CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Content-based image retrieval aims to retrieve images from a database which are similar to each other in terms of their visual contents. Over the years, various feature descriptors based on color, texture and shape information have been proposed to characterize visual elements of image and video data. These features have subsequently been employed to improve the automated search and retrieval of image and videos in CBIR applications. The applications of CBIR are: users searching a particular image on the web, Various types of professionals like police force for picture recognition in crime prevention, Medicine diagnosis, Architectural and engineering design, Fashion and publishing, Geographical information, remote sensing systems and home entertainment. However, selecting and developing methods for correctly identifying and effectively integrating suitable visual features for a specific vision task remain a very challenging problem.

The proposed method uses support vector machines for retrieval of images based on content. First, 190 dimensional feature vectors are extracted and similarity metrics is defined. Using the SVM learning techniques, the images which are similar to the query image are retrieved.

1.1.1 CLASSICAL TECHNIQUES OF IMAGE RETRIEVAL

The classic image retrieval techniques are based on two types of visual features: global features and local features. Global features-based algorithms aim at the whole image as visual content, e.g. color, texture, shape and local features-based algorithms focus mainly on key points or salient patches. Various algorithms have been designed for extracting global and local features. In the MPEG-7 standard, the color descriptors consist of a number of histogram descriptors, such as dominant color descriptor, color layout descriptor and scalable color descriptor. Various algorithms have been developed for texture analysis in some literatures, such as the gray level co-occurrence matrices, Tamura texture feature, Markov random field model, Gabor filtering, local binary pattern, etc. There are three texture descriptors in the MPEG-7 standard: homogeneous texture descriptor, texture browsing descriptor and the edge histogram descriptor. Texture features can be combined with color features to improve the discrimination power and can obtain better retrieval performance. There are some algorithms which can ultimately combine color and texture together, such as texton co occurrences matrix, multi-texton histogram, color edge co-occurrence histogram, micro-structure descriptor, color difference histogram etc. The classical representations of shape feature include moment invariants, Fourier transforms coefficients, edge curvature and arc length. In MPEG-7 standard, three shape descriptors are used for object-based image retrieval: 3-D shape descriptor, region based shape descriptor and curvature scale space descriptor. In many cases, shape feature extraction need image segmentation which is still an open problem and limited its application in many fields, thus many researchers have adopted local features (e.g., keypoints, salient patches) instead of using the traditional shape features. There are many famous keypoints detectors and descriptors, such as Harris keypoints detector, SIFT, SURF, PCA-SIFT and ORB, where SIFT is the most popular local feature representation. It can be used to perform reliable matching between different views of an object or scene. In order to perform as good as SIFT with lower computational complexity the SURF or ORB can be considered as an efficient alternative to SIFT. Recently, bag-of-visual words models or its variants have been reported in the literatures and used for object-based image retrieval, object recognition and scene categorization. Sivic and Zisserman have proposed the bag-of-visual words model which in essence borrows techniques from text retrieval. In BOW model, local features extracted from an image by using SIFT, SURF or other keypoint detectors, and then mapped into a set of visual words. Finally, an image is represented as a histogram of visual word occurrences. It is so called the standard BOW baseline and can be considered as one of state-of-the-art methods. Since the visual words usually come from clustering implementation which needs heavy computational burdens. Besides, visual words have two major limitations that the lack of any explicit semantic meanings and the ambiguity of visual words. Indeed, improving the visual vocabulary, incorporating spatial information and semantic attributes can reduce the limitations and can also improve the performances of BOW models. SVM technique can be employed in CBIR for better retrieval of images compared to the existing models.

1.1.2 SVM

Support Vector Machine are supervised learning models used among others for classification and regression analysis. They were introduced in 1992 in the Conference on Learning Theory by Boser, Guyon and Vapnik. It became quite popular since then because it is a theoretically well motivated algorithm which was developed since the 60s from Statistical Learning Theory (Vapnik and Chervonenkis) and it also holds good empirical performance in a diverse number of scientific fields and application domains such as bioinformatics, text and image recognition, music retrieval and many more. SVMs are based on the idea of separating data with a large “gap” also know as margins. Optimal marking classifier will act as stepping stone for the introduction to Lagrange duality. Another aspect of SVM which is important is the notion of kernels which allow SVMs to be applied efficiently in high dimensional feature spaces. Let’s start by setting up our problem. The context is known from before where the images from different classes are to be classified accordingly. In other words this is a binary classification problem. Based on this classification image similar to the query image can be retrieved. Figure 1.1 depicts two classes of images, the positive and negative. For the sake of the example let’s consider the circles to be the positive and the triangles to be the negative. A hyper plane separates them and the three labeled data points. Notice that point A is the furthest from the decision boundary. In order to make a prediction for the value of the label $\overrightarrow{yi}$ at point A, the value of the label is going to be $\overrightarrow{yi}$= 1. On the other hand, point C even though it is on the correct side of the decision boundary predict a label value of $\overrightarrow{yi}$ = 1, a small change to the decision boundary could have caused the prediction to be negative $\overrightarrow{yi}$ = −1. Prediction at A is more confident than at C. Point B lies in-between these two cases. One can extrapolate and say that if a point is far from the separating hyperplane, predictions can be more confident. Given a training set, find a decision boundary that allows to make the correct and confident predictions (i.e. far from the decision boundary) on the training examples.

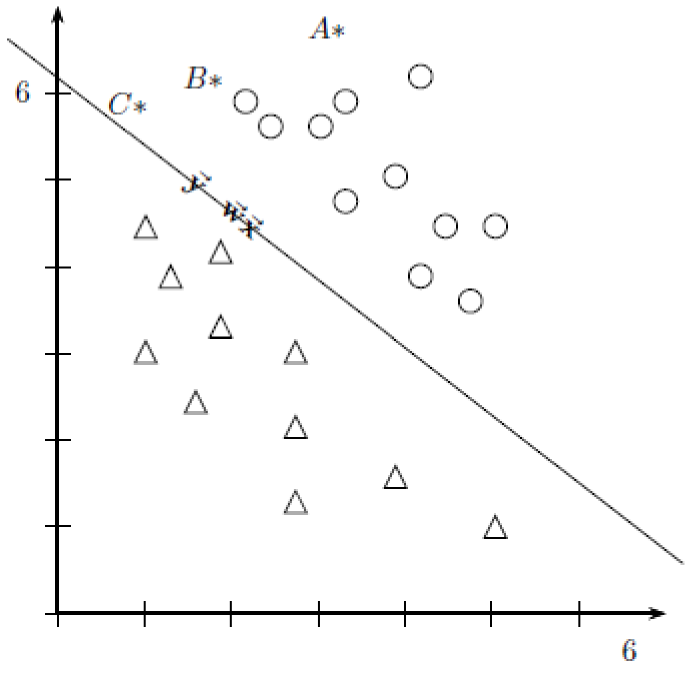


Figure 1.1: Images projected on a 2D plane.

Let's consider the binary classification problem where labels $\vec{y} \in \{-1, 1\}$ and features $\vec{x}$. Then binary classifier might look like

$$h_{\vec{w},\vec{b}}(\vec{x}) = g(\vec{w}^T\vec{x} + \vec{b}) \begin{cases} g(z) = 1 \text{ if } z \geq 0, \\ g(z) = 0 \text{ otherwise.} \end{cases}$$

Now distinguish between two different notions of margin such as functional and geometric margin. The functional margin of ($\vec{w}$,$\vec{b}$) with respect to the training example ($\overrightarrow{xi}$,$\overrightarrow{yi}$) is

$$\hat{\vec{\gamma}}_i = \vec{\mathbf{y}}_i(\vec{\mathbf{w}}^T\vec{\mathbf{x}} + \vec{\mathbf{b}})$$

If $\overrightarrow{yi}$ = 1, then for prediction to be confident and correct (i.e. the functional margin to be large), $\vec{w}^{T}\vec{x}+\vec{b}$, needs to be a large positive number. If $\overrightarrow{yi}$ = −1, then for the functional margin to be large (i.e. to make a confident and correct prediction) $\vec{w}^{T}\vec{x}+\vec{b}$ needs to be a large negative number. Note that if $\vec{w}$ is replaced with 2$\vec{w}$ and $\vec{b}$ with 2$\vec{b}$, then since g($\vec{w}^{T}\vec{x}+\vec{b}$) = g(2$\vec{w}^{T}\vec{x}$+2$\vec{b}$), would not change $\vec{w}$, $\vec{b}$ at all, which means that it depends only on the sign, but not on the magnitude of $\vec{w}^{T}\vec{x}+\vec{b}$. The geometric margin, is interpreted using Figure 1.2.

