Image Retrieval Employing Genetic Dissimilarity Weighting and Feature Space Transformation Functions [*]

Letricia P. S. Avalhais, Sergio F. da Silva,
Jose F. Rodrigues Jr. and Agma J. M. Traina
{letricia, sergio, junio, agma}@icmc.usp.br
Institute of Mathematics and Computer Science
University of São Paulo
São Carlos, Brazil

ABSTRACT

We present two promising Relevance Feedback methods based on Genetic Algorithms used to enhance the performance on the task of image retrieval according to the user's interests. The first method adjusts the dissimilarity function by using weighting functions while the second method redefines the feature space by means of linear and nonlinear transformation functions. Experimental results on real datasets demonstrate that our methods are effective and the results show that the transformation approach outperforms the weighting approach, achieving a precision gain of up to 70%.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Search Process, Relevance Feedback.

General Terms

Algorithm, Experimentation.

Keywords

Image Retrieval, Genetic Algorithm, Weighting, Functions.

1. INTRODUCTION

Techniques for image retrieval follow two main approaches [8]: the first one uses descriptions provided by textual annotation, which may introduce inconsistencies due to the human annotator. This is due to the fact that, in many domains, text cannot accurately capture the visual attributes of an image based on human perception. The second approach uses Content-Based Image Retrieval (CBIR), according to which the images are indexed/retrieved considering their visual content extracted as color histograms, texture and shape features [1]. Such features together define *feature vectors* containing m elements that are interpreted as points in an m-dimensional space. In the space of features, one assumes that a point of interest is surrounded by the points that represent the images that are the most similar to it, a similarity query well-known as k-Nearest Neighbor.

Despite dealing with inherent information obtained from the images, CBIR systems often present inaccurate results due to the challenging problem of associating low-level features with the high-level semantic of the images. This lack of correspondence between the high-level similarity on the point of view of the user and the low-level image features is known as *semantic gap* [14]. This problem can be caused, for example, by assuming that all features are equally relevant while processing similarity queries. Furthermore, some features can be very representative for some queries while being irrelevant for other queries. Also, in a given CBIR context, some features have poor or no semantic meaning, while other features are successful in capturing the semantics.

As an attempt to attenuate the semantic gap problem, Relevance Feedback (RF) methods have been proposed [3] [6] [17]. RF methods are very suitable for the task of providing to a CBIR system a mechanism that allows it to learn which features best capture the user's interests.

In the RF process, the user is supposed to evaluate the images retrieved in the current query by assigning them values that state relevance or no relevance. After this step the system reformulates the preceding query, taking into account the user's evaluations to improve its results. A review on RF for image retrieval is presented in [20].

Genetic Algorithms (GAs) provide a general adaptive search methodology based on natural selection, that has been successfully employed to perform feature selection and weighting on dissimilarity functions. In this work we propose two GA-based RF methods to enhance the accuracy on image retrieval tasks. The first method adjusts the dissimilarity function by means of weighting functions while the second transforms the feature space through transformation functions.

The remainder of this paper is structured as follows. Section 2 presents the related work. Section 3 presents the preliminaries and notations that we use throughout the paper. Section 4 describes formally the proposed methods. Section 5 details the experimental evaluation. Finally, Section 6 presents the conclusions and points out the future works.

[*] The authors are grateful for the financial support granted by FAPESP (process 2009/04232-8), CAPES, CNPq and Microsoft Research.

SAC'12 March 25-29, 2012, Riva del Garda, Italy.

2. RELATED WORK

Works on improving image retrieval effectiveness mainly employ RF, dissimilarity function learning and feature selection methods. Dynamic weighting mechanisms are applied to modify the distance function through appropriate weights, so that the distance between relevant images becomes smaller if compared to the non-relevant images.

In the work of Stejic *et al.* [15], the authors incorporate GA into RF mechanisms in order to infer weights assignment on features. However, the authors did not provide an effective model to learn the user's requests, because the *retrieval precision* evaluation function that they employ represents only the ratio of retrieved relevant images. Our work addresses this question through a GA-based RF mechanism that relies on an order-based ranking evaluation function.

