Content-Based Image Retrieval system based on Bayesian Relevance Feedback

Ahmed Samady, Fahd Chibani, Mohamed Amine Fakhre-Eddine
IT Department, FST, Abdelmalek Essaadi University
ahmed.samady@etu.uae.ac.ma, fahd.chibani@etu.uae.ac.ma, contact@fakhreeddine.dev,
mohamedamine.fakhreeddine@etu.uae.ac.ma

***Abstract*—The theory, design principles, implementation, and performance outcomes of our two-week-long content-based image retrieval (CBIR) system are presented in this paper. Furthermore, we suggest a Bayesian Inference-based relevance feedback method. In order to achieve this, we use user provided labels to determine whether a query is relevant. We then use a Bernoulli likelihood model, a Gaussian model, and Markov Chain Monte Carlo (MCMC) sampling in order, to update the weights of each feature, which improves and refines the results of a user query. Lastly, we demonstrated our implementation validating the theoretical conclusions.**

***Index Terms*—Content-Based Image Retrieval (CBIR), Bayesian Inference, Relevance Feedback, Image Search**

I. Introduction

NOwadays, images make up a large part of our data, and the quantity is only increasing with time, so dealing with them is essential, with that comes the responsibility of image retrieval from large datasets based on user queries, wants and needs. Content-Based Image Retrieval (CBIR) is a technique to retrieve images semantically relevant to the user's query from an image database, based on several descriptors and features. This poses a computation and performance problem, which can be improved with Relevance Feedback, a process in which we gather user feedback after a simple search, as labels of relevant and irrelevant images to the search. Until we obtain the results we want, we iteratively refine our search results. A number of relevance feedback techniques were introduced and reviewed in the literature [1], in this paper, we will focus on implementing a Content-Based Image Retrieval system featuring simple search, and advanced search based on Bayesian relevance feedback.

The paper then is organized as follows. In Section III, we will define the descriptors used in our method. In Sections V and VI, we will introduce the math behind our simple search, and advanced search methods respectively. Finally we will finish with an implementation and demonstration in Section VII and a Conclusion.

II. Literature review

A. Relevance feedback

The relevance feedback concept has been extensively studied in the text retrieval field, highlighting the importance of user interaction in improving information retrieval system performance. Techniques developed for text retrieval should be adapted to content-based image retrieval [2], considering differences in feature number, meaning, and similarity measures [3], [4].

B. Relevance feedback for CBIR

Relevance feedback strategies are motivated by the observation that users are unaware of the distribution of images in the feature space or the feature space itself. These techniques involve optimizing one or more Content-Based Image Retrieval (CBIR) components, such as formulating new queries or modifying similarity metrics to consider each feature's relevance to the user query [5].

The following describes a typical relevance feedback scenario in CBIR:

Step 1 Using a query-by-example, sketch, etc., the machine provides preliminary retrieval results.

Step 2 The user evaluates whether and to what extent the images that are currently on display are pertinent to the query.

Step 3 The machine learns and makes another attempt. Return to **Step 2**.

Keep in mind that until the user is happy with the retrieval results, steps two and three must be repeated [1].

A new relevance feedback mechanism was presented by G. Ciocca and R. Schettini. It examines the feature distributions of the images that the user deems relevant or irrelevant and dynamically updates the query and similarity measure to suit the user's specific information needs [6].

Using a generative model for feature representation based on embedded mixtures, Nuno Vasconcelos and Andrew Lippman first presented a Bayesian framework for CBIR. It has been demonstrated that this genuinely generic image representation works well across a wide range of image databases and is capable of jointly modeling color and texture. They subsequently elaborated, demonstrating that the formulation of CBIR as a Bayesian inference problem results in a natural criteria for assessing local image similarity without the need for image segmentation. This makes it possible to implement retrieval systems in practice where users can supply image regions or objects as queries. Furthermore, they introduced a Bayesian learning algorithm that incorporates user feedback during a retrieval session by using belief propagation. This algorithm creates a powerful paradigm for user interaction when paired with local similarity [7].

By using a probability distribution over potential image targets instead of refining a query, Ingemar J. Cox, Matt L.

Masters of Science and Technology in Artificial Intelligence and Data Science, January 25, 2025

Miller, Thomas P. Minka, Thomas V. Papathomas, and Peter N. Yianilos created *PicHunter*, an app based on Bayes' rule to predict what the user wants given their actions. Additionally, there is an entropy-minimizing display algorithm that aims to maximize the amount of data gathered from a user at every search iteration. Additionally, instead of using a potentially inaccurate or inconsistent annotation structure that the user must become familiar with and use to formulate queries in, they employed hidden annotation [8].

Zhong Su, Hongjiang Zhang, Stan Li, and Shaoping Ma presented a novel method based on a Bayesian classifier that treats examples of positive and negative feedback using distinct strategies. While negative examples are used to change the ranking of the retrieved candidates, positive examples are used to estimate a Gaussian distribution that represents the desired images for a given query. Furthermore, they reduced the dimensionality of feature spaces and extracted and updated a feature subspace based on user feedback using the Principal Component Analysis (PCA) technique [9].