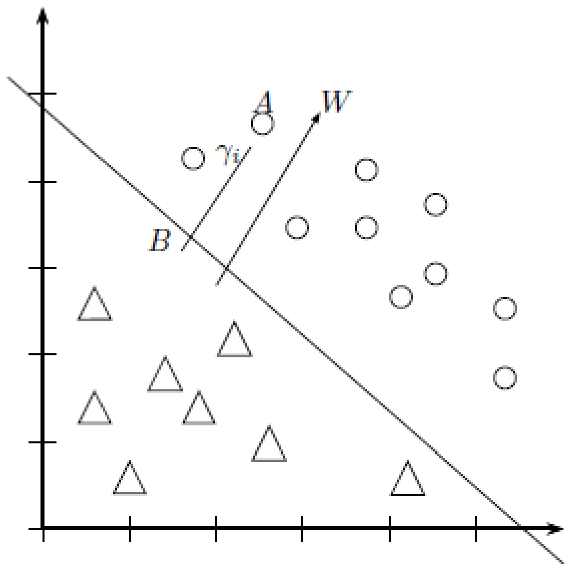


Figure 1.2: Interpretation of geometric margin.

The decision boundary corresponding to ($\vec{w}$,$\vec{b}$) along with the orthogonal vector $\vec{w}$. Point A resembles some training example $\vec{b}$ with label $\overrightarrow{yi}$ = 1. The distance to the decision boundary denoted by γi is given by the line segment AB. If we consider $\vec{w}$ / $||\vec{w}||$ to be a unit-length vector pointing in the same direction as $\vec{w}$ then point B = $\overrightarrow{xi}$ − γi ·$\vec{w}$ / $||\vec{w}||$. Since this point lies on the decision boundary then it satisfies that $\vec{w}^T \vec{x} + \vec{b} = 0$, which means that all points $\vec{x}_i$ on the decision boundary satisfy the same equation.

Usually the geometric margin of ($\vec{w}$,$\vec{b}$) with respect to the training example ($\overrightarrow{xi}$,$\overrightarrow{yi}$) is defined to be

$$\vec{\gamma}_i = \left(\left(\frac{\vec{\mathrm{w}}}{||\vec{\mathrm{w}}||}\right)^T + \frac{\vec{\mathrm{b}}}{||\vec{\mathrm{w}}||}\right)$$

If $||\vec{w}||$ = 1, then the functional margin is equal to the geometric margin. Notice also that the geometric margin is invariant to rescaling of the parameters (i.e. if $\vec{w}$ is replaced with 2$\vec{w}$ and $\vec{b}$ with 2$\vec{b}$, then the geometric margin does not change). This way it is possible to impose an arbitrary scaling constraint on $\vec{w}$ without changing anything significant from our original equation. Given a dataset D = {($\overrightarrow{xi}$,$\overrightarrow{yi}$); , i = 1, . . . ,N}, it is also possible to define the geometric margin of ($\vec{w}$,$\vec{b}$) with respect to D to be the smallest of geometric margins on the individual training examples γ = mini=1,...,N $\vec{\gamma}$. Thus, the goal for classifier is to find a decision boundary that maximizes the geometric margin in order to reflect a confident and correct set of predictions, resulting in a classifier that separates the positive and negative training examples with a geometric margin. Supposed that the training data are linearly separable, find a separating hyperplane that achieves the maximum geometric margin. Start by posing the following optimization problem

$$\begin{aligned} &\max_{\vec{\gamma},\vec{\mathrm{w}},\vec{\mathrm{b}}} \vec{\gamma} \\ &\text{s.t. } \vec{\mathrm{y}}_i(\vec{\mathrm{w}}^T\vec{\mathrm{x}}_i + \vec{\mathrm{b}}) \geq \vec{\gamma},\ i = 1, \ldots, N \\ &||\vec{\mathrm{w}}|| = 1 \end{aligned}$$

The $||\vec{w}||$ = 1 constraint ensures that the functional margin equals to the geometric margin, in this way it is guaranteed that all the geometric margins are at least $\vec{\gamma}$. Since "$||\vec{w}|| = 1$" constraint is a non-convex one, and this is hard to solve instead what try to transform the problem into an easier one. Consider

$$\max_{\hat{\vec{\gamma}},\vec{\mathbf{w}},\vec{\mathbf{b}}} \frac{\hat{\vec{\gamma}}}{||\vec{\mathbf{w}}||}$$
$$\text{s.t. } \vec{\mathbf{y}}_i(\vec{\mathbf{w}}^T\vec{\mathbf{x}}_i + \vec{\mathbf{b}}) \geq \hat{\vec{\gamma}},\ i = 1, \ldots, N$$

Notice that the constraint $||\vec{w}|| = 1$ is ridden making the objective difficult and also since $\vec{\gamma} = \vec{\gamma} / ||\vec{w}||$, will provide an acceptable and correct answer. The main problem is that still objective function $\vec{\gamma}$ / $||\vec{w}||$ is non-convex, thus keep searching for a different representation. Recall that adding an arbitrary scaling constraint on $\vec{w}$ and $\vec{b}$ without changing anything from original formulation. Introducing the scaling constraint such that the functional margin of $(\vec{w},\vec{b})$ with respect to the training set $(\vec{xi},\vec{yi})$ must be 1, this is $\vec{\gamma}$ = 1. Multiplying $(\vec{w},\vec{b})$ by some constant yields the functional margin multiplied by the same constant. One can satisfy the scaling constraint by rescaling $(\vec{w}, \vec{b})$. Plugging this constraint into Equation and substitute $\vec{\gamma}$ / $||\vec{w}||$ = 1 / $||\vec{w}||$ then following optimization problem is obtained.

$$\min_{\hat{\vec{\gamma}},\vec{\mathbf{w}},\vec{\mathbf{b}}} \frac{1}{2}||\vec{\mathbf{w}}||^2$$
$$\text{s.t. } \vec{\mathbf{y}}_i(\vec{\mathbf{w}}^T\vec{\mathbf{x}}_i + \vec{\mathbf{b}}) \geq 1,\ i = 1, \ldots, N$$

Notice that maximizing $\vec{\gamma}$ / $||\vec{w}||$ = 1 / $||\vec{w}||$ is the same thing as minimizing $||\vec{w}||^2$. Now transform optimization problem into one with a convex quadratic objective and linear constraints which can be solved using quadratic programming. The solution to the above optimization problem will give us the optimal margin classifier which will lead to the dual form of our optimization problem, which in return plays an important role in the use of kernels to get optimal margin classifiers, in order to work efficiently in very high dimensional spaces. Re express the constraints as $g_i(\vec{w}) = -\vec{yi}(\vec{w}^T \vec{x} + \vec{b}) + 1 \leq 0$. Notice that constraints that hold with equality, $g_i(\vec{w}) = 0$, correspond to training examples $(\vec{xi}, \vec{yi})$, that have functional margin equal to one.

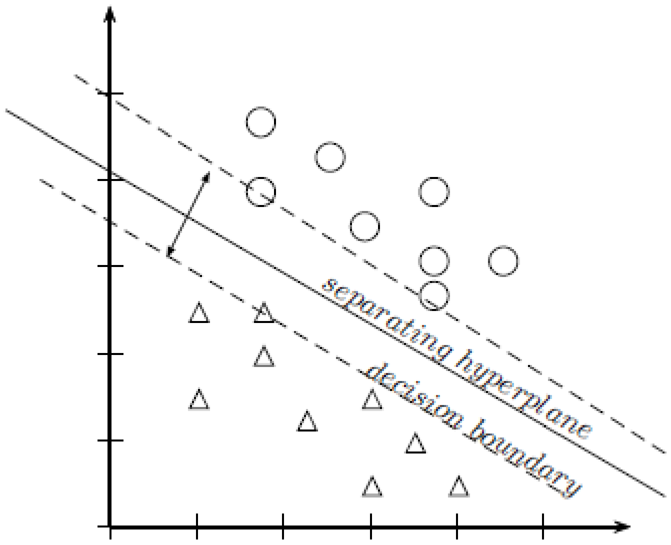


Figure 1.3: Support vectors and maximum margin separating hyperplane

The three points that lie on the decision boundary (two positive and one negative) are the ones with the smallest margins and thus closest to the decision boundary. Notice that these three points are called support vectors and usually they can be smaller in number than the training set. In order to tackle the problem Lagrangian optimization problem is framed.

