Relevance Feedback based on Genetic Programming for Image Retrieval

C. D. Ferreira\textsuperscript{a}, J. A. Santos\textsuperscript{a}, R. da S. Torres\textsuperscript{a,\textasteriskcentered}, M. A. Gonçalves\textsuperscript{b} R. C. Rezende\textsuperscript{a}, Weiguo Fan\textsuperscript{c}

\textsuperscript{a}Institute of Computing, University of Campinas, Campinas, SP, 13083-970, Brazil

\textsuperscript{b}Department of Computer Science, Federal University of Minas Gerais, Belo Horizonte, MG, 31270-010, Brazil.

\textsuperscript{c}Virginia Polytechnic Institute and State University Blacksburg, VA, 24061-0101, USA

Abstract

This paper presents two content-based image retrieval frameworks with relevance feedback based on genetic programming. The first framework exploits only the user indication of relevant images. The second one considers not only the relevant but also the images indicated as non-relevant.

Several experiments were conducted to validate the proposed frameworks. These experiments employed three different image databases and color, shape, and texture descriptors to represent the content of database images. The proposed frameworks were compared, and outperformed, five other relevance feedback methods regarding their effectiveness and efficiency in image retrieval tasks.

Key words: relevance feedback, content-based image retrieval, genetic programming
1 Introduction

Large image collections have been created and managed in several applications, such as digital libraries, medicine, and biodiversity information systems [10]. Given the large size of these collections, it is essential to provide efficient and effective mechanisms to retrieve images.

This is the objective of the so-called content-based image retrieval (CBIR) systems [29]. In these systems, the searching process consists of, for a given query image, finding the most similar images stored in the database. The searching process relies on the use of image descriptors. A descriptor can be characterized by two functions: feature vector extraction and similarity computation. The feature vectors encode image properties, like color, texture, and shape. The similarity between two images is computed as a function of the distance between their feature vectors.

Usually, different descriptors are statically combined [38,9], that is, the descriptor composition is fixed and used to process all queries submitted to the retrieval system. Nevertheless, different people may have distinct visual perceptions of a same image. Therefore, a static combination of descriptors may not characterize properly this diversity. Furthermore, it is not easy for a user to map his/her visual perception of an image into low-level features such as color and shape (“semantic gap”). Motivated by these limitations, relevance feedback (RF) approaches were incorporated into CBIR systems [31,26,5,27,13].

* Corresponding author. Tel/fax: +55 19 3521-5887/+55 19 3521-5847
Email addresses: cristiano.ferreira@students.ic.unicamp.br (C. D. Ferreira), jefersson.santos@students.ic.unicamp.br (J. A. Santos), rtorres@ic.unicamp.br (R. da S. Torres), mgoncalv@dcc.ufmg.br (M. A. Gonçalves), rcrezende@gmail.com (R. C. Rezende), wfan@vt.edu (Weiguo Fan).
Basically, the image retrieval process with relevance feedback is comprised of four steps: (i) showing a small number of retrieved images to the user; (ii) user indication of relevant and non-relevant images; (iii) learning the user needs by taking into account his/her feedbacks; (iv) and selecting a new set of images to be shown. This procedure is repeated until a satisfactory result is reached.

An important element of a relevance feedback technique is the learning process. Several relevance feedback methods designed for CBIR systems implement the learning of the user needs by assigning different weights to the descriptors used in the searching process [31,30,15]. This strategy allows only a linear combination of the similarity values defined by each descriptor. However, more complex combination functions may be necessary to express specific user visual perceptions.

Another common drawback of existing RF methods is concerned with the fact that they, in general, ignore the similarity function defined for each available descriptor. In some RF approaches the learning process is based only on the image feature vectors [36,5]. Others define specific distance functions for computing the similarity between two images [30,15]. In both cases, the overall CBIR system effectiveness may decrease if the similarity functions of the descriptors are not used. In fact, the effectiveness of a descriptor depends not only on the feature vector codification, but also on the defined similarity function.

In this paper two new relevance feedback methods for interactive image search are proposed. These methods adopt a genetic programming approach to learn user preferences in a query session. Genetic programming (GP) [22] is a Machine Learning technique used in many applications, such as data mining, signal processing, and regression [4,18,41]. This technique is based on the
evolution theory to find near optimal solutions. It is a kind of evolutionary algorithm [3] which is distinguished from the others mainly by the individual representation. The use of GP is motivated in this work by the previous success of using this technique in information retrieval [18,12] and CBIR [9] tasks.

The main contribution of this paper is the proposal of new RF frameworks that use GP to find a function that combines non-linearly similarity values computed by different descriptors. Furthermore, in our approach, the similarity functions defined for each available descriptor are used to compute the overall similarity between two images.

The effectiveness and efficiency of the proposed methods are compared with other relevance feedback techniques [31,30,36,15,28] for image retrieval tasks. Experiments conducted considering three different image collections and the use of color, texture, and shape descriptors demonstrate that the proposed frameworks are effective and efficient for CBIR, outperforming the (state-of-the-art) baselines.

This paper differs from the papers published in [19,14] with regard the following aspects: (i) the relevance feedback methods were simplified; (ii) the related work section was improved; (iii) more experiments aiming to compare the proposed RF approaches with a recently proposed method [28] were conducted. It is worth mentioning that the relevance feedback method proposed in [14] was used as one of the baselines in our experiments.

This paper is organized as follows. Section 2 discusses related work. Section 3 describes the CBIR model used (Section 3.1) and gives a brief overview of the Genetic Programming basic concepts (Section 3.2). Section 4 details the
GP-based frameworks proposed in this paper. Experimental design and results are reported in Sections 5 and 6, respectively. Finally, conclusions and future work are discussed in Section 7.

2 Related work

Relevance feedback (RF) [42,27,11] is a technique initially proposed for document retrieval that has been used with great success for human-computer interaction in CBIR. RF addresses two questions referring to the CBIR process. The first one is the semantic gap between high-level visual properties of images and low-level features used to describe them. Usually, it is not easy for a user to map her visual perception of an image into low-level features as color, texture, and shape. Another issue is concerned with the subjectivity of the image perception. Different people can have distinct visual perceptions of a same image.