Giorgio Giacinto and Fabio Roli estimated the difference between relevant and non-relevant images using the Bayesian decision theory. After that, a new query is calculated whose neighborhood is probably located in a feature space region that has pertinent images [5].

III. Descriptors

Both of our methods are based on a number of descriptors to extract useful and meaningful information from images.

A. Dominant Colors

Images depend heavily on color; even a small group of colors can convey a whole image. Numerous research tasks, including retrieving images using dominant colors [10], [11], rely on this "set" of dominant colors, also known as a color palette. Applications such as palette creation, color editing, and image search can all make use of an image's dominant colors. Traditionally, techniques based on histograms or clustering are used to extract dominant colors [12], which we opted for in our research [13].

1) **Input** $X = \{X_1, X_2, ...X_n\}$ where X_i represents RGB values of each pixel.
2) **Objective**

$$Minimize \sum_{i=1}^{N}\sum_{j=1}^{k} r_{ij}\|X_i - \mu_j\|^2$$

where:

- μ_j is the center of cluster j,
- $r_{ij} = 1$ if X_i is assigned to cluster j, otherwise $r_{ij} = 0$.

3) **Steps**
   - Initialize k cluster centers $\mu_1, \mu_2, ..., \mu_k$.
   - Assign each pixel to the nearest cluster center:

$$r_{ij} = \begin{cases} 1 & \text{if } j = argmin_k\|X_i - \mu_k\| \\ 0 & \text{otherwise.} \end{cases}$$

   - Update cluster centers:

$$\mu_j = \frac{\sum_{i=1}^{N} r_{ij}X_i}{\sum_{i=1}^{N} r_{ij}}$$

This process iterates until convergence.

B. Color Histogram

One of the most effective descriptors for identifying objects and scenes is color. For digital images, a color histogram shows the number of pixels with colors in each of the specified color ranges, along with the image's color model and the collection of all likely colors [14], [15]. Histogram features have shown great promise in CBIR, object detection, and image classification.

For each channel c:

1) Divide the intensity range $[0, 256)$ into b bins.
2) Count the number of pixels n_k whose intensity $I(x, y)$ falls into bin k:

$$n_k = Number\ of\ pixels \quad where\ \frac{256}{b}k \leq I(x,y) \leq \frac{256}{b}(k+1)$$

3) Normalize the histogram:

$$h_k = \frac{n_k}{\sum_{j=1}^{b} n_j}$$

C. Hu Moments

The weighted average of an image's pixel intensities is called an image moment. For a wide range of applications, moments and their associated invariants have been thoroughly studied in order to describe the patterns found in images [16].

1) **Raw Moments**

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y)$$

where $I(x, y)$ is the pixel intensity.

2) **Central Moments** With the exception of subtracting the centroid from the x and y in the moment formula, central moments are quite similar to the raw image moments we previously saw.

$$\mu_{ij} = \sum_x \sum_y (x - \bar{x})^i (y - \bar{y})^j I(x, y)$$

where $I(x, y)$ is the pixel intensity, $\bar{x} = \frac{m_{10}}{m_{00}}$, $\bar{y} = \frac{m_{01}}{m_{00}}$ are the centroids.

3) **Normalized central moments**

$$\eta_{ij} = \frac{\mu_{ij}}{\mu_{00}^{(i+j)/2+1}}$$

4) **Hu moments** The translation invariance of central moments is fantastic. However, shape matching requires more than that. We want to compute moments that are invariant to translation, scale, and rotation.

Using central moments that are invariant to image transformations, a set of seven numbers known as Hu Moments (or Hu moment invariants) are calculated. It has been established that the first six moments are independent of translation, scale, rotation, and reflection. The sign of the seventh moment shifts to reflect the image [16].

$\phi_1 = \mu_{20} + \mu_{02}$
$\phi_2 = (\mu_{20} - \mu_{02})^2 + 4(\mu_{11})^2$
$\phi_3 = (\mu_{30} - 3\mu_{12})^2 + (\mu_{03} - 3\mu_{21})^2$
$\phi_4 = (\mu_{30} + \mu_{12})^2 + (\mu_{03} + \mu_{21})^2$
$\phi_5 = (\mu_{30} - 3\mu_{12})(\mu_{30} + \mu_{12})((\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2) + (3\mu_{21} - \mu_{03})(\mu_{21} + \mu_{03})(3(\mu_{30} + \mu_{12})^2 - (\mu_{03} + \mu_{21})^2)$
$\phi_6 = (\mu_{20} - \mu_{02})((\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2) + 4\mu_{11}(\mu_{30} + \mu_{12})(\mu_{21} + \mu_{03})$
$\phi_7 = (3\mu_{21} - \mu_{03})(\mu_{30} + \mu_{12})((\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2) + (\mu_{30} - 3\mu_{12})(\mu_{21} + \mu_{03})(3(\mu_{30} + \mu_{12})^2 - (\mu_{03} + \mu_{21})^2)$

D. Gabor Texture

Using multi-resolution analysis, the gabor filter—a tool for extracting texture features—has shown great efficacy in characterizing visual content. Thus, using for texture feature extraction in CBIR [17], [18].

$$g(x, y, \lambda, \theta, \sigma, \gamma, \phi) = \exp(-\frac{x'^2 + \gamma'^2 y'^2}{2\sigma^2}) \cos(2\pi \frac{x'}{\lambda} + \phi)$$

where:

- $x' = x\cos\theta + y\sin\theta$
- $y' = x\sin\theta + y\cos\theta$
- λ wavelength,
- θ orientation,
- σ standard deviation,
- γ aspect ratio,
- ϕ phase offset.