$$\mathcal{L}(\alpha, \vec{\mathrm{w}}, \vec{\mathrm{b}}) = \frac{1}{2}\|\vec{\mathrm{w}}\|^2 - \sum_{i}^{N} \alpha_i \left[\vec{\mathrm{y}}_i \left(\vec{\mathrm{w}}^T \vec{\mathrm{x}}_i + \vec{\mathrm{b}}\right) - 1\right].$$

with only one Lagrangian mulptiplier “αi” since the problem has only inequality constraints and not any equality constraints. First, find the dual form of the problem, to do so minimize $L(\alpha, \vec{w}, \vec{b})$ with respect to $\vec{w}$ and $\vec{b}$ for a fixed α. Setting the derivatives of L with respect to $\vec{w}$ and $\vec{b}$ to zero, the below equation is obtained.

$$\nabla_{\vec{\mathrm{w}}}\mathcal{L}(\alpha, \vec{\mathrm{w}}, \vec{\mathrm{b}}) = \vec{\mathrm{w}} - \sum_{i=1}^{N} \alpha_i \vec{\mathrm{y}}_i \vec{\mathrm{x}}_i = 0.$$

$$\vec{\mathrm{w}} = \sum_{i=1}^{N} \alpha_i \vec{\mathrm{y}}_i \vec{\mathrm{x}}_i. \left\{ \text{derivative of } \frac{\partial \mathcal{L}}{\partial \vec{\mathrm{w}}} \right.$$

$$\frac{\partial \mathcal{L}(\alpha, \vec{\mathrm{w}}, \vec{\mathrm{b}})}{\partial \vec{\mathrm{b}}} = \sum_{i=1}^{N} \alpha_i \vec{\mathrm{y}}_i = 0. \left\{ \text{derivative of } \frac{\partial \mathcal{L}}{\partial \vec{\mathrm{b}}} \right.$$

Substituting and simplifying

$$\mathcal{L}(\alpha, \vec{\mathrm{w}}, \vec{\mathrm{b}}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \vec{\mathrm{y}}_i \vec{\mathrm{y}}_j \alpha_i \alpha_j (\vec{\mathrm{x}}_i)^T \vec{\mathrm{x}}_j - \vec{\mathrm{b}} \sum_{i=1}^{N} \alpha_i \vec{\mathrm{y}}_i.$$

$$\mathcal{L}(\alpha, \vec{\mathrm{w}}, \vec{\mathrm{b}}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \vec{\mathrm{y}}_i \vec{\mathrm{y}}_j \alpha_i \alpha_j (\vec{\mathrm{x}}_i)^T \vec{\mathrm{x}}_j.$$

Utilizing the constraint αi ≥ 0 and the constraint the following dual optimization problem arises:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \vec{\mathrm{y}}_i \vec{\mathrm{y}}_j \alpha_i \alpha_j \langle \vec{\mathrm{x}}_i, \vec{\mathrm{x}}_j \rangle$$

$$\text{s.t. } \alpha_i \geq 0, i = 1, \ldots, N$$

$$\sum_{i=1}^{N} \alpha_i \vec{\mathrm{y}}_i = 0.$$

Find the α that maximizes $\vec{w}$(α) in order to find the optimal $\vec{w}$ as a function of α. Once the optimal $\vec{w}*$ is found, consider the primal problem then find the optimal value for the intercept term $\vec{b}$.

$$\vec{\mathrm{b}}* = -\frac{\max_{i:\vec{\mathrm{y}}_i=-1} \vec{\mathrm{w}}*^T \vec{\mathrm{x}}_i + \min_{i:\vec{\mathrm{y}}_i=1} \vec{\mathrm{w}}*^T \vec{\mathrm{x}}_i}{2}$$

Suppose the parameters of model to a training set is fit, and a prediction at a new point input $\vec{x}$ is made, then calculate $\vec{w}^{T}\vec{x}+\vec{b}$, and predict $\vec{y} = 1$ if and only if this quantity is bigger than zero. But this quantity can also be written:

$$\begin{aligned}\vec{\mathrm{w}}^T\vec{\mathrm{x}}+\vec{\mathrm{b}} &= \left(\sum_{i=1}^{N}\alpha_i\vec{\mathrm{y}}_i\vec{\mathrm{x}}_i\right)^T\vec{\mathrm{x}}+\vec{\mathrm{b}} \\ &= \sum_{i=1}^{N}\alpha_i\vec{\mathrm{y}}_i\left\langle\vec{\mathrm{x}}_i,\vec{\mathrm{x}}\right\rangle+\vec{\mathrm{b}}\end{aligned}$$

Earlier the different values for αi will all be zero except for the support vectors. Many of the terms in the sum above will be zero. Find only the inner products between $\vec{x}$ and the support vectors in order to calculate and make our prediction. Exploit this property of using inner products between input feature vectors in order to apply kernels to the classification problem. For kernels, input data is needed. Images are usually described by a number of pixel values which is referred to as attributes, indicating the different intensity colors across the three different color channels {R,G,B}. Process the pixel values in order to retrieve more meaningful representations, in other words map our initial pixel values through some processing operation to some new values, these new values are called features and the operation process is referred to as feature mapping usually denoted as φ. Instead of applying SVM directly to the attributes $\vec{x}$, use SVM to learn from some features $\varphi(\vec{x}i)$. Since the SVM algorithm can be written entirely in terms of inner products $<\vec{x},\vec{z}>$ instead replace them with $<\varphi(\vec{x}), \varphi(\vec{z})>$. This way given a feature mapping φ, the corresponding kernel is defined as $K<\vec{x},\vec{z}> = <\varphi(\vec{x})^{T}, \varphi(\vec{z})>$. If every inner product $<\vec{x},\vec{z}>$ is replaced in the algorithm with $K<\vec{x},\vec{z}>$, then the learning process will be happening using features φ. Compute $K<\vec{x},\vec{z}>$ by finding $\varphi(\vec{x})$ and $\varphi(\vec{z})$ even though they may be expensive to calculate because of their high dimensionality. Kernels such as $K<\vec{x},\vec{z}>$, allows SVMs to perform learning in high dimensional feature spaces without the need to explicitly find or represent vectors $\varphi(\vec{x})$. For instance, suppose < $\vec{x}$,$\vec{z}$> ∈ Rn, and let's consider K< $\vec{x}$,$\vec{z}$> = < $\vec{x}$,$\vec{z}$> which is equivalent to

$$\begin{aligned} K(\vec{x},\vec{z}) &= \left(\sum_{i=1}^{n} \vec{x}_i\vec{z}_i\right)\left(\sum_{j=1}^{n} \vec{x}_j\vec{z}_j\right) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \vec{x}_i\vec{x}_j\vec{z}_i\vec{z}_j \\ &= \sum_{i,j=1}^{n} (\vec{x}_i\vec{x}_j)(\vec{z}_i\vec{z}_j) \\ &= \phi(\vec{x})^T\phi(\vec{z}) \end{aligned}$$

for n = 3 the feature mapping φ is computed as

$$\phi(\vec{x}) = \begin{bmatrix} x_1x_1 \\ x_1x_2 \\ x_1x_3 \\ x_2x_1 \\ x_2x_2 \\ x_2x_3 \\ x_3x_1 \\ x_3x_2 \\ x_3x_3 \end{bmatrix}$$

A kernel K($\vec{x}$,$\vec{z}$) = ($\vec{x}$,$\vec{z}$ + c)^d corresponds to a feature mapping of feature space. K($\vec{x}$,$\vec{z}$) still takes time even though it is operating in a space, because it doesn't need to explicitly represent feature vectors in this high dimensional space. If K($\vec{x}$,$\vec{z}$) is thought of as some measurement of how similar are φ($\vec{x}$) and φ($\vec{z}$), or $\vec{x}$ and $\vec{z}$, then K($\vec{x}$,$\vec{z}$) = φ($\vec{x}$)$_{T\,.}$ φ($\vec{z}$) is expected to be large, if φ($\vec{x}$) and φ($\vec{z}$) are close together and vice versa. Suppose that for some learning problem of some kernel function K($\vec{x}$,$\vec{z}$) is considered as a reasonable measure of how similar $\vec{x}$ and $\vec{z}$ are. For instance,

$$K(\vec{x},\vec{z}) = e^{-\frac{\|\vec{x}-\vec{z}\|^2}{2\sigma^2}} \begin{cases} 1 \text{ if } \vec{x} \text{ and } \vec{z} \text{ is close} \\ 0 \text{ otherwise} \end{cases}$$

the question then becomes, can this definition be used as the kernel in an SVM algorithm. In general, given any function K is there any process which will allow to describe if it exists some feature mapping φ so that $K(\vec{x},\vec{z}) = \varphi(\vec{x})^T\varphi(\vec{z})$ for all $\vec{x}$ and $\vec{z}$. If suppose that K is a valid kernel then $\varphi(\vec{x}_i)^T\varphi(\vec{x}_j) = \varphi(\vec{x}_j)^T\varphi(\vec{x}_i)$, meaning that the kernel matrix denoted as K_m, describing similarity between datum $(\vec{x}_i)$ and $(\vec{x}_j)$, must be symmetric. Denote $\varphi k(\vec{x})$, the k-the coordinate of the vector $\varphi(\vec{x})$, then for any vector $\vec{z}$, this is obtained.