One of the first relevance feedback-based CBIR methods was proposed in [31]. In this work, the learning process is based on assigning weights to each descriptor (interweight), and also to each feature vector bin, that is, to each position in this vector (intraweight). The learning algorithm heuristically estimates the weight values that best encodes the user needs in the retrieval process. In [30], the weight assignment is again employed. However, an optimization framework is applied to estimate the weights. This framework is based on the minimization of the Generalized Euclidean distance. Furthermore, this technique uses the query point movement approach, which tries to estimate the query pattern feature vector that best represents the user perception. These methods [31,30] are usually used as baselines in experiments to validate RF
Doulamis et al. [15] also employ an optimization framework to assess intraweights. However, instead of using the Generalized Euclidean distance, their method aims to maximize the *normalized cross-correlation* between the query image and each relevant image found during the retrieval process. The authors argue that the invariance regarding feature vector scaling/translation makes the normalized cross-correlation criterion more reliable than Euclidean distance-based similarity metrics for image retrieval tasks.

Another pioneer work in the area is the *PicHunter* system, presented in [6]. The PicHunter uses a Bayesian framework in the learning process. This mechanism assigns a probability to each database image and tries to predict the one closer to the user needs. In [16], another approach for *relevance feedback* using Bayesian inference is proposed: the *rich get richer (RGR)*. This method considers the consistency among successive feedbacks provided by the user in the learning process. Ves et al. [13] model the user perception as a multivariate normal distribution and estimate these distribution parameters based on user feedbacks. In [25], Leon et al. move away from the Bayesian paradigm and try to assess the relevance probability of a image by using logistic regression analysis. This approach constructs logistic regression models considering sets of features semantically related. The relevance probability produced by each model is grouped using ordered weighted averaging operators.

Herráez et al. [1] try to learn the user perception by defining a *fuzzy set* of images in which the user are interested. The feedback information is employed by a membership function to assign the degree of pertinence of each database image in this set. At the end of each iteration the images are ranked according to this membership value.
Another learning technique commonly used in RF is **Support Vector Machine (SVM)**. Basically, the goal of SVM-based methods is to find a hyperplane which separates the relevant from the non-relevant images in a high dimensional feature space. In [36], Tong et al. propose the use of a *support vector machine active learning* method to separate relevant images from the others. On each iteration, the images closer to the separation hyperplane, the most ambiguous ones, are displayed to the user. At the end of the process, the most (positive) distant images from hyperplane are shown.

Liu et al. [26] propose a different procedure to select the images that are displayed to the user. They argue there may exist some redundancy among the most ambiguous images and it can affect negatively the retrieval process. The proposed selection process [26] uses an unsupervised learning process to cluster the images near to the classification boundary and select one image from each cluster to be labeled. Another feature of the method proposed in [26] is the use of a co-training framework to add unlabeled images to the training set.

One of the difficulties which the relevance feedback must deal is the asymmetry of the training set, since the number of relevant images is usually smaller than the number of non-relevant ones. Min et al. [28] argue that, in SVM-based techniques, the decision hyperplane can be biased if the relevant and non-relevant images are treated equally during the training process. To overcome this problem, they employ *Fuzzy SVM*, associating a significance degree to each training set image to balance the bias caused by the asymmetry problem.

A genetic algorithm (GA)-based relevance feedback method is proposed in [33]. The database images are uniformly partitioned and GA and relevance feedback are used to determine the feature that best describes each region. Wang
et al. [39] employ Interactive Genetic Algorithms (IGA) and SVM to learn the user perception. IGA rely on the evaluation of individual accomplished by humans. SVM is used to learn a classifier from the user feedback. The unlabeled images classified as most relevant by the SVM classifier are added to the IGA training set. By using this unlabeled data, it is expected to speed up method convergence, decreasing the user fatigue due to the IGA learning process.

In the aforementioned methods, basically, the learning process relies on either assigning weights to similarity values determined by different descriptors or finding a function to compute the relevance degree of each image.

The first group allows only a linear combination of the similarity values. However, a more complex non-linear combination among the similarity values may be necessary to express the user needs.

The second group ignores the similarity function defined for the used descriptors. Nevertheless, the effectiveness of a descriptor depends not only on the feature vector codification, but also on the similarity function defined. While the effectiveness of SVM-based methods relies on the discriminative power of the feature vectors, other approaches define specific similarity functions (e.g., Generalized Euclidean distance [30]) to be employed in the searching process.

In the methods proposed in this paper, the similarity functions defined for all available descriptors are used. Furthermore, the proposed GP frameworks allow a more complex combination of the similarity values than linear combination. Genetic programming is used to obtain this combination function.

A GP-based RF method ($GP_{LSP}$) that exploits the content of image regions is proposed in [14]. In the $GP_{LSP}$ method, the similarity between images is expressed as a combination of regions’ similarities. $GP_{LSP}$, using negative and
positive feedbacks, is used in our experiments as a baseline.

The methods proposed in this paper differ from the GP-based RF approaches proposed in [19] with regard to the following aspects: the definition of the training set, the definition of the query pattern, and the fitness computation process. However, the overall effectiveness has not changed. Therefore, the methods proposed in [19] are not used as baselines.

GP has also been used in Information Retrieval tasks [23,18,12]. In [23], a GP-based framework is proposed to associate advertisement to Web pages based on these page contents. In [18] and [12] GP-based frameworks to obtain ranking functions for document retrieval are presented.

In [9], a GP framework is used in the CBIR domain. The objective of this framework is to find a function which combines distance values calculated from different descriptors. This approach does not use relevance feedback.

3 Background

This section presents the CBIR model adopted in our work and a brief overview of the Genetic Programming basic concepts.

3.1 CBIR model

This paper uses the CBIR model proposed in [10], described in the following.