Each filter response is averaged to give a texture descriptor.

E. Edge Histogram

Based on the local edge distribution in an image, the Edge Histogram Descriptor (EHD) uses a histogram to describe edge distribution. For CBIR systems, these two features are extremely potent, particularly for sketch-based image retrieval. Furthermore, CBIR systems produce more accurate image retrieval results when color and texture features are combined [19], [20].

1) Apply **Canny edge detection**
   - **Compute image gradients**

$$G_x = \frac{\partial I}{\partial x}, G_y = \frac{\partial I}{\partial y}$$

   - **Gradient magnitude**

$$G = \sqrt{G_x^2 + G_x^2}$$

2) Compute histogram of edge intensities from G.

F. Fourier Descriptors

One of the most crucial aspects of content-based image retrieval is shape. Some shape-based retrieval techniques are available, but they fail to accurately represent the shape, making matching challenging. One method for retrieving images based on the shape of the image components is to use Fourier Descriptors (FDs), which are made invariant against scale, rotation, and translation to achieve a well representation and normalization [21]–[23].

1) Extract contour points $p = \{(x_i, y_i)\}_{i=1}^N$
2) Represent as a complex sequence

$$z_i = x_i + jy_i, \ i = 1, 2, ..., N$$

3) Apply the Discrete Fourier Transform (DFT)

$$Z_k = \sum_{i=1}^{N} z_i \exp(-j\frac{2\pi ki}{N}), k = 1, 2, ..., N-1$$

4) Use the first n magnitudes as descriptors

$$|Z_k| = \sqrt{Re(Z_k)^2 + Im(Z_k)^2}$$

IV. Distances

To make use of these features, we have to compute the distance between a query image and an indexed image for each feature.

- **Color Histogram** (H_c)
- **Dominant Colors** (D_c)
- **Edge Histogram** (H_e)
- **Gabor Features** (G)
- **Hu Moments** (M_h)
- **Fourier Descriptors** (F_c)

For every feature, the distance between a database image (d) and the query image (q) is calculated as follows:

1) **Color Histogram Distance** (d_{H_c})

$$d_{H_c} = \frac{1}{3}\sum_{i=1}^{3} \chi^2(H_{c,q}^i, H_{c,d}^i)$$

where χ^2 is the Chi-Square distance between histograms of each color channel i defined as:

$$\chi^2 = \frac{1}{2}\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{x_i + y_i}$$

2) **Dominant Colors Distance** (d_{D_c})

$$d_{D_c} = \|D_{c,q} - D_{c,d}\|_2$$

where $\|\cdot\|_2$ denotes the Euclidean distance.

3) **Edge Histogram Distance** (d_{H_e})

$$d_{H_c} = \chi^2(H_{e,q}, H_{e,d})$$

4) **Gabor Features Distance** (d_G)

$$d_G = \|G_q - G_d\|_2$$

5) **Hu Moments Distance** (d_{M_h})

$$d_{M_h} = \|M_{h,q} - M_{h,d}\|_2$$

6) **Fourier Descriptors Distance** (d_{F_d})

$$d_{F_d} = \|F_{d,q} - F_{d,d}\|_2$$

V. Simple search

A. Similarity

Now that we have calculated the respective distances, we can compute the similarity between a query image and an indexed image. Here we introduce a new set of features and weights, to include the previously stated descriptors thus being more general. General weights will comprise **Color**, **Shape**, and **Texture**.

The following formula is used to calculate each feature's individual similarity based on its distance:

$$S_f = \frac{1}{1 + d_f}$$

where d_f is the distance for a specific feature f, and S_f is the similarity for that feature.

The following is how the system calculates grouped similarities for **Color**, **Texture**, and **Shape**:

1) **Color Similarity** (S_c)

$$S_c = w_{D_c} S_{D_c} + w_{H_c} S_{H_c}$$

2) **Texture Similarity** (S_t)

$$S_t = w_G S_G + w_{H_e} S_{H_e}$$

3) **Shape Similarity** (S_s)

$$S_s = w_{M_h} S_{M_h} + w_{F_d} S_{F_d}$$

The weighted sum of the grouped similarities represents the overall similarity between a database image and a query image:

$$S = \frac{w_c S_c + w_t S_t + w_s S_s}{w_c + w_t + w_s}$$

where:

- w_c, w_t, and w_s are the general weights for **Color**, **Texture**, and **Shape** features, respectively.
- w_{D_c}, w_{H_c}, w_G, w_{H_e}, w_{M_h}, and w_{F_d} are feature-specific weights.

B. Ranking

Finally, we rank the indexed images based on similarity with the query image and return the top 15 images.