$$\begin{aligned}
\vec{z}^T K \vec{z} &= \sum_i \sum_j \vec{z}_i K_{ij} \vec{z}_j \\
&= \sum_i \sum_j \vec{z}_i \phi_k(\vec{x}_i)^T \phi_k(\vec{x}_j) \vec{z}_j \\
&= \sum_i \sum_j \vec{z}_i \sum_k \phi_k(\vec{x}_i) \phi_k(\vec{x}_j) \vec{z}_j \\
&= \sum_k \sum_i \sum_j \vec{z}_i \phi_k(\vec{x}_i) \phi_k(\vec{x}_j) \vec{z}_j \\
&= \sum_k \left(\sum_i \vec{z}_i \phi_k(\vec{x}_i) \right)^2 \\
&\geq 0.
\end{aligned}$$

which shows that the kernel matrix K_m is positive semi-definite $(K \geq 0)$ since the choice of $\vec{z}$ was arbitary. If K is a valid kernel meaning that it corresponds to some feature mapping φ, then the corresponding kernel matrix $K_m \in R^{mxm}$ is symmetric positive semi-definite. This is a necessary and sufficient condition for K_m to be a valid kernel also called the Mercer kernel. The message is that if any algorithm that is written in terms of only inner products $< \vec{x},\vec{z}>$ between the input attribute vectors, then by replacing it with a kernel $K< \vec{x},\vec{z}>$ the algorithm can be permitted to work efficiently in the high dimensional feature space.

1.2 PROJECT OBJECTIVE

The objective is to retrieve images from a database which are similar to each other in terms of their content. Selecting methods for correctly identifying and effectively integrating suitable features for a image to retrieve similar images from the database.

- Select a query image.
- Extract features of the query image and the candidate images.
- Apply existing method to find the image similarity and retrieve images.
- Apply SVM for classification and retrieve similar images.
- Compare the existing and proposed method.

1.3 SOFTWARE DESCRIPTION

1.3.1 ABOUT MATLAB

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects, which together represent the state-of-the-art in software for matrix computation. MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation.

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar non interactive language such as C or Fortran. MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M -files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation and many others.

1.3.2 THE MATLAB SYSTEM

The MATLAB system consists of five main parts:

1.3.3 THE MATLAB LANGUAGE

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

1.3.4 THE MATLAB WORKING ENVIRONMENT

This is the set of tools and facilities that you work with as the MATLAB user or programmer. It includes facilities for managing the variables in your workspace and importing and exporting data. It also includes tools for developing, managing, debugging, and profiling M-files, MATLAB's applications.

1.3.5 HANDLE GRAPHICS

This is the MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level commands that allow you to fully customize the appearance of graphics as well as to build complete Graphical User Interfaces on your MATLAB applications.

1.3.6 THE MATLAB MATHEMATICAL FUNCTION LIBRARY

This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigen values, Bessel functions, and fast Fourier transforms.

1.3.7 THE MATLAB APPLICATION PROGRAM INTERFACE (API)

This is a library that allows you to write C and Fortran programs that interact with MATLAB. It include facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

1.3.8 USES OF MATLAB

MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams. It is used in a range of applications including:

- Signal Processing and Communications
- Image and Video Processing
- Control Systems
- Test and Measurement
- Computational Finance
- Computational Biology

1.4 SOFTWARE AND HARDWARE REQUIREMENTS

- ➢ Operating System : Windows 7 / Windows 8 / Windows 10
- ➢ Processors : Any Intel or AMD X86 processor
- ➢ Disk Space : 2-3 GB
- ➢ RAM : 2 GB

CHAPTER 2

LITERATURE SURVEY

2.1)Yan Zhou, Fan-Zhi Zeng, Hui-min Zha, Paul Murray, Jinchang Ren "Hierarchical Visual Perception and Two-Dimensional Compressive Sensing for Effective Content-Based Color Image Retrieval", April 2016 describes hierarchical visual structuring, color perception and compressed sensing to perform CBIR. Once the image has been converted from RGB to HSV color space, it performs discrete cubic partitioning. Then, by introducing hierarchical operators and defining SGLCM in the HSV space, the 2D CS measurement model is used to extract two classes of hierarchical features. One of them, known as ''hierarchical HSV features,'' reflects the image's color and the positional relationship among pixels. The other is known as ''SGLMC texture features'' and these accurately describe and facilitate the comparison of texture features between images. The similarity among images is computed by fusion of the two hierarchical features and the traditional GLMC and SGLCM texture features. The computational complexity was less but it requires empirical setting of parameters in the feature fusion step and also its performance was inconsistent for different query images.

2.2) Ji Wan, Dayong Wang, Steven C.H. Hoi, Pengcheng Wu, Jianke Zhu, Yongdong Zhang, Jintao Li "Deep learning for content-based image retrieval: a comprehensive study", November 2014 describes a deep learning framework for CBIR by training large-scale deep convolutional neural networks for learning effective feature representations of images. It consists of two stages: training a deep learning model from a large collection of training data and applying the trained deep model for learning feature representations of CBIR tasks in a new domain. The performance is inconsistent and retrieval time is high. By retraining the deep models with classification or similarity learning objective on the new domain, the retrieval performance could be boosted significantly which is much better than the improvements made by shallow similarity learning.

2.3) Hua Wang, Feiping Nie, Heng Huang, Chris Ding "Heterogeneous visual features fusion via sparse multimodal machine", 2013 describes a sparse multimodal learning approach to combine features using joint structured sparsity regularizations. It is a Sparse Multimodal Learning method to integrate different types of visual features for scene and object classifications. Instead of learning one parameter for all features from one modality as in multiple kernel learning, this method learns the parameters for each feature on different classes via the joint structured sparsity regularizations. The combined convex regularizations consider the importance of both feature modality and individual feature. The natural property of sparse regularization automatically identifies the important visual features for different visual recognition tasks. They derived an efficient optimization algorithm to solve the non-smooth objective and provided a rigorous proof on its global convergence. The recall rate is high and problem exists with correlation also.

2.4) Liang Zheng, Shengjin Wang, Wengang Zhou, Qi Tian "Bayes merging of multiple vocabularies for scalable image retrieval", 2014 describes a Bayesian merging approach to down-weight the indexed feature in the intersection set to overcome the correlation problem. The Bayes merging approach explicitly estimates the matching strength of the indexed features in the intersection sets, while preserving those in the difference set. In a probabilistic view, Bayes merging is capable of jointly modeling an image and feature level similarity from multiple sets of indexed features. Specifically, it exploits the probability that an indexed feature is a true match (both locally and globally) if it is located in the intersection sets of multiple inverted files. Bayes merging effectively reduces the impact of vocabulary correlation, thus improving the retrieval accuracy significantly. But for large databases, vocabularies are never large enough, so the correlation problem would be more severe in the large-scale case.

2.5) Liang Zheng, Shengjin Wang, Ziqiong Liu, Qi Tian "Packing and padding: coupled multi-index for accurate image retrieval", 2014 describes a coupled multi-index framework to overcome information loss using SIFT. Each keypoint in the image is described by both SIFT and color descriptors. Two distinct features are then coupled into a multi-index, each as one dimension. c-MI enables indexing-level feature fusion of SIFT and color descriptors, so the discriminative power of BoW model is greatly enhanced. To overcome the illumination changes and improve recall, a large multiple assignment is used for color feature. Moreover, c-MI is efficient in terms of both memory and time (about half compared to the baseline) costs, thus suitable for large scale settings. Since c-MI can be extended to include other local descriptors, different feature selection strategies can be employed. This approach improves the recall rate. The code book has to be trained which is a disadvantage.