A descriptor $D$ is defined as a pair $(\epsilon_D, \delta_D)$, where $\epsilon_D : \hat{I} \rightarrow \mathbb{R}^n$ is a function, which extracts a feature vector $\vec{v}_i$ from an image $\hat{I}$, and $\delta_D : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is a similarity function (e.g., based on a distance metric) that computes the
similarity between two images as a function of the distance between their corresponding feature vectors. A feature vector $\mathbf{v}_I$ of an image $\hat{I}$ is a point in $\mathbb{R}^n$ space: $\mathbf{v}_I = (v_1, v_2, \ldots, v_n)$, where $n$ is the dimension of the vector.

A composite descriptor $\hat{D}$ is a pair $(\mathcal{D}, \delta_D)$, where $\mathcal{D} = \{D_1, D_2, \ldots, D_k\}$ is a set of $k$ predefined simple descriptors, and $\delta_D$ is a similarity combination function which combines the similarity values $d_i$ obtained from each descriptor $D_i \in \mathcal{D}$, $i = 1, 2, \ldots, k$.

3.2 Genetic Programming

Genetic programming (GP) [22], as well as other evolutionary computation algorithms, is an artificial intelligence problem-solving technique based on the principles of biological inheritance and evolution. In GP approach, the individuals represent programs that undergo evolution. The fitness evaluation consists in executing these programs, and measuring their degrees of evolution. Genetic programming, then, involves an evolution-directed search in the space of possible computer programs that best solve a given problem.

**Algorithm 1** Basic GP evolution algorithm for $N$ generations.

1. Generate a initial population of individuals
2. for $N$ generations do
   3. Calculate the fitness of each individual
   4. Select the individuals to genetic operations
   5. Apply reproduction
   6. Apply crossover
   7. Apply mutation
3. end for

A basic evolution algorithm used in genetic programming is described in Algorithm 1. At beginning of the evolution, an initial population of individuals is created (line 1). Next, a loop of successive steps is performed to evolve these individuals: the fitness calculation of each individual (line 3), the selection
of the individuals based on their fitness (line 4), to breed a new population by applying genetic operators (lines 5–7). In the following, these steps are presented in more details.

Usually, a GP individual represents a program and is encoded in a tree. In this encoding, an individual contains two kinds of nodes, *terminals* (leaf nodes) and *functions* (intern nodes). Terminals are usually program inputs, although they may also be constants. Functions take inputs and produce outputs. A function input can be either a terminal or the output of another function.

The fitness of an individual is determined by its effectiveness in producing the correct outputs for all cases in a *training set*. The training set contains inputs and their previously known correspondent outputs.

To evolve the population, and optimize the desired objectives, it is necessary to choose the correct individuals to be subject to genetic operators. Thus, *selection operators* are employed to select the individuals, usually, based on their fitness. Examples of selection method are *roulette wheel*, *tournament*, and *rank-based selections* [3].

Genetic operators introduce variability in the individuals and make evolution possible, which may produce better individuals in posterior generations. The *crossover* operator exchanges sub-trees from a pair of individuals, generating two others. *Mutation* operator replaces a randomly chosen sub-tree from an individual by a sub-tree randomly generated. The *reproduction* operator simply copies individuals and inserts them in the next generation.
4 GP-based relevance feedback frameworks

This section presents two novel frameworks for RF in CBIR systems based on genetic programming. The first framework, $GP^+$, incorporates only the user indication of relevant (positive) images into the query pattern. The second one, $GP^\pm$, considers not only the relevant but also the non-relevant (negative) images indicated by the user.

4.1 $GP^+$ Framework

Let $\hat{D} = (D, \delta_D)$ be a composite descriptor (see Section 3.1) employed to rank $N$ database images defined as $DB = \{db_1, db_2, \ldots, db_N\}$. The set of $K$ simple descriptors of $\hat{D}$ is represented by $D = \{D_1, D_2, \ldots, D_K\}$. The similarity between two images $I_j$ and $I_k$, computed by $D_i$, is represented by $d_{i,I_j,I_k}$. All similarities $d_{i,I_j,I_k}$ are normalized between 0 and 1. A Gaussian normalization [31] can be employed for this task.

Let $L$ be the number of images displayed on each iteration. Let $IRR = \{irr_1, irr_2, \ldots, irr_{N_{IRR}}\}$ be the set that contains all images labeled as non-relevant during a retrieval session. Let $REL = \{rel_1, rel_2, \ldots, rel_{N_{REL}}\}$ be the set that contains the query image and all images labeled as relevant during a retrieval session.

Let $Q^+ = \{q_1, q_2, \ldots, q_M\}$ be the query pattern, where $M = \lceil N_{REL} \times \eta_{REL} \rceil$ and $0 < \eta_{REL} \leq 1$ is a constant (defined as 0.5 in our experiments). The query pattern images $q_i \in Q^+$ are randomly selected from the set of relevant images $REL$. 
Algorithm 2 presents an overview of the retrieval process proposed in this paper. The user interactions are indicated in italic. At the beginning of the retrieval process, the user indicates the query image $q$ (line 1). Based on this image, an initial set of images is selected to be shown to the user (line 2). Thus, the user is able to indicate the relevant images, from this initial set, starting the relevance feedback iterations. Each iteration involves the following steps: user indication of relevant and non-relevant images (line 4); the learning of the user preference by using GP (line 5); database images ranking (line 6); and the exhibition of the $L$ most similar images (line 7).

Algorithm 2 The proposed GP-based relevance feedback process.

1. User indication of query image $q$
2. Show the initial set of images
3. while the user is not satisfied do
   4. User indication of the relevant images
   5. Apply GP to find the best individuals (similarity composition functions) – see Algorithm 1
   6. Rank the database images
   7. Show the $L$ most similar images
4. end while

The selection of the initial image set, the use of GP to find the best similarity composition functions, and the algorithm to rank database images are presented in details in the following subsections.