C. Flowchart

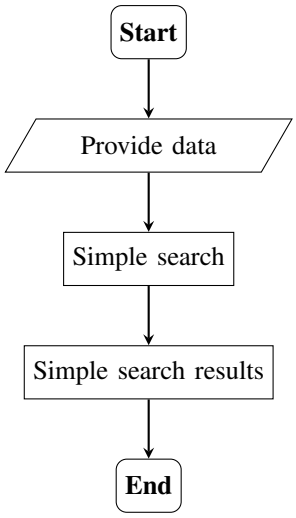

Fig. 1. Simple search flowchart.

VI. Advanced search

A. First search

We calculate the similarity at the start of an advanced search in the same manner as for the simple search, with the exception of the overall similarity which was computed as follows:

$$S = \frac{S_c + S_t + S_s}{w_{D_c} + w_{H_c} + w_G + w_{H_e} + w_{M_h} + w_{F_d}}$$

B. Relevance feedback

User-provided feedback data serves as the foundation for the relevance feedback mechanism. In order to improve the ranking of the images, the similarity weights are updated in response to the feedback.

1) Feedback Data Representation: The features and corresponding relevance label for every feedback instance i are shown as follows:

- $X_i = [x_{i,1}, x_{i,2}, ..., x_{i,n}]$ The feature vector of the image.
- $y_i \in \{0, 1\}$: The user-provided relevance label indicating whether the image is relevant ($y_i = 1$) or not ($y_i = 0$).

The total dataset from the feedback consists of:

- Feature matrix: $X \in^{m \times n}$, where m is the number of feedback samples and n is the number of features.
- Labels vector: $Y = [y_1, y_2, ..., y_n]$.

2) Bayesian Re-evaluation of Weights: Using Bayesian inference, the weights $W = [w_1, w_2, ..., w_n]$ are updated. The steps listed below provide a summary of the methodology:

a) Likelihood Function: A Bernoulli likelihood model is used to estimate the likelihood of the feedback data:

$$p(y_i|X_i, W) = \sigma(\mu_i)^{y_i} \cdot (1 - \sigma(\mu_i)^{y_i})^{1-y_i}$$

where:

$$\mu_i = X_i^T W = \sum_{j=1}^{n} x_{i,j} w_j$$

and:

$$\sigma(\mu_i) = \frac{1}{1 + e^{-\mu_i}}$$

is the sigmoid function.

The likelihood for all feedback samples is:

$$p(Y|X, W) = \prod_{i=1}^{m} \sigma(\mu_i)^{y_i} \cdot (1 - \sigma(\mu_i)^{y_i})^{1-y_i}$$

b) Prior Distribution: A Gaussian model is used to represent the weights W's prior distribution:

$$p(W) = \prod_{j=1}^{m} \mathcal{N}(w_j|\mu_j, \sigma_j^2)$$

where μ_j and σ_j^2) are the mean and variance of the prior for the j-th weight.

c) Posterior Distribution: The Bayes theorem is used to calculate the weights' posterior distribution:

$$p(W|X, Y) \propto p(Y|X, W) \cdot p(W)$$

Markov Chain Monte Carlo (MCMC) sampling is used to approximate the posterior because it cannot be handled analytically. The posterior mean is used to update the weights:

$$\hat{w} = E[p(W|X, Y)]$$

3) Updating the Weights:

a) Updating the Weights: The current similarity metric weights are used to determine the initial weights $w^{(0)}$:

$$w_j^{(0)} = initial\ weight\ for\ the\ j-th\ feature.$$

b) Updated Weights: The updated weights following Bayesian inference are as follows:

$$w_j^{(t+1)} = \hat{w}_j$$

where $\hat{w}_j$ is the mean posterior weight for the j-th feature.

c) Weight Smoothing: The weights are interpolated with the prior iteration using a learning rate α to guarantee seamless transitions between weight updates:

$$w_j^{(t+1)} = w_j^{(t)} + \alpha \cdot (\hat{w}_j - w_j^{(t)})$$

d) Normalization and Constraints: To make sure the updated weights are still valid, they are normalized and constrained:

Normalization

The weights are normalized to sum to 1:

$$\tilde{w}_j = \frac{w_j}{\sum_{k=1}^{n} w_k}$$

Constraints

The weights are constrained to falling within a certain range. $[min, max]$:

$$w_j = max(min, min(w_j, max))$$

C. Flowchart

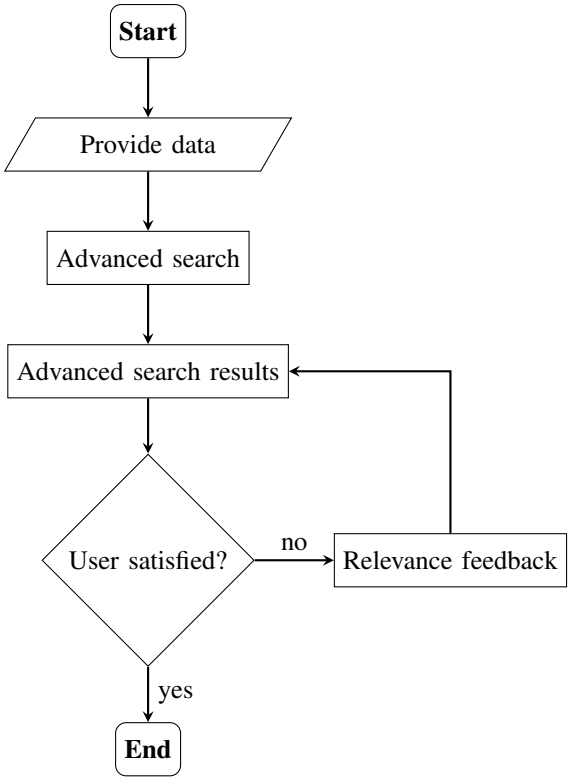

Fig. 2. Advanced search flowchart.