2.6) Sadegh Fadaei, Rassoul Amirfattahi, Mohammad Reza, Ahmadzadeh "A New Content-Based Image Retrieval System Based on Optimized Integration of DCD, Wavelet and Curvelet Features", 2016 describes a feature combination method with similarity measure for CBIR. It is based on the optimized combination of the color and texture features to enhance the image retrieval precision. It focuses on a uniform partitioning scheme which is applied in the HSV color space to extract Dominant Color Descriptor features. The DCD features are initially extracted as the color features and then an appropriate similarity measure is applied. Also, several wavelet and curvelet features are defined as texture features to overcome the noise and the problem of image translation. Finally, the color and texture features are optimally combined by using the particle swarm optimization algorithm. Not only the proposed color, wavelet and curvelet features outperform the existing ones, but also their optimum combination has a better accuracy in comparison with several contemporary CBIR systems. Other feature types such as shape features could be applied for CBIR systems. Furthermore, a segmentation scheme could be used to extract the features from more important regions of the image instead of the whole image. Finally, the most relevant features among the whole feature set could be selected instead of optimal features combination.

2.7) Ray-I Chang, Shu-Yu Lin, Jan-Ming Ho, Chi-Wen Fann, Yu-Chun Wang "A Novel Content Based Image Retrieval System using K-means/KNN with Feature Extraction", December 2012 describes a novel system architecture for CBIR system which combines techniques include content-based image and color analysis, as well as data mining techniques. It uses segmentation and grid module, feature extraction module, K-means and k-nearest neighbor clustering algorithms and bring in the neighborhood module to build the CBIR system. Concept of neighborhood color analysis module which also recognizes the side of every grids of image is developed. To build a generalized query method which increase the system searching ability and provide more accurate content descriptions of places of interest by performing color feature analysis and CCH image extraction simultaneously is a future work.

2.8) Priyanka Sharma "Content Based Image Retrieval Using SVM", August 2016 describes that feature extraction was seen as the binary classification problem and SVM (Support Vector Machine) was used for solving this problem in CBIR. In this SVM is used as the classifier which is performing the task of classifying the image and this process of classification is given to all the traits of the image which are extracted after the feature extraction process. It is mostly used for estimating the highest margin hyper planes within the feature space which is also a high dimensional feature space. Very few features are extracted. So feature extraction process is a disadvantage.

2.9) Sumiti Bansal, Er.Rishamjot Kaur “Content Based Image Retrieval Using C SVM Technique”, April 2015 describes a cluster based support vector machine technique for the retrieval of images from the database. The aim is to retrieve images efficiently in large databases. The system has two phases: preprocessing phase and image retrieval phase. The retrieval accuracy is remarkable but the retrieval time taken is a disadvantage.

CHAPTER 3

EXISTING SYSTEM

3.1 INTRODUCTION

Content-based image retrieval aims to retrieve images from a database which are similar to each other in terms of their visual contents. The image is converted from RGB to HSV and then discrete cubic partitioning is performed. Then hierarchical operators and SGLCM are defined in HSV space. Using 2D CS measurement model hierarchical HSV features and SGLCM texture features are extracted. These features are used to compute the similarity between the query image and the candidate images in the database.

Firstly the query image is processed using hierarchical operators. Then each resultant hierarchical mapping matrix is processed to compute a two dimensional compressive sensing measurement. Then texture features are extracted using SGLCM. Candidate images are retrieved from the database for which color and texture features are extracted for comparison with the query image. A normalized similarity score is found for every image and the images are ranked based on similarity. The image with the highest similarity score is ranked first. The images are retrieved according to the rank.