Based on the premise that dissimilarity functions and image descriptors are problem-oriented, Torres *et al.* [18] proposed the use of nonlinear combination of multiple image similarities, addressing the problem through the use of Genetic Programming (GP). The learning process relies on training according to a training set and not on the user's feedback. Thus, the outcomes of their methods are not adjusted to the user's transient interests.

Other works attempt to improve precision of CBIR systems by working directly with the feature space [13] [5]. The work of Silva *et al.* [5] relies on ranking evaluation functions in order to choose the best set of features to represent images in the CBIR context. The contribution of each feature is binary (selected or not selected), while our methods allow to take advantage of the relative importance of each feature in image retrieval.

The methods introduced in this work explores both approaches, the weighting in dissimilarity functions and feature space transformations, as it will be detailed in Section 4.

3. PRELIMINARIES AND NOTATION

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ represent the set of feature vectors extracted from the image database $\mathbf{I} = \{\mathbf{i}_1, \ldots, \mathbf{i}_n\}$ using the feature extractor ε, i.e., $\varepsilon(\mathbf{i}_i) = \mathbf{x}_i = \{x_1, \ldots, x_m\}$, $\mathbf{x}_i \in \mathbb{R}^m$. Now consider $\mathbf{x}_q$ as being the feature vector extracted from the query image $\mathbf{i}_q$ and $\mathbf{x}_i$ the feature vector extracted from an arbitrary image $\mathbf{i}_i$. A dissimilarity function $d()$, also called distance function, provides a mechanism to measure the dissimilarity between $\mathbf{x}_q$ and $\mathbf{x}_i$.

Since the smaller the dissimilarity the larger the similarity is, the elements of $\mathbf{X}$ can be sorted according to their similarities to $\mathbf{x}_q$. In CBIR the most popular form of similarity query is the k-Nearest Neighbor query, or kNN query for short.

The kNN query aims at finding the k closest objects to a query object [19]. A kNN query is formally presented in Definition 1.

Definition 1. Given the set of objects $\mathbf{X}$, a distance function $d()$ and the query object $\mathbf{x}_q$, the response ranking set $\mathbf{T}$ of a kNN query is defined as:

$$kNN(\mathbf{x}_q, k, d(), \mathbf{X}) = \{\mathbf{T} \subseteq \mathbf{X}, |\mathbf{T}| = k \wedge \forall \mathbf{x}_i \in T, \mathbf{x}_j \in \mathbf{X} \setminus \mathbf{T} : d(\mathbf{x}_q, \mathbf{x}_i) \leq d(\mathbf{x}_q, \mathbf{x}_j)\} \tag{1}$$

In order to evaluate the accuracy of kNN queries into the optimization process of RF we employed a ranking quality measure, described in the Definition 2. This is an order-based function, i.e., it considers the order of the retrieved images [11], and it is suitable to our purpose due to the advantage of providing different values for different queries results, even though the number of relevant images retrieved is the same.

Definition 2. Given the ranking $\mathbf{T}_i = (\mathbf{t}_1, \ldots, \mathbf{t}_k)$ as the result of the query $kNN_i(\mathbf{x}_q, k, d(), \mathbf{X})$ and $\mathbf{R}_q = \{\mathbf{r}_1, \ldots, \mathbf{r}_\rho\}$, $\mathbf{R}_q \subseteq \mathbf{X}$ the set of objects that belong to the same class of $\mathbf{x}_q$, also called here the relevant objects for $\mathbf{x}_q$, the measure of the quality of $\mathbf{T}_i$ is calculated by the function:

$$\Phi(\mathbf{T}_i, \mathbf{R}_q) = \sum_{j=1}^{k} \frac{r(\mathbf{t}_j)}{A} \cdot \left(\frac{(A-1)}{A}\right)^{j-1} \tag{2}$$

where $r(\mathbf{t}_j) = 1$, if $\mathbf{t}_j \in \mathbf{R}_q$ or $r(\mathbf{t}_j) = 0$ otherwise, and $A \geq 2$ is an adjustment parameter that expresses the relative importance of the position of the elements on the ranking. Small values for A means that it is giving more importance for the relevant elements on the first positions of the ranking. When A is large the fraction $\frac{(A-1)}{A}$ is close to 1, then the position of the elements on the ranking is not strongly considered.