4.1.1 Selecting the initial image set

The initial set of images showed to the user is defined by ranking the database images according to their similarity to the query image $q$. This process is performed in two steps. First, each simple descriptor $D_j \in \mathcal{D}$ is used to compute the similarity $d_{jqdb_i}$, where $db_i \in DB$ ($1 \leq i \leq N$). Next, the arithmetic mean is used to combine all these similarity values, that is $\delta_{MEAN}(q, db_i) = \frac{\sum_{j=1}^{K} d_{jqdb_i}}{K}$.

This combination uses all descriptors available and assigns the same degree of importance to all of them. This kind of combination allows the definition
of the initial set of images without previous knowledge about the descriptor effectiveness on the image database used.

Hence, the \( L \) first images are exhibited to the user. The user, then, identifies the set \( R = \{ R_1, R_2, \ldots, R_P \} \) of \( P \) relevant images. All images \( \{ R_i | R_i \notin REL \} \) are inserted into the relevant set \( REL \).

4.1.2 Finding the best similarity combinations – The GP framework

The goal of our learning mechanism is to find the similarity combination functions that best encode the user needs. We employ GP to find these combinations. As presented in Section 3.2, the GP technique requires the definition of several components, such as selection method, genetic operators, etc. Only two of these components require specific definition for the proposed learning process: the individual definition and the fitness computation. The other components used are presented in Section 3.2.

4.1.2.1 Individual definition In our method, each GP individual represents a candidate function \( \delta_D \), that is, a similarity combination function. This is encoded in a tree structure, as proposed in [9]. Intern nodes contain arithmetic operators. Leaf nodes have similarity values \( d_{i_{l_i} k} \), where \( 1 \leq i \leq K \). Figure 1 shows an example of an individual. The individual in this figure represents the function \( f(d_{1_{l_i} k}, d_{2_{l_i} k}, d_{3_{l_i} k}) = (d_{1_{l_i} k} + d_{3_{l_i} k}) \ast \sqrt{\frac{d_{2_{l_i} k}}{d_{1_{l_i} k}}} \). This figure considers the use of three distinct descriptors and the set of operators \( \{+, /, *, \sqrt{\cdot} \} \) as intern nodes.

This individual representation is very suitable for the proposed GP search engine, since it directly encodes the candidates for the \( \delta_D \) function.
Given an individual $\delta_i$ the similarity $Sim_{\delta_i}^+(Q^+, I)$ between the query pattern $Q^+$ and an image $I$ is defined in Equation 1.

$$Sim_{\delta_i}^+(Q^+, I) = \sum_{l=1}^{M} \delta_i(q_l, I)$$  

(1)

4.1.2.2 Individual fitness computation

The goal of the proposed fitness computation process is to assign the highest fitness values to the individuals that best encode the user preferences. In our approach, the fitness computation is based on the ranking of the training set images defined by each individual. Individuals that rank relevant images at the first positions must receive a high fitness value. The proposed fitness computation process is based on this objective criterion.

Training set definition. The fitness of an individual is computed based on its effectiveness in a training set. In the proposed method, the training set is defined as the following.

Definition 1 The **training set** is defined as a pair $T^+ = (T, r^+)$ where:

- $T = \{t_1, t_2, \ldots, t_{N_T}\}$ is a set of $N_T$ distinct training images.
- $r^+: T \rightarrow \mathbb{R}$ is a function that indicates the user feedback for each image in
For instance, \( r^+(t_i) \), where \( t_i \in T \), can be defined as 1 if \( t_i \) is relevant and 0, otherwise.

The set \( T \) of training images is comprised of:

- \( N_{TREL} = \min \left( \frac{N_T}{2}, N_{REL} \right) \) images labeled as relevant, randomly chosen from \( REL \);
- \( N_{TIRR} = \min (N_T - N_{TREL}, N_{IRR}) \) non-relevant images, randomly chosen from \( IRR \);
- \( N_{TUNL} = N_T - N_{TREL} - N_{TIRR} \) non-labeled images, randomly chosen from database.

The addition of non-labeled images into the training set aims to increase the training set size in the first iterations, when there are only few labeled images.

**Fitness computation.** The fitness of an individual \( \delta_i \) is computed based on the similarity between the query pattern and all images from the training set. The fitness computation process is divided into two phases, described in the following.

**Phase 1.** In the first phase, the training images \( t_k \in T \) are sorted according to their similarity with the query pattern \( Q^+ \), computed as defined in Equation 1.

**Phase 2.** Once the ranking is obtained, the second phase starts. The goal of this phase is to evaluate the ranked list \( rk_{\delta_i} \) generated in Phase 1. This evaluation consists in assigning high values to ranked lists, in which relevant images present in the training set are ranked at the first positions. This evaluation is accomplished by applying an evaluation function \( f(rk_{\delta_i}) \) that considers the rank position of the relevant images in \( rk_{\delta_i} \).
In our approach the function \( f(rk_\delta_i) \) follows the utility theory principles [20]. According to this theory, there is an utility function which assigns a value to an item, regarding the user preference. Usually, it assumes that the utility of an item decreases according to its position in a certain ranking [17]. Formally, given two items \( It_i \) and \( It_{i+1} \), where \( i \) is a position in a ranking, the following condition must be satisfied by a utility function \( U(x) \): \( U(It_i) > U(It_{i+1}) \). In this paper, each item is an image.

An example of \( f(rk_\delta_i) \) evaluation function is

\[
f(rk_\delta_i) = \sum_{l=1}^{L} r(rk_\delta_i[l]) \times \frac{1}{A} \times \left( \frac{A-1}{A} \right)^{i-1} \quad [17]
\]

where \( A \) is defined as 2, \( rk_\delta_i[l] \) is the \( l^{th} \) image in the ranking \( rk_\delta_i \), and \( r(rk_\delta_i) = 1 \), if \( rk_\delta_i[l] \) is relevant or \( r(rk_\delta_i) = 0 \), otherwise.

Hence, applying \( f(rk_\delta_i) \) to the ranking \( rk_\delta_i \) defines the fitness \( f_\delta_i \) of the individual \( \delta_i \).