3.2 SYSTEM ARCHITECTURE

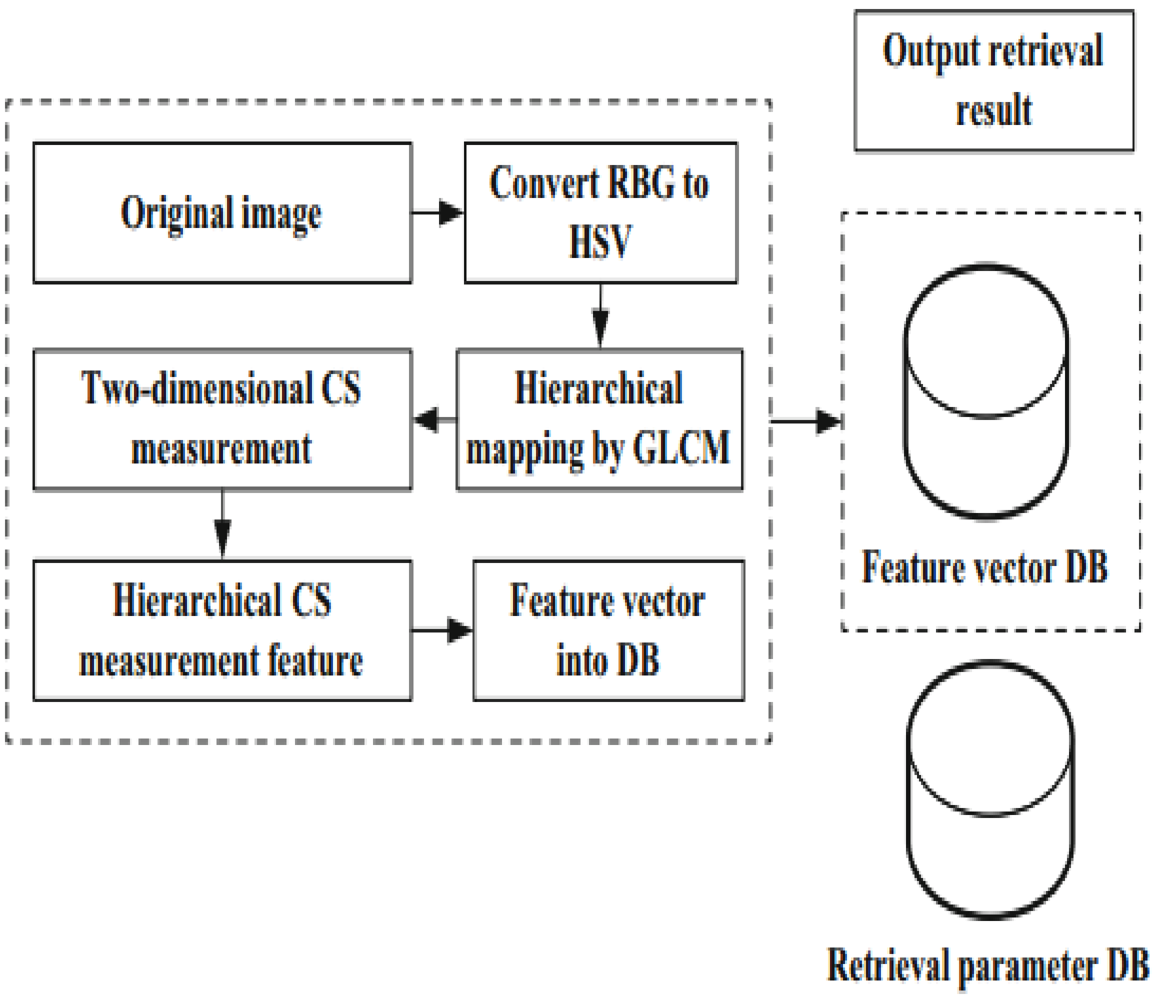


Figure 3.1: Exising system architecture

3.3 DATA FLOW DIAGRAM

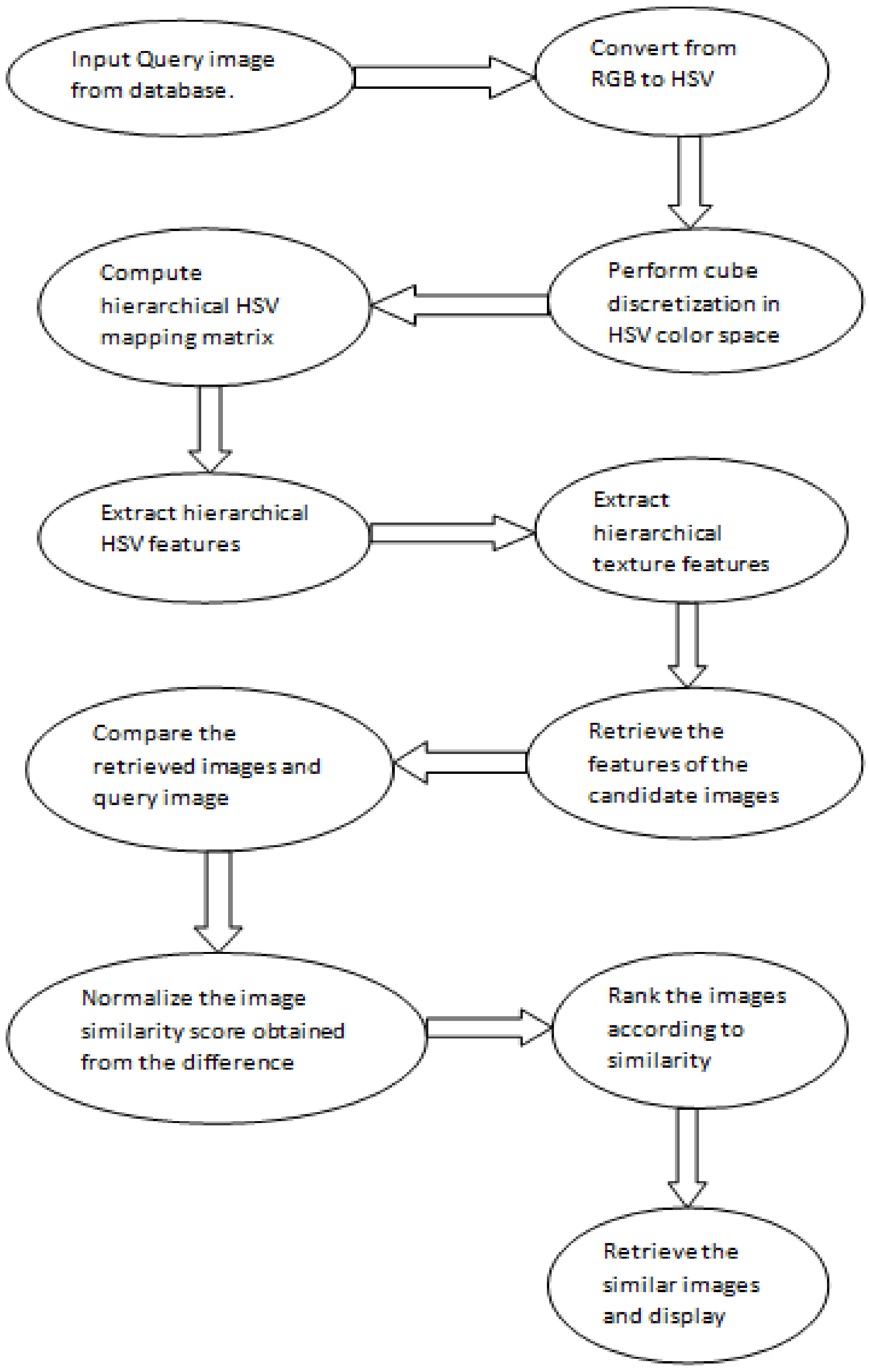

Input Query image from database.
Convert from RGB to HSV
Perform cube discretization in HSV color space
Compute hierarchical HSV mapping matrix
Extract hierarchical HSV features
Extract hierarchical texture features
Retrieve the features of the candidate images
Compare the retrieved images and query image
Normalize the image similarity score obtained from the difference
Rank the images according to similarity
Retrieve the similar images and display

Figure 3.2: Existing data flow diagram

3.4 MODULE DESCRIPTION

- ❖ Query image preprocessing
- ❖ Feature Extraction
- ❖ Similarity Based Ranking

3.4.1 QUERY IMAGE PREPROCESSING

For the query example image T, first convert the data from RGB format into the HSV color space. Let mx = max (R, G,B),mn = min (R, G, B) and md = mx - mn. The conversion of an image from RGB space to HSV space can now be defined as follows:

$$V = m_x$$

$$S = \begin{cases} m_d/m_x, & \text{if} \quad m_x! = 0 \\ 0, & \text{if} \quad m_x == 0 \end{cases}$$

$$H = \begin{cases} 0, & \text{if } (m_x == m_n) \\ 60(G-B)/m_d, & \text{if } (R == m_x) \\ 60(B-R)/m_d + 120, & \text{if } (G == m_x) \\ 60(R-G)/m_d + 240, & \text{if } (B == m_x) \end{cases}$$

Then discrete cubic partitioning is performed in HSV color space. This is the preprocessed image.

3.4.2 FEATURE EXTRACTION

Obtain the hierarchical HSV mapping matrix to extract the features of the image. The matrix can be obtained by row priority arrangement and by defining the hierarchical mapping operators in the cube.

$$HIER_l(i,j) = \begin{cases} 1, \ if(H(i,j), S(i,j), V(i,j))^T \in V_l \\ 0, \ \text{else} \end{cases}$$
$$(l = 1,2,\ldots,L; i,j = 1,2,\ldots,N)$$

The hierarchical HSV mapping matrix HIERl(X) reflects the distribution of the locations of image pixels whose color components are contained within the same cube, Vl of the HSV color space. Using a 2D CS model, the CS measurement vector Y_l can be calculated as:

$$Y_l = \Phi_1 \cdot \mathrm{HIER}_l(X) \cdot \Phi_2^T \in R^{M \times M}, \quad l = 1, 2, \ldots L.$$

Yl reflects the hierarchical features of the original image in HSV color space and, as such, it is called a hierarchical HSV feature. The SGLCM is calculated from the HSV color space. The texture features are extracted using :

$$PY_l = PY_{ij} = \Phi_1 \cdot P(\theta_i, d_j) \cdot \Phi_2^T \in R^{M \times M}$$

where $l = (i - 1) \times L_4 + j, i \in [1, L_4], j \in [1, L_5], l \in [1, L]$.

If $l > L_4 \times L_5$, then $PY_l = 0$.

Thus features for query and candidate images are extracted using the above methods.

3.4.3 SIMILARITY BASED RANKING

The differences between query and candidate images are quantified and normalized. The database images which are compared with the query image are ranked based on their similarity. The database image which is most similar is ranked 1st down to the image deemed to be least similar which is ranked last. The similar images are retrieved and displayed.

The results for query images chosen from every class is displayed below.

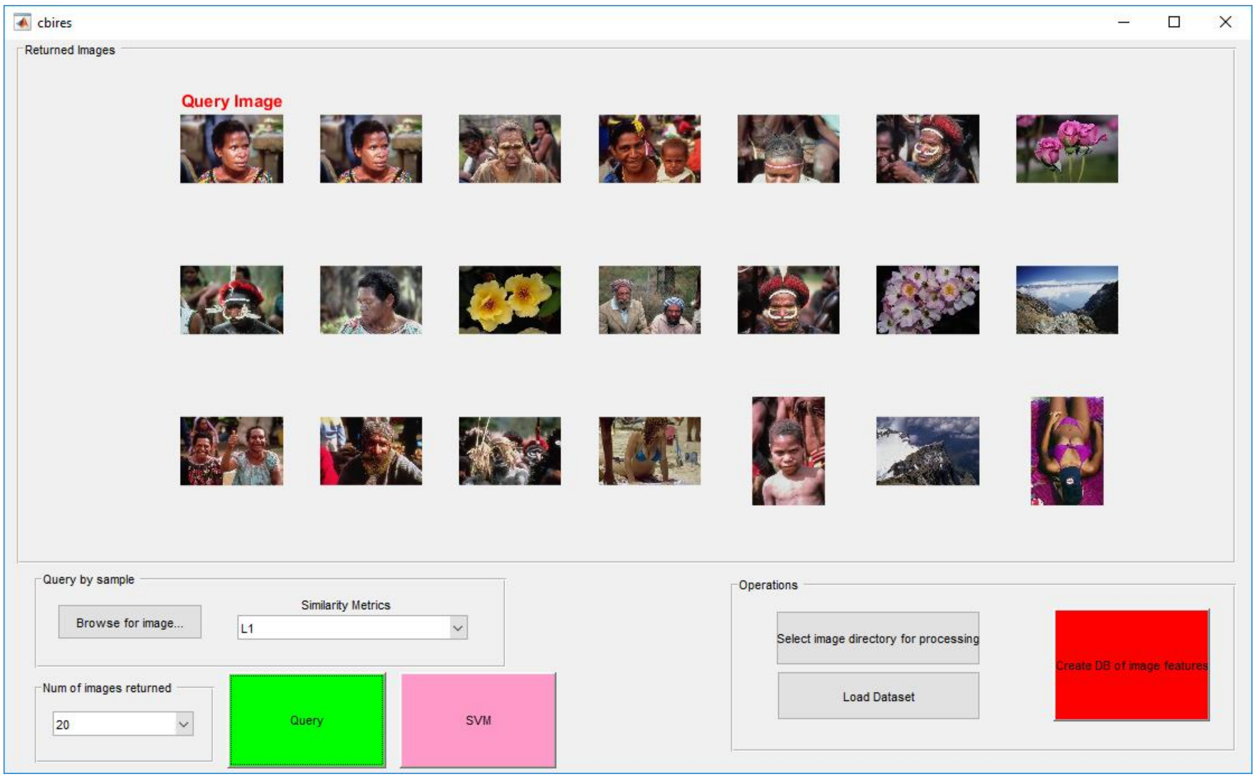


Figure 3.3: Retrieved images in existing system when query image is from the Africa class.