The ranking quality measure is applied in the GA as the fitness function. Indeed, our main goal relies on achieving a ranking that maximizes the value of this function.

4. PROPOSED METHODS

4.1 Overview of the System

The entire scheme of the system is shown in Figure 1. Initially, the user chooses the query image $\mathbf{i}_q$ from the image dataset $\mathbf{I}$. Then, the system computes the distance between the respective feature vector $\mathbf{x}_q$ to the features of the other images.

After this, a ranking is generated and presented to the user, who is in charge to evaluate the first k images in the ranking assigning them relevance values as *relevant*, $r(\mathbf{t}_j) = 1$ if the image is relevant to the query; and *not desirable*, $r(\mathbf{t}_j) = -1$ if the image is not supposed to be in the ranking. By default, the system assigns *not relevant*, $r(\mathbf{t}_j) = 0$ to every image that eventually has not been evaluated by the user.

This iterative process is determinant to define the set of relevant objects $\mathbf{R}_q$. Since many images can belong to multiple classes even to the user's judgment, it is the specific user's need that will define whenever an image is related or not to the image query $\mathbf{i}_q$.

Once the user's feedback is provided, the GA searches for vectors of *weighting* or *transformation functions* ($\delta_{\mathbf{W}}$ or $\delta_{\mathbf{T}}$) taking into account the highest values of fitness. These feature vectors will be used in order to redefine the distance function or the feature space in such a way that the resulting ranking maximizes the fitness value given by Equation 2 considering the user's feedback.

Values of relevance provided by the user are stored in a buffer between the RF steps. The cycles of RF/GA-search are repeated until the best solution is found, which means that the resulting ranking contains only relevant images in

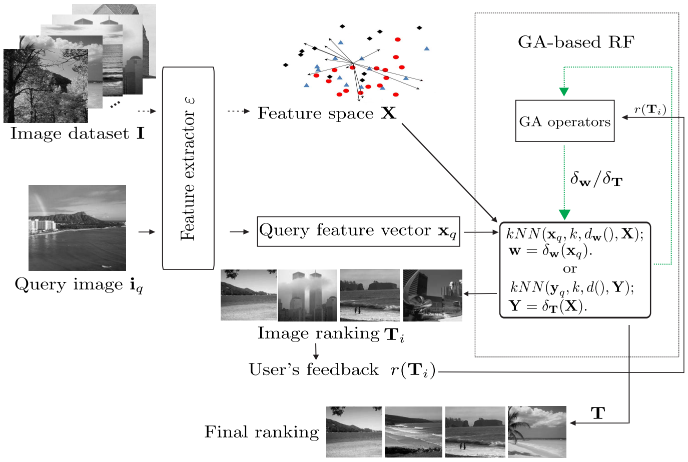


Image dataset $\mathbf{I}$
Feature extractor ε
Feature space $\mathbf{X}$
GA-based RF
GA operators
$r(\mathbf{T}_i)$
$\delta_{\mathbf{w}}/\delta_{\mathbf{T}}$
Query image $\mathbf{i}_q$
Query feature vector $\mathbf{x}_q$
$kNN(\mathbf{x}_q, k, d_{\mathbf{w}}(), \mathbf{X})$;
$\mathbf{w} = \delta_{\mathbf{w}}(\mathbf{x}_q)$.
or
$kNN(\mathbf{y}_q, k, d(), \mathbf{Y})$;
$\mathbf{Y} = \delta_{\mathbf{T}}(\mathbf{X})$.
Image ranking $\mathbf{T}_i$
User's feedback $r(\mathbf{T}_i)$
Final ranking
$\mathbf{T}$

Figure 1: An overview of the developed system.

its first k positions or until a predefined maximum number of cycles is reached.

In order to achieve the maximization of the function in Equation 2 and consequently to enhance the accuracy of the distance function in respect to the images representation, we propose two approaches:

- inferring a weight vector by means of weighting functions. This allows the distance function to take into account the degree of contribution of each feature according to the user's feedback.
- optimizing the feature space by inferring a feature space transformation, adjusting the original space feature for a new one, in order to better represent the user's similarity criteria.