VII. Implementation and Demonstration

A. Description

To test our methods, we put together an application using web technologies for user accessibility, the application is session-based, meaning no prior search or usage is stored or saved. It comprises a wide range of features [24].

B. Application flowchart

C. Features

Fig. 4. Login page

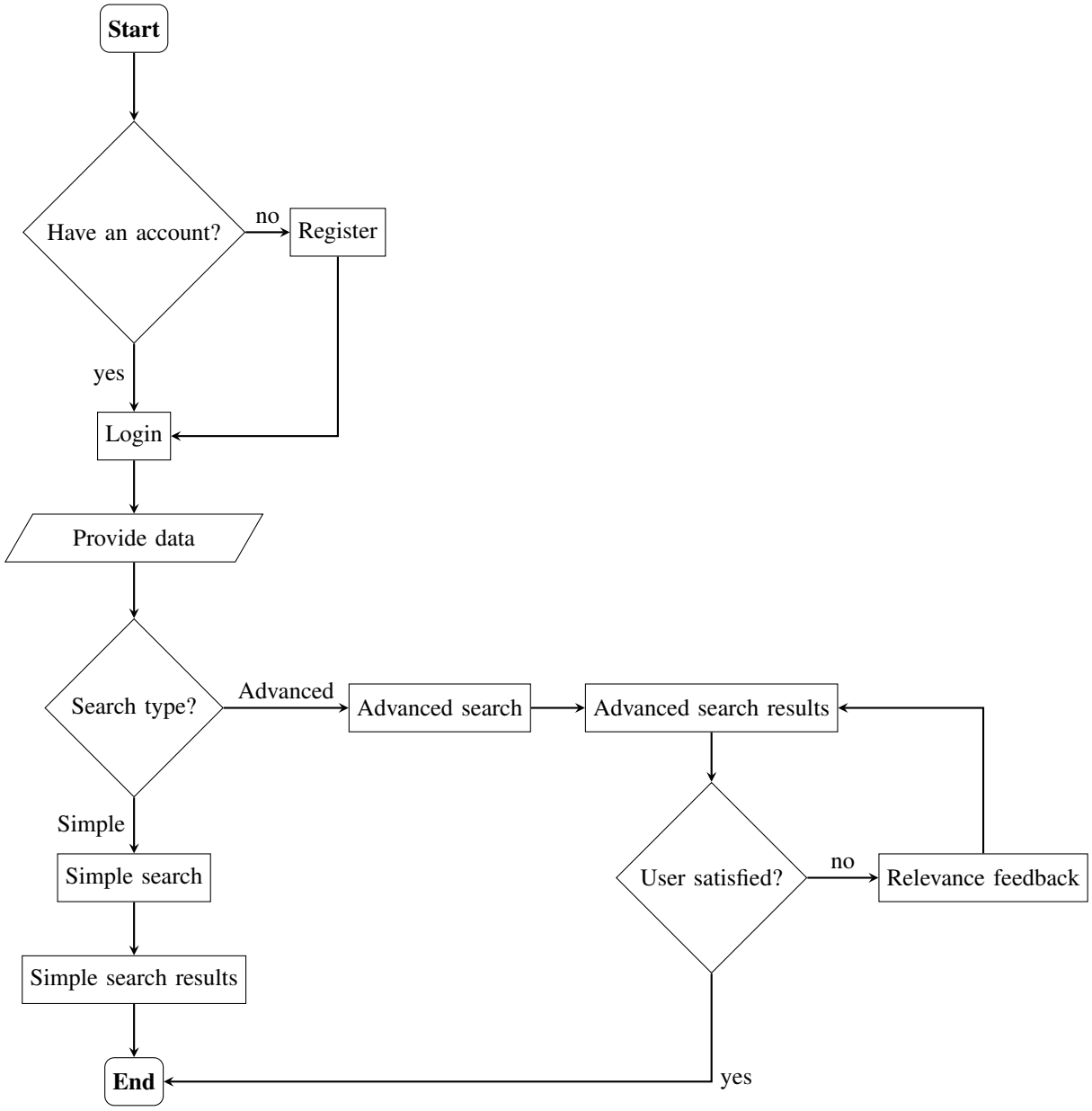

Fig. 3. Application flowchart

2) Providing and editing images: The user has complete control over how he constructs an image dataset, including the ability to upload, remove, and perform basic editing operations like cropping, rotating, flipping, and zooming.

