jónsson, L. Amsaleg, An efficient disk-based index for approximative search in very large high-dimensional collections, IEEE Trans. Pattern Anal. Mach. Intel. 31 (5) (2008) 869–883.
[35] E. Valle, M. Cord, S. Philipp-Foliguet, High-dimensional descriptor indexing for large multimedia databases, in: ACM conference on Information and knowledge management, ACM, New York, NY, USA, 2008, pp. 739–748.
[36] F. Ye, Z. Shi, Z. Shi, A comparative study of PCA, LDA and kernel LDA for image classification, in: International Symposium on Ubiquitous Virtual Reality, IEEE Computer Society, Washington, DC, USA, 2009, pp. 51–54.
[37] V. Castelli, A. Thomasian, C.-S. Li, CSVD: clustering and singular value decomposition for approximate similarity search in high-dimensional spaces, IEEE Trans. Knowl. Data Eng. 15 (2003) 671–685.
[38] Y. Rui, T.S. Huang, M. Ortega, S. Mehrotra, Relevance feedback: a power tool for interactive content-based image retrieval, IEEE Trans. Circuits Syst. Video Technol. 8 (5) (1998) 644–655.
[39] A. Vadivel, A. Majumdar, S. Sural, Characteristics of weighted feature vector in content-based image retrieval applications, Intel. Sensing Inform. Process. 18 (1) (2004) 127–132.
[40] T. Tuytelaars, K. Mikolajczyk, Local invariant feature detectors: a survey, Found. Trends Comput. Graphics Vision 3 (3) (2008) 177–280.
[41] H. Bay, T. Tuytelaars, L.V. Gool, SURF: speeded up robust features, Computer Vision Image Understanding 110 (2008) 346–359.
[42] I. Laptev, B. Caputo, C. Schldt, T. Lindeberg, Local velocity-adapted motion events for spatio-temporal recognition, Computer Vision Image Understanding 108 (3) (2007) 207–229 (special issue on spatiotemporal coherence for visual motion analysis).
[43] J. Huang, S.R. Kumar, M. Mitra, W.-J. Zhu, R. Zabih, Image indexing using color correlograms, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997, pp. 762–768.
[44] B. Tao, B.W. Dickinson, Texture recognition and image retrieval using gradient indexing, J. Visual Commun. Image Represent. 11 (3) (2000) 327–342.
[45] K. Porkaew, K. Chakrabarti, S. Mehrotra, Query refinement for multimedia similarity retrieval in MARS, in: ACM Multimedia, 1999, pp. 235–238.
[46] Z. hua Zhou, K. jia Chen, H. bin Dai, Enhancing relevance feedback in image retrieval using unlabeled data, ACM Trans. Inform. Syst. 24 (2006) 219–244.
[47] R.S. Torres, A.X. Falcão, L.F. Costa, A graph-based approach for multiscale shape analysis, Pattern Recog. 37 (6) (2004) 1163–1174.
[48] R. Dorairaj, K. Namuduri, Compact combination of MPEG-7 color and texture descriptors for image retrieval, in: Conference Record of the Thirty-Eighth Asilomar Conference on Signals, vol. 1 (38), 2004, pp. 387–391.
[49] M. Arevalillo-Herráez, F.J. Ferri, J. Domingo, A naive relevance feedback model for content-based image retrieval using multiple similarity measures, Pattern Recog. 43 (3) (2010) 619–629.
[50] M. Broilo, F. De Natale, A stochastic approach to image retrieval using relevance feedback and particle swarm optimization, IEEE Trans. Multimedia 12 (4) (2010) 267–277.
[51] C.-H. Lee, M.-F. Lin, Ego-similarity measurement for relevance feedback, Expert Syst. Appl. 37 (1) (2010) 871–877.
[52] H. Nezamabadi-pour, E. Kabir, Concept learning by fuzzy k-NN classification and relevance feedback for efficient image retrieval, Expert Syst. Appl. 36 (3, Part 2) (2009) 5948–5954.