4.2 Weighting Functions Approach

Using GA to infer weight vector for the dissimilarity function has achieved promising results as pointed out previously. Our first approach is based on this concept, however the weights are generated by well-known mathematical functions. This proposal came from investigating the relation between the distribution of the features before and after the use of a weight vector to calculate the distances. The assumption made was that there could be a well defined way to assign weight to each feature to improve the ranking generated by the similarity queries. In accordance to this assumption, we implemented a CBIR system that integrates RF mechanism based on GA search heuristic to find a weight vector calculated by a set of weighting functions.

The problem, formally presented in Definition 3, consists on finding a sequence of functions that generates the best weight vector for a specific query. The chromosome of the GA is encoded slightly different as the most usual approach, where it is assigned to the genotype real values corresponding to the weights. In this case, the values assigned to the genotype are integers that correspond to the number of a weighting function.

Definition 3. Let $\mathbf{F} = \{f_1, \ldots, f_\rho\}$ be a set of functions and $\delta_{\mathbf{w}} = (f'_1, \ldots, f'_m)$, $f'_i \in \mathbf{F}$ be the ordered set of weighting functions inferred. Now let $\mathbf{w} = \delta_{\mathbf{w}}(\mathbf{x}_q) = \{f'_1(x_{q_1}), \ldots, f'_m(x_{q_m})\}$ be the weighting vector for the distance function $d_{\mathbf{w}}()$. The problem using the weighting functions is to find a set $\delta_{\mathbf{w}}$ such that

$$\arg\max_{\delta_{\mathbf{w}}} \Phi(\mathbf{T}, \mathbf{R}_q) \tag{3}$$

where $\mathbf{T} = kNN(\mathbf{x}_q, k, d_{\mathbf{w}}(), \mathbf{X})$.

The advantage of dealing with weighting functions is that the search space for the GA is significantly reduced. This is because the search space in the continuous interval $[0, 1]$ is higher than the discrete and finite search space $\mathbf{F}$.

4.3 Feature Space Transformation Functions Approach

The second approach we introduce is based on feature space transformation, where each feature value is redefined by applying a transformation function, as detailed in Definition 4. These functions provide linear transformation and nonlinear transformations on original features space, a way to capture nonlinear relationships in the feature space.

Transformations on the feature space are performed using the GA to find the sequence of functions to be applied to the original feature space. For each sequence of transformation functions provided by the GA, a new feature space is calculated using the original feature values, and the kNN query is executed.

Definition 4. Given $\mathbf{F}$, as previously set out, let $\delta_{\mathbf{T}}$, defined analogously to $\delta_{\mathbf{w}}$, be an ordered set of transformation functions inferred and $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ the new feature space, and each $\mathbf{y}_i$ corresponding to a $\mathbf{x}_i$ transformed by the functions in $\delta_{\mathbf{T}}$, i.e., $\mathbf{y}_i = \delta_{\mathbf{T}}(\mathbf{x}_i) = \{f'_1(x_{i_1}), \ldots, f'_m(x_{i_m})\}$.

The problem using the feature space transformation functions is to find a set $\delta_{\mathbf{T}}$ such that

$$\arg\max_{\delta_{\mathbf{T}}} \Phi(\mathbf{T}, \mathbf{R}_q) \tag{4}$$

where $\mathbf{T} = kNN(\mathbf{y}_q, k, d(), \mathbf{Y})$.

4.4 Genetic Algorithm Description

When implementing a GA, it is necessary to consider its parameters and operators, such as the chromosome coding, the fitness function, selection, crossover and mutation operators. The values of the parameters and the chosen operators can be decisive regarding to the effectiveness of the algorithm. In this work, such choices were made experimentally and are described as follows:

- *Chromosome coding*: in both methods proposed a chromosome was coded as an integer-valued vector with m positions, $\mathbf{C} = (g_1, \ldots, g_m)$, where g_i corresponds to an identifier of a function in $\mathbf{F}$. However, in the weighting functions approach, the chromosome gives the weight vector $\mathbf{w}$ for the distance function $d_{\mathbf{w}}()$, while in the feature space transformation functions approach it provides the new feature space $\mathbf{Y}$.
- *Fitness function*: as fitness function we employed the ranking quality measure presented in Definition 2.
- *Selection for recombination operator*: we used exponential ranking selection to select pairs of individuals

to reproduce. The probability of being selected p_i for an individual $\mathbf{C}_i$ is given by Equation 5:

$$p_i = \frac{c^{Sp-i}}{\sum_{j=1}^{Sp} c^{Sp-j}}, 0 \leq c \leq 1 \qquad (5)$$

where $i \in \{1, \ldots, Sp\}$, Sp is the population size and $c = 0.9$.