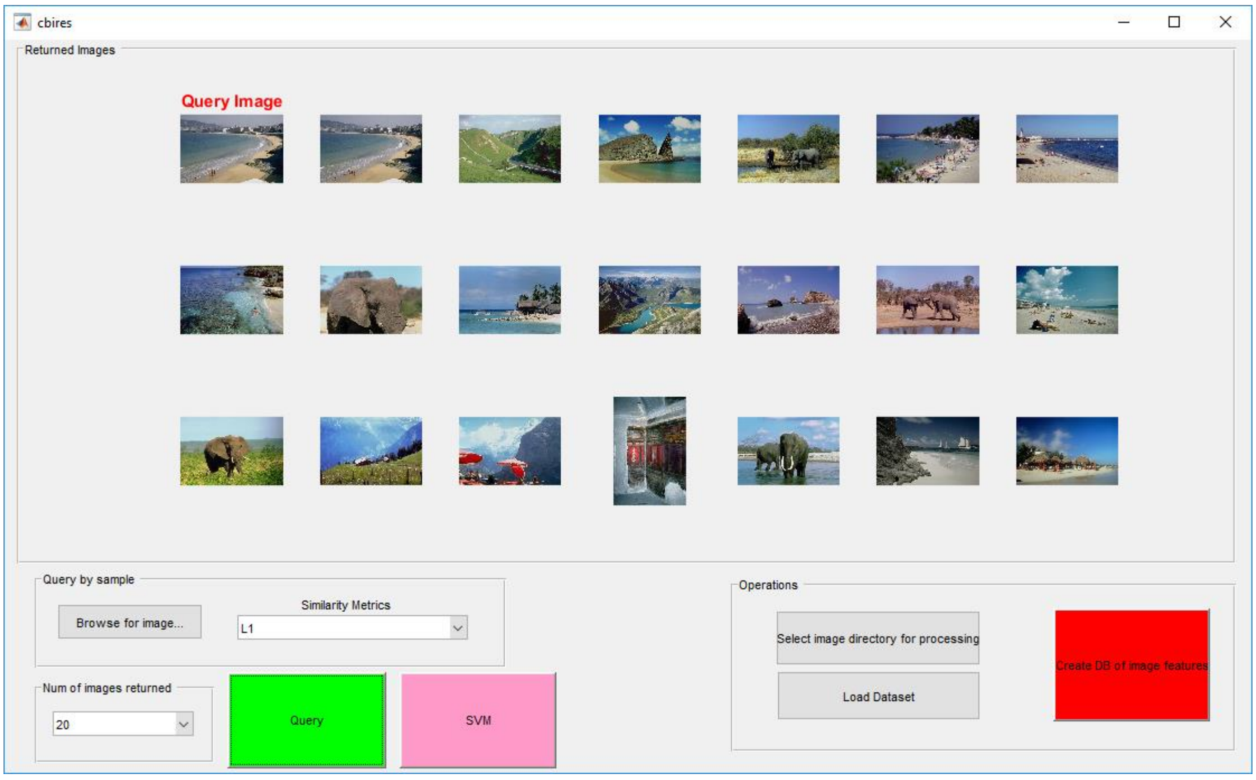
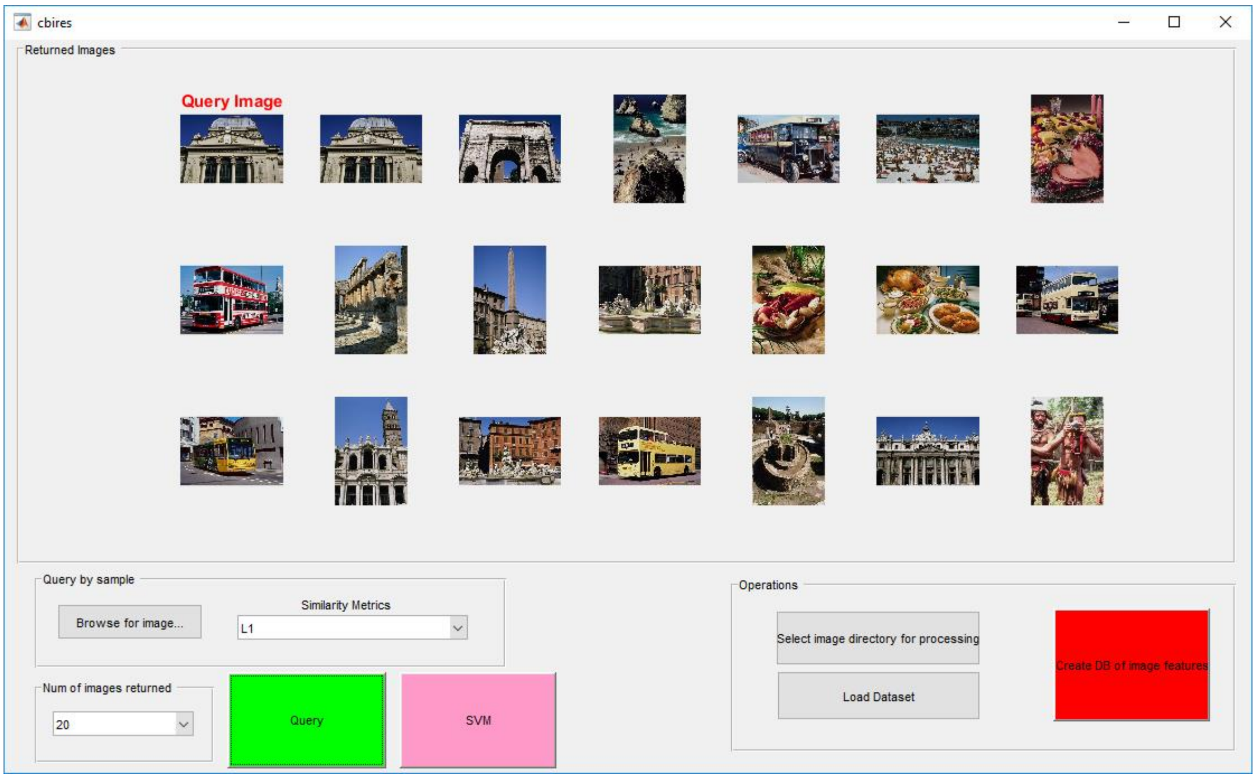