Fig. 5. Registration page

1) Basic login and registration:

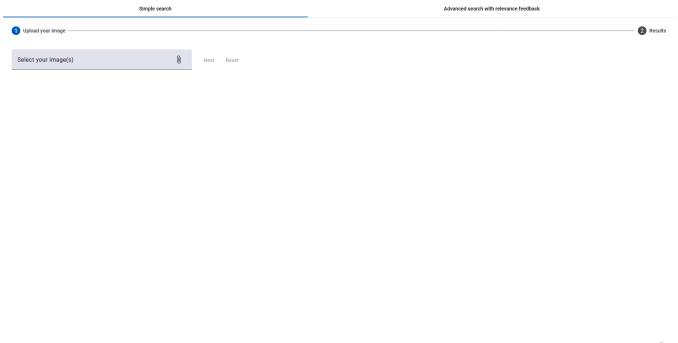

Fig. 6. Dataset building page

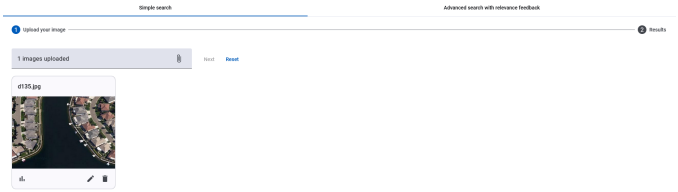

Fig. 7. Dataset building page with an image uploaded

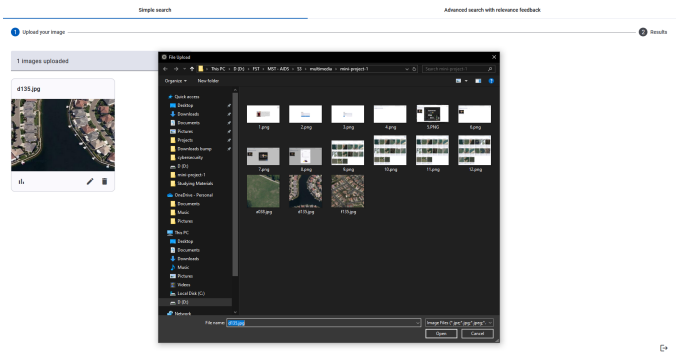

Fig. 8. Image upload demonstration

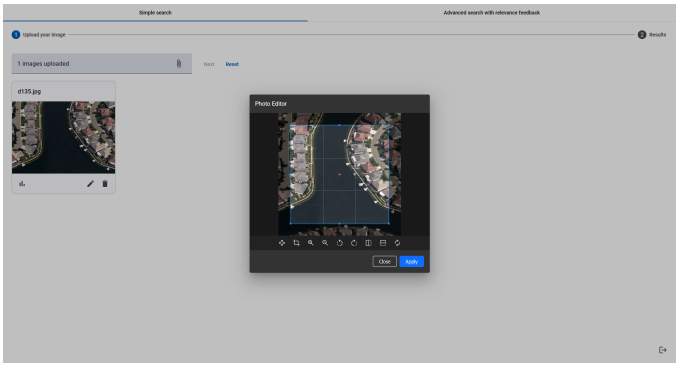

Fig. 9. Image editing demonstration

3) Consult descriptors: The histogram, dominant colors, and Hu moments for a selected image are easily available to the user.

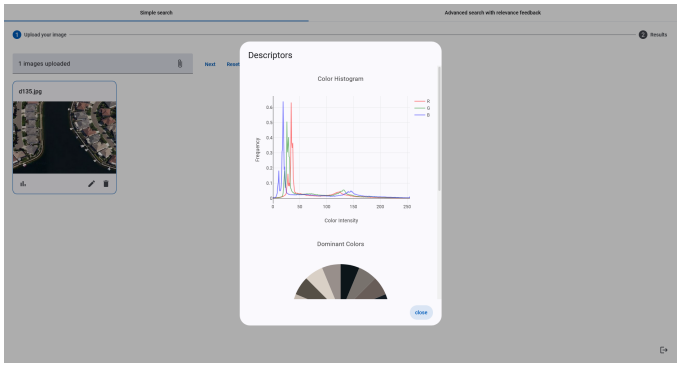

Fig. 10. Image descriptors

4) Simple search: After choosing an image, the user can use our simple search method to obtain the results.

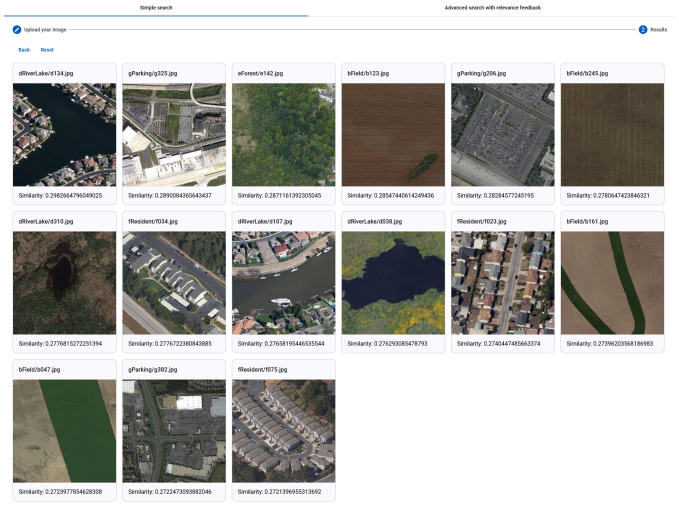

Fig. 11. Simple search results

5) Advanced search: To obtain the first search results, the user can select an image and send it to the server, just like in the simple search.