- *Selection for reinsertion operator*: elitism was employed to select the surviving chromosome for the next generation. This is because elitism guarantees that the best individual of the population in a generation g will be present in the population of generation $g + 1$, and so it guarantees that the best solution will always be improved or, at least, maintained.

- *Crossover operator*: we employed uniform crossover. A mask is randomly built and it indicates which chromosome will supply each gene for the first offspring. The second offspring is generated by the complement of the same mask.

- *Mutation operator*: uniform mutation is applied on the offspring chromosomes. The genes are selected with probability P_m and their values are changed by another valid value chose randomly.

5. EXPERIMENTAL RESULTS

In this section we present representative experiments on the proposed methods. The evaluation of our methods is based on a comparison with the classical weighting approach, where weights are directly generated in the range $[0, 1]$. The datasets, the features extractors and the GA parameters used are also described.

Image Datasets Descriptions

Experiments were executed considering several datasets, but due to space limitations, we only describe the results for two of them. The first image database is *Corel* 1000 [1] [9], which contains 1000 images classified in 10 classes with 100 images per class (africa, beach, buildings, buses, dinosaurs, elephants, flowers, food, horses and mountains).

The second dataset belongs to the medical domain (*Lung Diseases*), which consists of 246 magnetic resonance images of human lungs, collected at the Clinical Hospital at University of São Paulo in Ribeirão Preto. This dataset is divided in 6 classes being 1 of normal and 5 abnormal patterns (consolidation, emphysema, thickness, honeycombing, groundglass opacity), with an average of 41 images per class, varying from 39 to 44.

Feature Extractors

For each dataset, the feature vectors were acquired by the feature extractors: *Color Moments* [16], *Co-occurrence* [7], *Sobel Histogram* [2], *Histogram* [4], *Run Length* [10] and *SIFT* [12]. Table 1 shows the number of features extracted and the type of the information captured by each extractor. When extracting *Color Moments* and *Sobel Histogram*, the images were partitioned into 16 rectangular regions. The respective features were extracted from each region and combined in a single vector.

We notice that if the feature extractor doesn't describe any relevant aspect of the user's interests, this can lead to the set $\mathbf{R}_q$ be empty, and consequently, the GA may not converge to satisfactory solutions.

Table 1: Feature extractors employed

Feature extractor	Number of features	Type
Color Moments	144	Color
Co-occurrence	88	Texture
Sobel Histogram	128	Shape
Histogram	256	Color
Run Length	44	Texture
SIFT	128	Shape

GA Parameters

The GA is configured with the following parameters: population size (Sp) = 50; maximum number of generations (Ng) = 100; crossover rate (P_c) = 0.8 and mutation rate (P_m) = 0.2.

Evaluation Graphics

All experiments were performed using the three following methods: direct weights generator (WG), weighting functions (WF) and transformation functions (TF). The performance of the fitness function is compared by adjusting the parameter A with values 2, 10 and 20. Also, the performance of each feature extractor employed is also analyzed. Regarding the kNN query, we empirically selected $k = 30$ and the maximum number of cycles of RF/GA-search is 10.

The effectiveness of our proposed methods WF and TF, and comparisons with the WG method was assessed by two graphics: Precision *versus* Recall, and Number of Relevant images *versus* Cycles of RF.