Algorithm 3 summarizes the fitness computation process. Lines 3–7 refer to the first phase of the fitness computation: the ranked list definition. Phase 2, the ranking evaluation process, corresponds to line 8.

**Algorithm 3** The fitness computation process.

1. **Input**: individual \( \delta_i \), query pattern \( Q^+ \), training set \( T \).
2. **Output**: the fitness of the individual \( \delta_i \).
3. **for** all \( t_k \in T \) **do**
   4. \( rk_\delta_i[k].\text{key} \leftarrow \text{Sim}_1^+(Q^+, t_k) \)
   5. \( rk_\delta_i[k].\text{element} \leftarrow t_k \)
4. **end for**
6. Sort \( rk_\delta_i \)
7. \( f_\delta_i \leftarrow f(rk_\delta_i) \)
8. **return** \( f_\delta_i \)
4.1.3 Ranking database images

Once the fitness of the individuals is computed, it is possible to define the best individual that will be used to rank the database images. However, it is possible that more than one individual has a high fitness. Actually, if the query pattern size $M$ is small, there is a highly probability that many individuals have a good fitness. Our strategy tries to improve the database images ranking by combining the ranked lists obtained by using these “good” individuals. This combination is achieved by applying a voting scheme. Let $\delta_{\text{best}}$ be the best individual obtained from GP (see Section 4.1.2) in the current iteration. The set $S$ of individuals selected to vote is defined as $S = \{\delta_i | \frac{f_{\delta_i}}{f_{\delta_{\text{best}}}} \geq \alpha\}$ where $\alpha \in [0,1]$. The $\alpha$ value is called voting selection ratio threshold.

In the voting scheme, all selected individuals vote for $L$ candidate images. The most voted images are showed to the user. Algorithm 4 presents this process in details.

Algorithm 4 The voting scheme used to rank the database images.

```plaintext
\textbf{Input} : Set $S$ of selected individuals, database $DB$, query pattern $Q^+$, number of displayed images $L$.
\textbf{Output} : ranked list of images to be displayed.

1. for all $\delta_i \in S$ do
2. for all $db_j \in DB$ do
3. \hspace{1em} $rk_i[j].key \leftarrow Sim_{\delta_i}(Q^+, db_j)$
4. \hspace{1em} $rk_i[j].element \leftarrow db_j$
5. end for
6. end for
7. Sort $rk_i[j]$
8. for $j \leftarrow 1 \text{ to } \beta$ do
9. \hspace{1em} votes[$rk_i[j].element$] $\leftarrow$ votes[$rk_i[j].element$] + $1/j$
10. end for
11. Sort $DB$ images regarding their votes
12. return the $L$ most voted images
```

Firstly, the database images are sorted by using each selected individual $\delta_i$, regarding the similarity $Sim_{\delta_i}(Q^+, db_j)$, between each database image $db_j \in DB$ and the query pattern $Q^+$ (lines 3–8). This way, each selected individual $\delta_i$ defines a ranked list containing the database images.
Each image at the first $L$ positions in each ranking list receives a vote inversely proportional to its position (lines 9–11). For instance, the first image receives a vote equal to 1, the second, 1/2, the third, 1/3 and so on. Then, the database images are sorted according to the sum of their votes (line 13). Finally, the $L$ most voted images are selected to be shown to the user (line 14).

4.2 $GP^\pm$ Framework

The $GP^+$ framework presented in Section 4.1 uses only the information provided by the images labeled as relevant in the query pattern. $GP^\pm$ framework extends $GP^+$ by incorporating also the non-relevant images. Our hypothesis here is that the use of the information provided by non-relevant images can improve the retrieval results.

Recall that $REL = \{rel_1, rel_2, \ldots, rel_{N_{REL}}\}$ is the set containing the query image and all images labeled as relevant during a retrieval session and $IRR = \{irr_1, irr_2, \ldots, irr_{N_{IRR}}\}$ is the set containing all images labeled as non-relevant. The new query pattern $Q^\pm$ is defined as $Q^\pm = Q_{rel} \cup Q_{irr}$, where:

- $Q_{rel}$ is a subset randomly chosen from $REL$, containing $N_{Q_{rel}} = \lceil N_{REL} \times \eta_{REL} \rceil$ and $0 < \eta_{REL} \leq 1$ is a constant (defined as 0.5 in our experiments);
- $Q_{irr}$ is a subset randomly chosen from $IRR$, containing $N_{Q_{irr}} = \lceil N_{IRR} \times \eta_{IRR} \rceil$ and $0 < \eta_{IRR} \leq 1$ is a constant (defined as 0.5 in our experiments).

Since the function $Sim_{\delta_i}^+$ (Equation 1) considers only the relevant images of the query pattern it must be redefined in the $GP^\pm$ framework. The new function, $Sim_{\delta_i}^\pm$, is defined as:

19
\[
Sim_{\tilde{S}}^\pi(Q^\pi, I) = \frac{N_{\text{qirr}} \times \sum_{\forall q_{\text{rel}} \in Q_{\text{rel}}} \delta_i(q_{\text{rel}}, I)}{N_{\text{qrel}} \times \sum_{\forall q_{\text{irr}} \in Q_{\text{irr}}} \delta_i(q_{\text{irr}}, I)}
\]

In the \(GP^\pm\) framework, Equation 3 is employed to sort the training set images during the fitness computation process of the GP individuals, and to sort the database images.

5 Experiments

This section describes in details the experiments performed to validate our proposed frameworks.

5.1 Image Descriptors

The proposed frameworks were presented in a generic way, since there are no restrictions regarding descriptors that can be used to characterize the images. Color, shape, and texture based descriptors are the most common ones and were used in our experiments.

Table 1 shows the image descriptors used in the experiments. In this table, column \textit{Descriptors} refers to the descriptor name; \textit{Type} presents the type of the descriptor (color, texture, or shape); \textit{Dimension} refers to the size of feature vector extracted by each descriptor; and \textit{Distance function} defines the distance function employed by each descriptor (as they were originally proposed).
Table 1
Image descriptors used in our experiments.