[53] G. Giacinto, F. Roli, Bayesian relevance feedback for content-based image retrieval, Pattern Recog. 37 (7) (2004) 1499–1508.
[54] R. Min, H. Cheng, Effective image retrieval using dominant color descriptor and fuzzy support vector machine, Pattern Recog. 42 (1) (2009) 147–157.
[55] J. Zhang, L. Ye, Local aggregation function learning based on support vector machines, Signal Process. 89 (11) (2009) 2291–2295.
[56] Z. Ye, J.X. Huang, H. Lin, Incorporating rich features to boost information retrieval performance: a SVM-regression based re-ranking approach, Expert Syst. Appl. 38 (6) (2011) 7569–7574.

[57] C.-H. Lin, R.-T. Chen, Y.-K. Chan, A smart content-based image retrieval system based on color and texture feature, Image Vision Computing 27 (2009) 658–665.
[58] S.D. MacArthur, C.E. Brodley, A.C. Kak, L.S. Broderick, Interactive content-based image retrieval using relevance feedback, Computer Vision Image Understanding 88 (2002) 55–75.
[59] J. Peng, Multi-class relevance feedback content-based image retrieval, Computer Vision Image Understanding 90 (2003) 42–67.
[60] R.S. Torres, A.X. Falcão, Content-based image retrieval: theory and applications, Rev. Inform. Teórica Aplicada 13 (2) (2006) 161–185.
[61] T. Cormen, C. Leiserson, R. Rivest, Introduction to Algorithms, The MIT Press, 1990.
[62] J.P. Papa, F.A.M. Cappabianco, A.X. Falcão, Optimizing optimum-path forest classification for huge datasets, in: International Conference on Pattern Recognition, 2010.
[63] W. Fan, E.A. Fox, P. Pathak, H. Wu, The effects of fitness functions on genetic programming-based ranking discovery for web search, J. Am. Soc. Inform. Sci. Technol. 55 (2004) 2004.
[64] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems), first ed., The MIT Press, 1992.
[65] N. Doulamis, A. Doulamis, Evaluation of relevance feedback schemes in content-based in retrieval systems, Signal Process.: Image Commun. 21 (4) (2006) 334–357.
[66] R. Min, H.D. Cheng, Effective image retrieval using dominant color descriptor and fuzzy support vector machine, Pattern Recog. 42 (1) (2009) 147–157.
[67] S.A. Nene, S.K. Nayar, H. Murase, Columbia University Image Library (Coil-100). <http://www1.cs.columbia.edu/CAVE/software/softlib/coil-100.php>.
[68] J.Z. Wang, J. Li, G. Wiederhold, Simplicity: semantics-sensitive integrated matching for picture libraries, IEEE Trans. Pattern Anal. Mach. Intel. 23 (2001) 947–963.
[69] B. Leibe, B. Schiele, Analyzing appearance and contour based methods for object categorization, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2003, pp. 409–415.
[70] T. Sikora, The MPEG-7 visual standard for content description – an overview, IEEE Trans. Circuits Syst. Video Technol. 11 (6) (2001) 696–702.
[71] Microsoft Research Cambridge, Object Recognition Image Database 1.0. <http://research.microsoft.com/vision/cambridge/recognition/>.
[72] M. Everingham, L.V. Gool, C.K.I. Williams, J. Winn, A. Zisserman, The PASCAL Visual Object Classes Challenge 2010 (voc2010). <http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2010/index.html>.
[73] T.-C. Lu, C.-C. Chang, Color image retrieval technique based on color features and image bitmap, Inform. Process. Manage. 43 (2) (2007) 461–472 (special issue on AIRS2005: information retrieval research in Asia).
[74] D. Zhang, G. Lu, A comparative study on shape retrieval using fourier descriptors with different shape signatures, J. Visual Commun. Image Represent. 1 (14) (2003) 41–60.
[75] V. Takala, T. Ahonen, M. Pietikäinen, Block-based methods for image retrieval using local binary patterns, in: Scandinavian Conference on Image Analysis (SCIA), 2005, pp. 882–891.