Corel 1000 Results

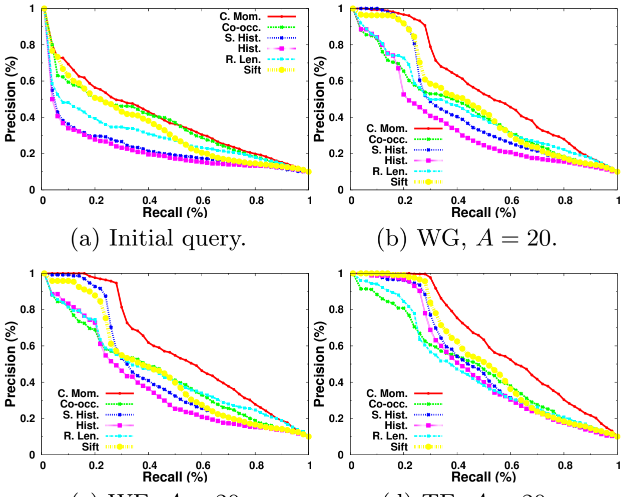
Experiments on *Corel* 1000 are presented considering an average of 10 image queries, one from each class, randomly chosen. Figure 2(a) shows low precision of each feature extractor in the initial query and a continuous improvement of precision as the execution evolves toward the last RF cycle, obtained using each method (Figures 2(b), (c) and (d)). It can be observed that the WF (Figure 2(c)) obtains higher precision then WG (Figure 2(b)), near to 20% of recall with *Color Moments* feature extractor, and TF (Figure 2(d)) achieved the highest precision, near to 30% of recall, also using *Color Moments*.

Considering the extractor *Color Moments*, Figure 3(a) shows that the best results were achieved when using TF, with $A = 20$, $A = 10$ and $A = 2$, respectively. Following, one can see the WF method with $A = 20$ and $A = 10$. In Figure 3(b) it is possible to observe that the precision increases until the fifth cycle, with low or no improvement after that.

The number of relevant images retrieved per cycle obtained by each feature extractor while using TF is shown on Figure 4(a). It can be observed that the texture-based feature extractors, *Co-occurrence* and *Run Length*, have similar and poor improvement while compared to the other ones. Figure 4(b) confirms the efficacy of the TF method, followed by the WF method, while analyzing the number of relevant

[1] `http://wang.ist.psu.edu/docs/related/`; Last accessed: August 23, 2011.

images retrieved per cycle for the average of all feature extractors.



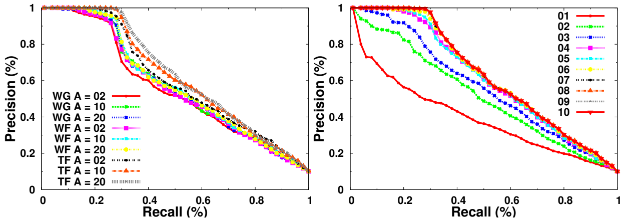
(a) Initial query.

(b) WG, $A = 20$.

(c) WF, $A = 20$.

(d) TF, $A = 20$.

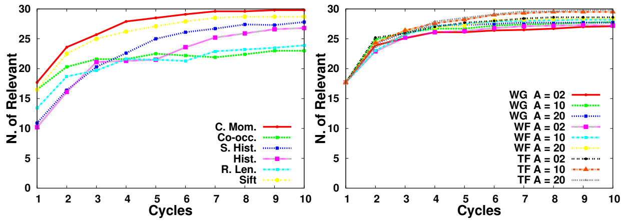
Figure 2: Precision *vs* Recall for each feature extractor and $A = 20$ (a) initial query, (b) cycle 10 using WG, (c) cycle 10 using WF, (d) cycle 10 using TF.



(a) Cycle 10.

(b) TF, $A = 20$.

Figure 3: Precision *vs* Recall for the feature extractor *Color Moments* (a) cycle 10 for each method and for each value of A, (b) evolution through the cycles 01 to 10 using TF and $A = 20$.



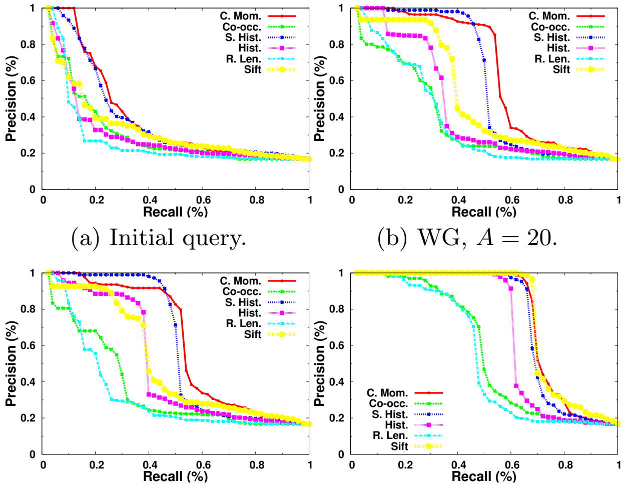
(a) TF, $A = 20$.