<table>
<thead>
<tr>
<th>Descriptor</th>
<th>Type</th>
<th>Dimension</th>
<th>Distance function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color Histogram [35] (C1)</td>
<td>Color</td>
<td>256</td>
<td>L1</td>
</tr>
<tr>
<td>Color Moments [34] (C2)</td>
<td>Color</td>
<td>9</td>
<td>d_mom [34]</td>
</tr>
<tr>
<td>BIC [32] (C3)</td>
<td>Color</td>
<td>128</td>
<td>L1</td>
</tr>
<tr>
<td>Gabor Filters [24] (T1)</td>
<td>Texture</td>
<td>32</td>
<td>Euclidean</td>
</tr>
<tr>
<td>Spline [37] (T2)</td>
<td>Texture</td>
<td>26</td>
<td>Euclidean</td>
</tr>
<tr>
<td>Moment Invariants [21] (S1)</td>
<td>Shape</td>
<td>14</td>
<td>Euclidean</td>
</tr>
<tr>
<td>Fourier [21] (S2)</td>
<td>Shape</td>
<td>126</td>
<td>Euclidean</td>
</tr>
<tr>
<td>MS Fractal [8] (S3)</td>
<td>Shape</td>
<td>25</td>
<td>Euclidean</td>
</tr>
<tr>
<td>BAS [2] (S4)</td>
<td>Shape</td>
<td>180</td>
<td>OCS [40]</td>
</tr>
<tr>
<td>SS [7] (S5)</td>
<td>Shape</td>
<td>30</td>
<td>OCS [40]</td>
</tr>
</tbody>
</table>

5.2 Image Collections

Table 2 presents the image collections employed in our experiments. The first column refers to the collection names. The second shows the size of each collection, the number of classes, and the size of each class. Finally, the third column presents the descriptors employed on each image collection (see Table 1).

Table 2
Image collections used.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size(#classes/class sizes)</th>
<th>Descriptors</th>
</tr>
</thead>
<tbody>
<tr>
<td>FISH</td>
<td>11,000(1,100/10)</td>
<td>S1, S2, S3</td>
</tr>
<tr>
<td>MPEG7</td>
<td>1,400(70/20)</td>
<td>S1, S2, S3, S4, S5</td>
</tr>
<tr>
<td>COREL</td>
<td>3,306(57/7 – 98)</td>
<td>C1, C2, C3, T1, T2</td>
</tr>
</tbody>
</table>

The use of the FISH collection aims to evaluate the effectiveness of the proposed frameworks when the number of relevant images for a given query is very small.

The objective of the experiment with MPEG7 is to evaluate the impact of not using appropriate similarity functions for each available descriptor.

As mentioned in Section 2, $W_{D_{heu}}$ approach requires the use of a similarity function that allows the assignment of weights for each feature vector bin. We use the Weighted Euclidean Distance for computing the similarity of BAS and SS feature vectors, since the OCS matching algorithm does not support weight assignment.
The $WD_{opt}$ technique is based on the use of *Generalized Euclidean distance* for computing the similarity between two feature vectors. Similarly, $QS_{str}$ employs the *normalized cross correlation* in image distance computation. The SVM-based methods rely only on the use of feature vectors. No descriptor similarity function is used in the learning process.

The use of COREL collection aims to evaluate the proposed frameworks with regard to their effectiveness in retrieving colorful images from a heterogeneous collection, where the number of relevant images per class is not balanced (varying from 7 to 98). This collection represents, therefore, a real-world scenario for validating the proposed RF methods.

### 5.3 Baselines

The proposed frameworks were compared with other five RF methods: $WD_{heu}$ [31], $WD_{opt}$ [30], $QS_{str}$ [15], $SVM_{active}$ [36], and $SVM_{fuzzy}$ [28]. The first three methods [31,30,15] are based on weight assignment, while the last two [36,28] rely on the use of SVM to classify database images as relevant and non-relevant.

The methods $WD_{heu}$, $WD_{opt}$, and $SVM_{active}$ are usually used as baselines in experiments to validate RF techniques [15,30,5]. $QS_{str}$ and $SVM_{fuzzy}$ are more recent methods that were showed to yield better results than eleven other RF techniques [15,28]. It is worth mentioning that $QS_{str}$ was not compared with any SVM-based RF method. We also included the $GP_{LSP}$ [14] relevance feedback method as baseline in the experiments that considered the COREL collection.

These baseline methods are described in Section 2.
5.4 Effectiveness Measures

We use two different measures to evaluate the effectiveness of the proposed RF frameworks: Precision vs. Recall curve ($P \times R$) and Retrieved relevant images vs. number of iterations curve ($Rel \times It$). A paired Wilcoxon test is also used to evaluate the statistical significance of the results.

$P \times R$ curve is a common effectiveness evaluation criterion used in information retrieval systems that have been employed to evaluate CBIR systems. Precision $P(q)$ can be defined as the number of retrieved relevant images $Rel(q)$ over the total number of retrieved images $N(q)$ for a given query $q$, that is $P(q) = \frac{Rel(q)}{N(q)}$. Recall $R(q)$ is the number of retrieved relevant images $Rel(q)$ over the total number of relevant images $M(q)$ present in the database for a given query $q$, that is $R(q) = \frac{Rel(q)}{M(q)}$. We use the $P \times R$ curve considering the results obtained on the last RF iteration (e.g., 10th).

$Rel \times It$ curves are used to show the percentage of relevant images retrieved to the user given a number of RF iterations. This curve allows evaluating how the number of retrieved relevant images grows over iterations. For iteration zero, we consider the number of relevant images retrieved in the initial set (see Section 4.1.1).

The average $P \times R$ and $Rel \times It$ curves, considering the results for all query images are used to compare the RF approaches.

We also performed a paired Wilcoxon test comparing the proposed RF frameworks with all baselines. This test aims to verify if both $P \times R$ and $Rel \times It$ curves obtained by using the proposed framework in CBIR tasks are statistically (significant) different from all the others. We used a Wilcoxon test.
because the samples do not have a normal distribution.