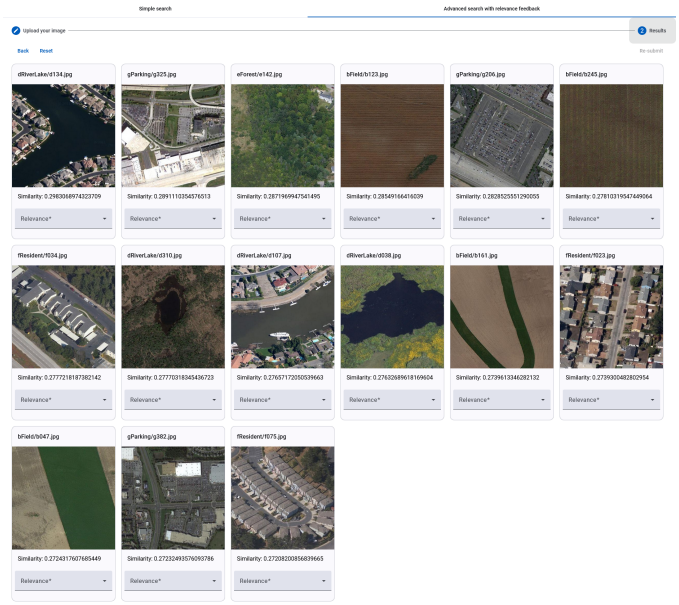

Fig. 12. First search results

After receiving the results, the user can specify which images are relevant and which are not, to send the labels back to the server and refine the search results.

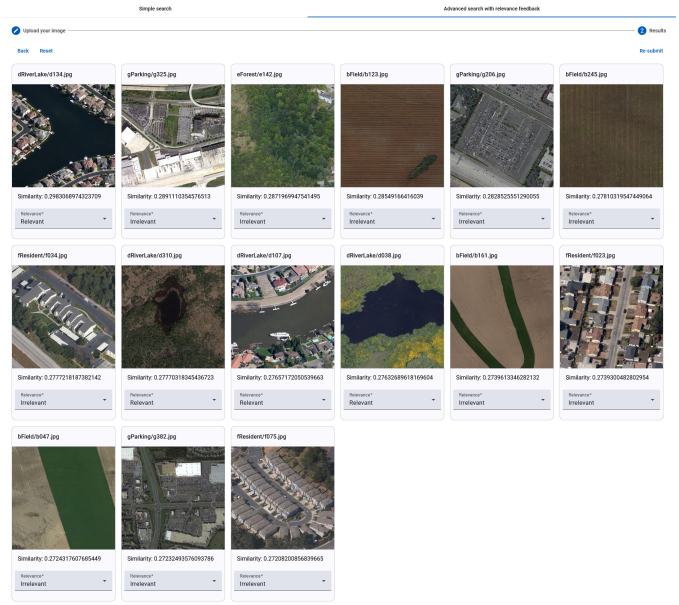

Fig. 13. Labeling search results

And then get back more coherent results.

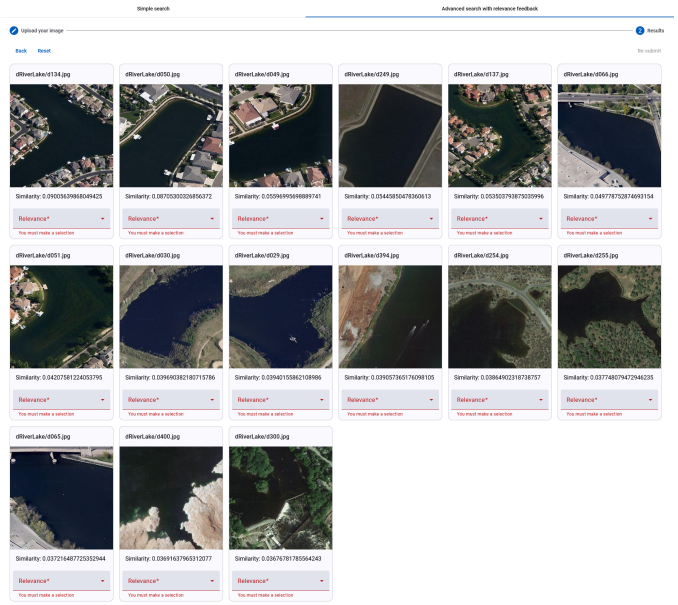

Fig. 14. Relevance feedback results

VIII. Conclusion

In this paper, we have introduced a new Relevance Feedback method based on multiple features describing general descriptors that leverages Bayesian Inference for content-based image retrieval. Following receipt of the initial search results, the user's feedback was used to refine feature weights using the Bayesian Inference method. In order to calculate the weights' posterior distribution, the feedback process contributes to the estimation of the feedback data's likelihood and supplies data for updating the Gaussian parameters that represent the weights' prior distribution. The experimental demonstration illustrates the distinction between the suggested advanced search method and the simple search method, as well as the enhancement in retrieval performance.