(b) *Color Moments*.

Figure 4: Number of Relevant (a) for each feature extractor using TF and $A = 20$, (b) using *Color Moments* for each method and for each value of A.

***Lung Diseases* Results**

For this dataset we randomly chose 6 images, one from each class, and the results presented are an average of these queries. Figure 5(a) shows the low precision of each feature extractor of the initial query, with the best performance, again, obtained by using *Color Moments*. Figure 5(b) and 5(c) illustrate the performance of WG and WF methods respectively, with a slightly better effectiveness of WF and best improvement achieved by the feature vector *Sobel Histogram* in both cases. Figure 5(d) shows significantly higher effectiveness of TF method (with 100% of precision until 60% of recall) in comparison to both WG and WF, and the best improvement achieved using *SIFT*.



(a) Initial query.

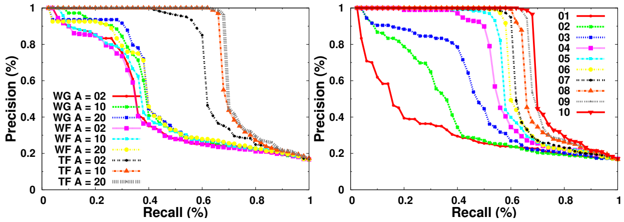
(b) WG, $A = 20$.

(c) WF, $A = 20$.

(d) TF, $A = 20$.

Figure 5: Precision *vs* Recall for each feature extractor and $A = 20$ (a) initial query, (b) cycle 10 using WG, (c) cycle 10 using WF, (d) cycle 10 using TF.

Considering the feature extractor *SIFT*, Figure 6(a) shows the highest precision when using TF and $A = 20$, 10 and 2, in decreasing order. Furthermore, the performance difference in using weighting and transformation techniques is noticeable. For instance, taking 60% of recall, the transformation approach achieved 100% of precision, while weighting was around 30% of precision. The results on each cycle, using *SIFT*, TF and $A = 20$, are illustrated on Figure 6(b). Once more, after the fifth cycle low improvements on precisions are reached.

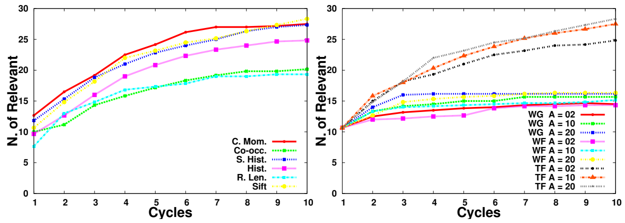


(a) Cycle 10.

(b) TF, $A = 20$.

Figure 6: Precision *vs* Recall for the feature extractor *SIFT* (a) cycle 10 for each method and for each value of A, (b) evolution through the cycles 01 to 10 using TF and $A = 20$.

Figure 7(a) shows that, also in this dataset, the texture-based feature extractors employed obtained the lowest effectiveness on TF method. Considering now the average of all feature extractors, Figure 7(b) illustrates that the TF method significantly outperforms the others, and WF using $A = 20$ has a slightly better result than the WG method in the last three cycles.



(a) TF, $A = 20$.

(b) *SIFT*.

Figure 7: Number of Relevant (a) for each feature extractor using TF and $A = 20$, (b) using *SIFT* for each method and for each value of A.

6. CONCLUSIONS

In the field of image retrieval, several works proposed to use GA combined with RF techniques to improve the accuracy of CBIR systems. In order to take advantage of this approach, this paper presented two methods. The first one infers a weight vector to adjust the Euclidean distance by means of weighting functions. The second one aims at optimizing the feature space using transformation functions. The advantage of using weighting functions instead of directly generated weights is that the search space is considerably reduced. Also, the results considering feature space transformation functions achieved successful effectiveness, obtaining the highest precision in all experiments, outperforming the other methods by up to 70%.