6 Experiment Results

Two different experiments were conducted: the first one aims to determine the best GP parameters to be used in the RF frameworks (Section 6.1); the second compares the proposed methods with the baselines RF techniques with regard to their effectiveness and efficiency (Sections 6.2 and 6.3).

In our experiments, the presence of users is simulated. In this simulation, all images belonging to the same class of the query image are considered relevant. 10 iterations were considered for each query. Experiments also evaluated the effectiveness of the proposed methods when 20 images are showed to the user on each iteration.

6.1 GP Parameters

The implementation of the proposed framework requires the definition of several GP parameters as pointed out in Section 3.2. Examples of parameters include population size, maximum number of generations, genetic operators rate, etc.

We tested the combination of these parameters with regard to their effect in the $GP^+$ and $GP^\pm$ frameworks. We conducted experiments in the FISH and COREL collections. 550 and 85 randomly chosen images were used as query images for the FISH and COREL collections, respectively.

Tables 3 and 4 show the best values found for each GP parameter. The param-
eters chosen for the COREL collection were also employed in the experiments with the MPEG7 collection.

Table 3
Best Parameter Values for the $GP^+$ framework.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>FISH Collection</th>
<th>COREL Collection</th>
</tr>
</thead>
<tbody>
<tr>
<td>population size</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>number of generations</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>initial population</td>
<td>half-and-half</td>
<td>half-and-half</td>
</tr>
<tr>
<td>initial tree depth</td>
<td>2 – 5</td>
<td>2 – 5</td>
</tr>
<tr>
<td>maximum tree depth</td>
<td>15</td>
<td>5</td>
</tr>
<tr>
<td>selection method</td>
<td>tournament (size 2)</td>
<td>tournament (size 2)</td>
</tr>
<tr>
<td>crossover rate</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>mutation rate</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>training set size</td>
<td>55</td>
<td>80</td>
</tr>
<tr>
<td>voting selection ratio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>functions set</td>
<td>+, ∗, /(protected), √</td>
<td>+, ∗, /(protected), √</td>
</tr>
</tbody>
</table>

Table 4
Best Parameter Values for the $GP^\pm$ framework.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>FISH Collection</th>
<th>COREL Collection</th>
</tr>
</thead>
<tbody>
<tr>
<td>population size</td>
<td>30</td>
<td>60</td>
</tr>
<tr>
<td>number of generations</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>initial population</td>
<td>half-and-half</td>
<td>half-and-half</td>
</tr>
<tr>
<td>initial tree depth</td>
<td>2 – 5</td>
<td>2 – 5</td>
</tr>
<tr>
<td>maximum tree depth</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>selection method</td>
<td>tournament (size 2)</td>
<td>tournament (size 2)</td>
</tr>
<tr>
<td>crossover rate</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>mutation rate</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>training set size</td>
<td>130</td>
<td>80</td>
</tr>
<tr>
<td>voting selection ratio</td>
<td>0.999</td>
<td>1</td>
</tr>
<tr>
<td>functions set</td>
<td>+, ∗, √</td>
<td>+, ∗</td>
</tr>
</tbody>
</table>

6.2 Comparison with Baselines

Experiments aiming to compare the proposed frameworks with the baselines were conducted using the three data sets presented in Section 5.2. For the FISH collection, 1 image of each class was randomly chosen as query image. Therefore, experiments using this collection considered 1100 query images. 140 query images were used for the MPEG7 collection, which represents 2
randomly chosen from each available class. For the COREL collection, we also used 2 images per class, which represents 170 query images. Experiments were conducted on a 3.0GHz Pentium 4 with 2G RAM.

Figures 2, 3, and 4 show the experimental results for the FISH, MPEG7, and COREL collections, respectively. Those figures present the Retrieved relevant images vs. number of iterations (Rel × It) curves, the Precision vs. Recall (P×R) curves, and tables with Wilcoxon significance test. In the Wilcoxon test tables each cell indicates if there is significance statistical difference between the proposed frameworks – $GP^+$ (squares) and $GP^\pm$ (circles) – and each baseline ($WD_{heu}$, $WD_{opt}$, $SVM_{active}$, $QS_{str}$, $SVM_{fuzzy}$, and $GP_{LSP}$ – in the case of COREL collection) for a given recall/iteration value. The presence of squares indicates statistical difference – significance level ($\alpha$) less than 0.05.

As can be observed in Figure 2(a), the proposed frameworks present the best
result since the first iteration for the FISH collection. Note that, from the eighth iteration on, the number of retrieved images by our frameworks is close to 100%, that is, almost all relevant images are returned for all queries. Another important remark is concerned with the bad effectiveness behavior of the SVM_{active}, QS_{str}, and SVM_{fuzzy} approaches. We believe that this result is due to the fact that these methods were not able to learn from the small number of relevant images (10) for each query image.

From the Wilcoxon tests (Figure 2(c)), it can be observed that the difference of the Rel × It curves are significant for all iteration values.

The superiority of the proposed frameworks are also confirmed by the P × R curves on the last iteration (Figure 2(b)) and by the Wilcoxon test results (Figure 2(d)), considering recall values greater than 0.4.

![Fig. 3. Comparison between the proposed frameworks and baselines for MPEG7 collection: (a) Rel × It considering 20 images displayed per iteration; (b) P × R considering 20 images displayed per iteration; (c) Wilcoxon test for the data in Figure 3(a); (d) Wilcoxon test for the data in Figure 3(b);](image-url)

For the MPEG7 collection, the number of retrieved images for GP^+ and GP^±
is better than all baselines (see the Rel × It curves – Figure 3(a)). This was confirmed by the Wilcoxon test (Figure 3(c)).

For $P \times R$ curves (Figure 3(b)), all evaluated methods present similar results until a recall value equal to 0.6. From this point on, the proposed methods start yielding significant better results (Figure 3(d)).