References

[1] Dongping Tian. A review on relevance feedback for content-based image retrieval. *Journal of Information Hiding and Multimedia Signal Processing*, 9:108–119, 01 2018.

[2] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., USA, 1986.

[3] David McG. Squire, Wolfgang Müller, Henning Müller, and Thierry Pun. Content-based query of image databases: inspirations from text retrieval. *Pattern Recognition Letters*, 21(13):1193–1198, 2000. Selected Papers from The 11th Scandinavian Conference on Image.

[4] Y. Rui, T.S. Huang, and S. Mehrotra. Content-based image retrieval with relevance feedback in mars. In *Proceedings of International Conference on Image Processing*, volume 2, pages 815–818 vol.2, 1997.

[5] Giorgio Giacinto and Fabio Roli. Bayesian relevance feedback for content-based image retrieval. *Pattern Recognition*, 37(7):1499–1508, 2004.

[6] G. Ciocca and R. Schettini. A relevance feedback mechanism for content-based image retrieval. *Information Processing Management*, 35(5):605–632, 1999.

[7] Nuno Vasconcelos and Andrew Lippman. Bayesian representations and learning mechanisms for content-based image retrieval. In *Electronic imaging*, 2000.

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

[9] Zhong Su, Hongjiang Zhang, Stan Li, and Shaoping Ma. Relevance feedback in content-based image retrieval: Bayesian framework, feature subspaces, and progressive learning. *IEEE Trans. Image Process.*, 12(8):924–937, 2003.

[10] Ahmed Talib, Massudi Mahmuddin, Husniza Husni, and Loay E. George. A weighted dominant color descriptor for content-based image retrieval. *Journal of Visual Communication and Image Representation*, 24(3):345–360, 2013.

[11] Guangyi Xie, Baolong Guo, Zhe Huang, Yan Zheng, and Yunyi Yan. Combination of dominant color descriptor and hu moments in consistent zone for content based image retrieval. *IEEE Access*, 8:146284–146299, 2020.

[12] Otávio A.B. Penatti, Eduardo Valle, and Ricardo da S. Torres. Comparative study of global color and texture descriptors for web image retrieval. *Journal of Visual Communication and Image Representation*, 23(2):359–380, 2012.

[13] J. MacQueen. Some methods for classification and analysis of multivariate observations. 1967.

[14] Umur Erkut, Faruk Bostancıoğlu, Murat Erten, Ahmet Murat Özbayoğlu, and Ercan Solak. Hsv color histogram based image retrieval with background elimination. In *2019 1st International Informatics and Software Engineering Conference (UBMYK)*, pages 1–5, 2019.

[15] H. Ali, M.I. Lali, M.Z. Nawaz, M. Sharif, and B.A. Saleem. Symptom based automated detection of citrus diseases using color histogram and textural descriptors. *Computers and Electronics in Agriculture*, 138:92–104, 2017.

[16] Zhihu Huang and Jinsong Leng. Analysis of hu's moment invariants on image scaling and rotation. In *2010 2nd International Conference on Computer Engineering and Technology*, volume 7, pages V7–476–V7–480, 2010.

[17] Youness Chawki, Khalid El Asnaoui, Mohammed Ouanan, and Brahim Aksasse. Content-based image retrieval using gabor filters and 2-d esprit method. In Mostafa Ezziyyani, Mohamed Bahaj, and Faddoul Khoukhi, editors, *Advanced Information Technology, Services and Systems*, pages 95–102, Cham, 2018. Springer International Publishing.

[18] Nitin Jain and S. S. Salankar. Content based image retrieval using gabor texture feature and color histogram. *International Journal of Enhanced Research in Science Technology Engineering,*, 3(9):97–102, September 2014.

[19] Balasubramani Ramasamy. Efficient use of mpeg-7 color layout and edge histogram descriptors in cbir systems. 9:157–163, 09 2009.

[20] Pradnya Vikhar and PP Karde. Mpeg-7 edge histogram descriptor (ehd) for advancement in cbir system. *European Journal of Advances in Engineering and Technology*, 3(10):36–39, 216AD.

[21] G.Siddartha Raghavendra, Mohammed Danish, S.Imtiyaz Khan, and S. Venkateswarlu. Fourier descriptors for shape based image retrieval. *International Journal of Engineering Research Technology (IJERT)*, 2(4), April 2013.

[22] Dengsheng Zhang and Guojun Lu. A comparative study on shape retrieval using fourier descriptors with different shape signatures. *J Vis Commun Image Represent*, 1, 01 2001.

[23] Mohamed Saad, H. Saleh, H. Konbor, and Maram Ashour. Integrated cbir using texture, fourier descriptor and color histogram. *CiiT International Journal of Digital Image Processing*, 4:625–630, 07 2012.

[24] Ahmed Samady, Fahd Chibani, and Mohamed Amine Fakhre-Eddine. Content-Based Image Retrieval system based on Bayesian Relevance Feedback, December 2024.