The higher performance obtained by using the feature space transformation is due to the fact that it leads to a configuration in which the space is transformed by different functions such as polynomials with several different degrees. Meanwhile, weighted distance functions limit the configuration to linear transformations.

As future work, one possible approach is to investigate the use of the weighting and transformation functions for further feature extractors and distance functions, as well as to study possible correlations between the different feature extractors and its behavior.

7. REFERENCES

[1] Y. Alemu, J. bin Koh, M. Ikram, and D.-K. Kim. Image retrieval in multimedia databases: A survey. *IIH-MSP*, 0:681–689, 2009.

[2] S. Brandt. Use of shape features in content-based image retrieval. Master's thesis, Helsinki University of Technology, 1999.

[3] P. Bugatti, A. Traina, and C. Traina. Improving content-based retrieval of medical images through dynamic distance on relevance feedback. In *CBMS*, pages 1 –6, june 2011.

[4] P. H. Bugatti, A. J. M. Traina, and C. Traina, Jr. Assessing the best integration between distance-function and image-feature to answer similarity queries. In *Proceedings of the 2008 ACM SAC*, pages 1225–1230, New York, NY, USA, 2008.

[5] S. F. da Silva, M. X. Ribeiro, J. do E.S. Batista Neto, C. Traina-Jr., and A. J. Traina. Improving the ranking quality of medical image retrieval using a genetic feature selection method. *DSS*, In Press:1–11, 2011.

[6] J. dos Santos, C. Ferreira, R. da S. Torres, M. Gonçalves, and R. Lamparelli. A relevance feedback method based on genetic programming for classification of remote sensing images. *Information Sciences*, 181(13):2671 – 2684, 2011.

[7] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural Features for Image Classification. *IEEE/SMC*, 3:610–621, 1973.

[8] A. Lakdashti, M. Shahram Moin, and K. Badie. Semantic-based image retrieval: A fuzzy modeling approach. In *IEEE/AICACS.*, pages 575–581, 2008.

[9] J. Li and J. Z. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE/TPAMI*, 25:1075–1088, 2003.

[10] H.-H. Loh, J.-G. Leu, and R. Luo. The analysis of natural textures using run length features. *Industrial Electronics, IEEE Transactions on*, 35(2):323 –328, may 1988.

[11] C. López-Pujalte, V. P. Guerrero-Bote, and F. de Moya-Anegón. Order-based fitness functions for genetic algorithms applied to relevance feedback. *J. Am. Soc. Inf. Sci. Technol.*, 54:152–160, January 2003.

[12] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV, IEEE*, volume 2, pages 1150–1157, Los Alamitos, CA, USA, 1999.

[13] d. l. T. F. Nguyen, M.H. Optimal feature selection for support vector machines. *Pattern Recogn.*, 43:584–591, 2010.

[14] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22:1349–1380, December 2000.

[15] Z. Stejic, Y. Takama, and K. Hirota. Genetic algorithms for a family of image similarity models incorporated in the relevance feedback mechanism. *Applied Soft Computing*, 2(4):306 – 327, 2003.

[16] M. A. Stricker and M. Orengo. Similarity of color images. In *Storage and Retrieval for Image and Video Databases*, pages 381–392, 1995.

[17] J.-H. Su, W.-J. Huang, P. S. Yu, and V. S. Tseng. Efficient relevance feedback for content-based image retrieval by mining user navigation patterns. *IEEE-TKDE*, 23:360–372, 2011.

[18] R. d. S. Torres, A. X. Falcão, M. A. Gonçalves, J. a. P. Papa, B. Zhang, W. Fan, and E. A. Fox. A genetic programming framework for content-based image retrieval. *Pattern Recogn.*, 42:283–292, February 2009.

[19] P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search: The Metric Space Approach*. Advances in Database Systems. Springer, 2006.

[20] X. S. Zhou and T. S. Huang. Relevance feedback in image retrieval: A comprehensive review. *Multimedia Systems*, 8:536–544, 2003. 10.1007/s00530-002-0070-3.