![Graphs showing comparison between proposed frameworks and baselines for COREL collection.](image)

Fig. 4. Comparison between the proposed frameworks and baselines for COREL collection: (a) Rel × It considering 20 images displayed per iteration; (b) $P \times R$ considering 20 images displayed per iteration; (c) Wilcoxon test for the data in Figure 4(a); (d) Wilcoxon test for the data in Figure 4(b);

For the COREL collection, the proposed frameworks yield better results than $WD_{heu}$ and $WD_{opt}$ for all iterations in the Rel × It curves (Figure 4(a)). $GP^+$ is better than $QS_{str}$ from the fifth iteration on, but it is worse than $SVM_{active}$ and $SVM_{fuzzy}$. $GP^+$ is also better than $GP_{LSP}$ until the fifth iteration. $GP^\pm$ is better than $QS_{str}$, $GP_{LSP}$, and $SVM_{active}$, and slightly inferior than $SVM_{fuzzy}$. These results are confirmed by the Wilcoxon tests (Figure 4(c)). Similar results are observed for $P \times R$ curves (Figure 4(b)). Note, however, that $GP^\pm$ yields results statistically equivalent to $SVM_{fuzzy}$ (Figure 4(d)).
6.3 Performance Evaluation

One of the key aspects that need to be addressed during the definition of RF techniques is concerned with the real time execution requirement [42].

Table 5 shows the average execution time on each RF iteration for all evaluated RF approaches and for all three collections used in our experiments. The presented time refers to the average time required by each RF approach to learn the user needs and to select images to be showed on each iteration.

Table 5
Average execution time on each RF iteration (in seconds).

<table>
<thead>
<tr>
<th>Base</th>
<th>$GP^+$</th>
<th>$GP^\pm$</th>
<th>$WD_{heu}$</th>
<th>$WD_{opt}$</th>
<th>$SVM_{active}$</th>
<th>$QS_{str}$</th>
<th>$SVM_{fuzzy}$</th>
<th>$GP_{LSF}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>FISH</td>
<td>0.38</td>
<td>2.27</td>
<td>0.08</td>
<td>0.81</td>
<td>12.80</td>
<td>0.46</td>
<td>9.33</td>
<td></td>
</tr>
<tr>
<td>MPEG7</td>
<td>0.41</td>
<td>2.67</td>
<td>0.02</td>
<td>0.40</td>
<td>5.34</td>
<td>0.13</td>
<td>15.39</td>
<td></td>
</tr>
<tr>
<td>COREL</td>
<td>1.00</td>
<td>3.00</td>
<td>0.07</td>
<td>4.43</td>
<td>15.04</td>
<td>0.53</td>
<td>19.85</td>
<td>4.36</td>
</tr>
</tbody>
</table>

As can be observed in Table 5, $WD_{heu}$ yields the best results for all collections. However, regarding its effectiveness, this RF approach is worse than the proposed GP-based frameworks for all recall/iteration values, except for the two first iterations when the FISH collection is considered (see Figure 2). A similar behavior can be observed for $WD_{opt}$. Even though this approach is faster than $GP^\pm$ and $GP^+$, its overall effectiveness is worse.

$SVM_{active}$ and $SVM_{fuzzy}$ present the best effectiveness results among the baselines, but for the FISH collection. As pointed out before, these methods suffer to learn the user needs when the training set contains few examples of relevant images. For the other collections, while $SVM_{active}$ is, in general, worse than the proposed frameworks, $SVM_{fuzzy}$ is slightly superior (COREL collection). However, these methods are much slower than the proposed approaches (see Table 5). For the COREL collection, for example, $SVM_{active}$ is 1,404% and 402% slower than $GP^+$ and $GP^\pm$, respectively. $SVM_{fuzzy}$, in turn, is 1,885%
and 562% slower than $GP^+$ and $GP^\pm$, respectively.

Both $GP^+$ and $GP^\pm$ are faster than $GP_{LSP}$. Regarding $QS_{str}$, the proposed frameworks are slower in most cases. However, $QS_{str}$ yields worse effectiveness results for all image collections.

The execution time (between 0.38 and 3.00 seconds) of the proposed frameworks are acceptable. This performance is dependent on the choice of the GP parameters (see Section 6.1). Both frameworks were able to learn the user perception after a few generations, using small population sizes and low trees.

The execution time of $GP^+$ is also affected by the number of images labeled as relevant by the user on each iteration. The query pattern size is directly proportional to the number of relevant images found. In the similarity computation between the query pattern and an image $I$, it is needed to compute the individual similarity between $I$ and each query pattern image. Consequently, the greater the number of relevant images, the higher the query pattern size and the number of comparison operations executed. This behavior can be observed in Table 5. For example, the best execution time of $GP^+$ was obtained for the FISH collection, which includes only 10 relevant images per class.

The execution time of $GP^\pm$ is also affected by the query pattern size. However, not only are the relevant images considered in similarity computation, but also the non-relevant ones. Since the number of non-relevant images tends to be high, much more similarity computations are performed. This overhead makes $GP^\pm$ slower than $GP^+$ for all collections.
7 Conclusions

We have presented two novel relevance feedback-based CBIR frameworks. These methods use genetic programming to learn the user preferences, using the similarity functions defined for all available descriptors. The objective of the GP-based learning methods is to find a descriptor combination function that best represents the user perception.

Experiments were performed on three different image collections using several descriptors to characterize color, texture, and shape features. In these experiments, the proposed methods were compared with five other relevance feedback techniques [31,30,36,15,28] in image retrieval tasks. Furthermore, statistical significance tests were performed to compare experimental results. These experiments showed that the proposed frameworks outperform state-of-the-art baselines regarding their effectiveness and efficiency.

Future work includes the extension of the proposed frameworks to support the definition of degrees of relevance associated to each image showed to the user. Another important issue that will be investigated is the use of indexing structures to speed up the overall retrieval performance.

Acknowledgment

The authors would like to thank CNPq, CAPES, FAPESP, FAPEX, CAPES, and Microsoft Research for the financial support.
References