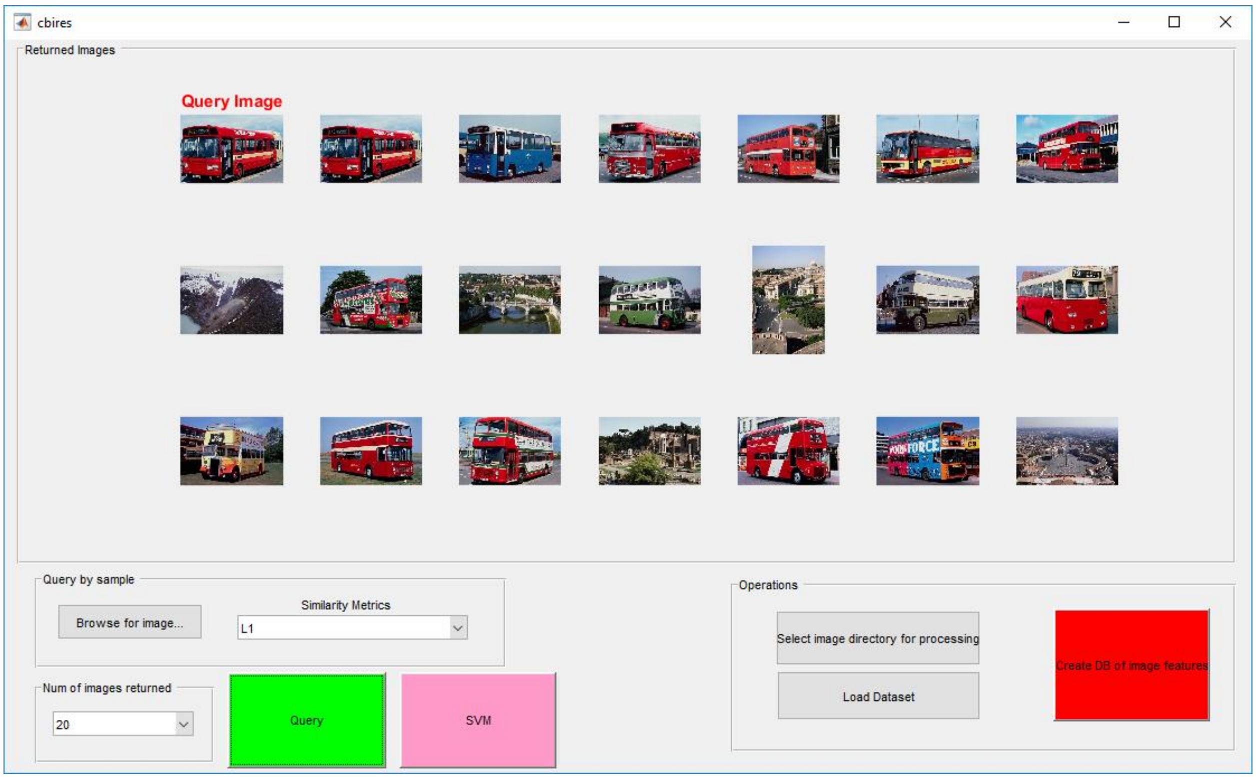
Figure 3.4: Retrieved images in existing system when query image is from the Beach class.



Similarity Metrics: L1

Num of images returned: 20

Figure 3.5: Retrieved images in existing system when query image is from the Monument class.



Similarity Metrics: L1

Num of images returned: 20

Figure 3.6: Retrieved images in existing system when query image is from the Bus class.

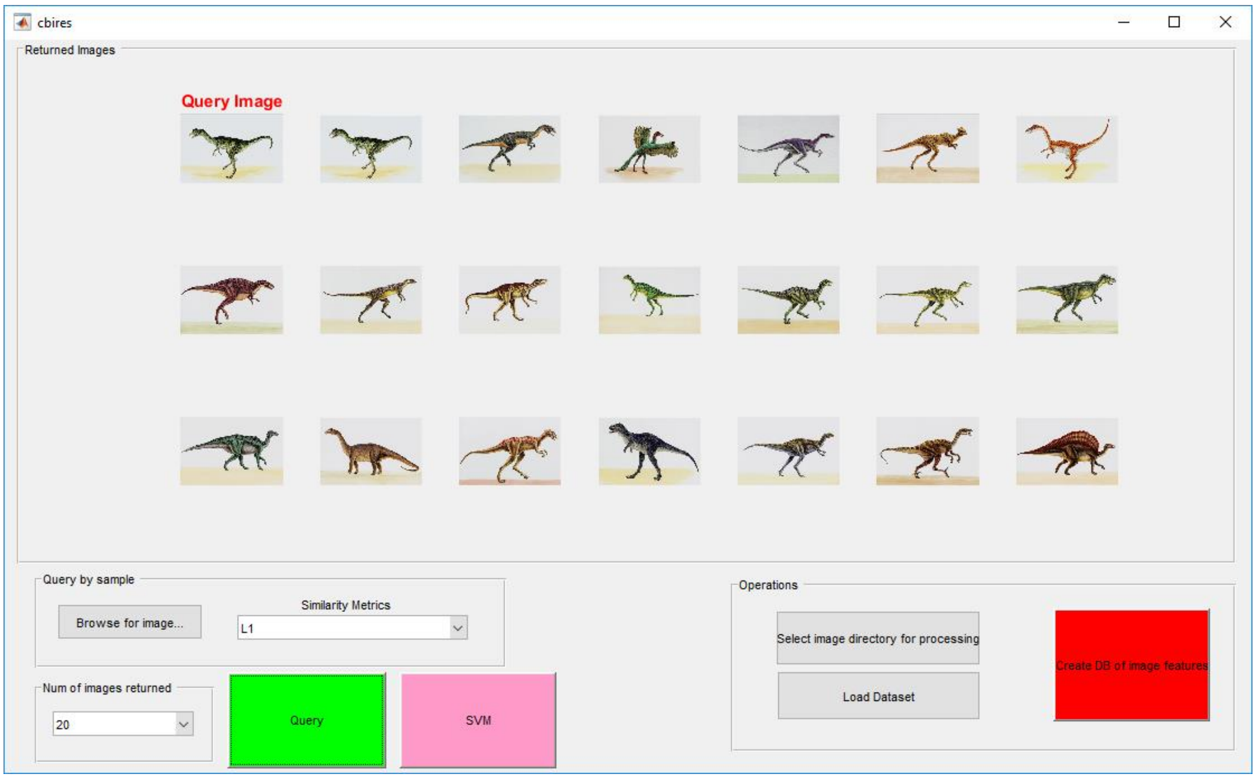


Figure 3.7: Retrieved images in existing system when query image is from the Dinosaur class.

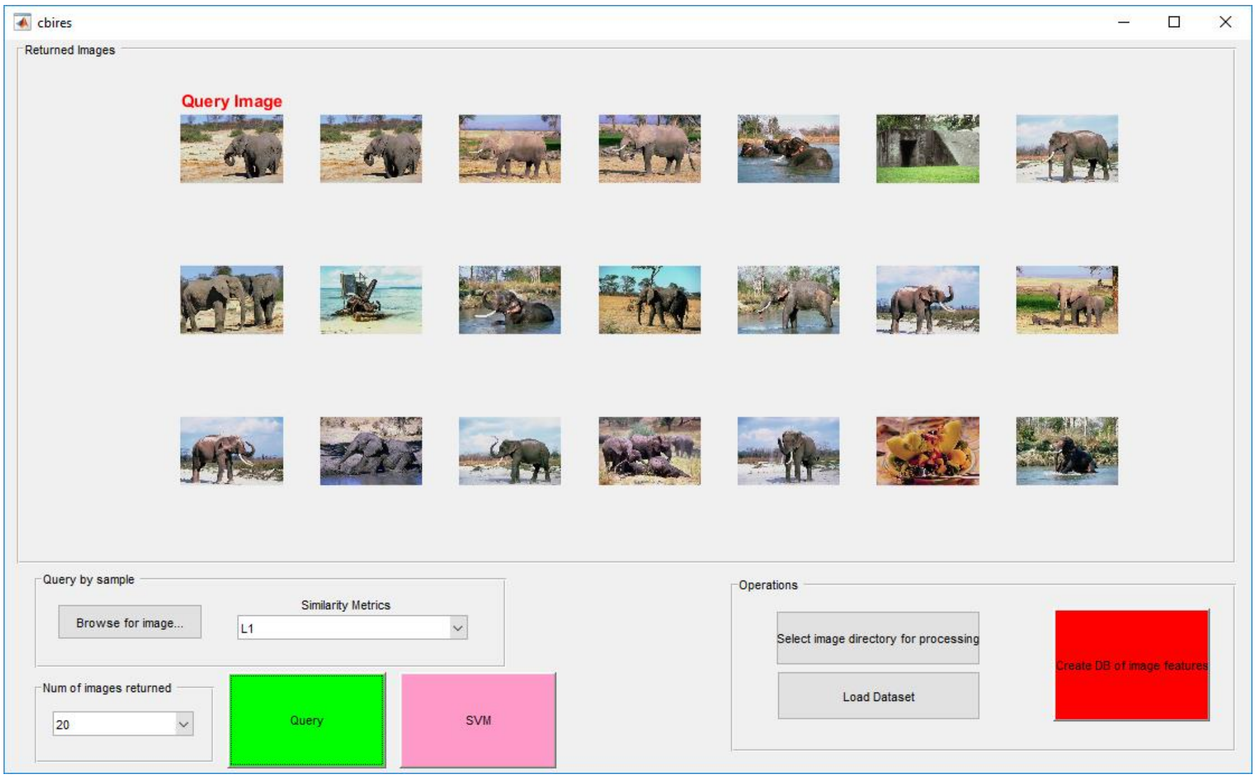


Figure 3.8: Retrieved images in existing system when query image is from the Elephant class.

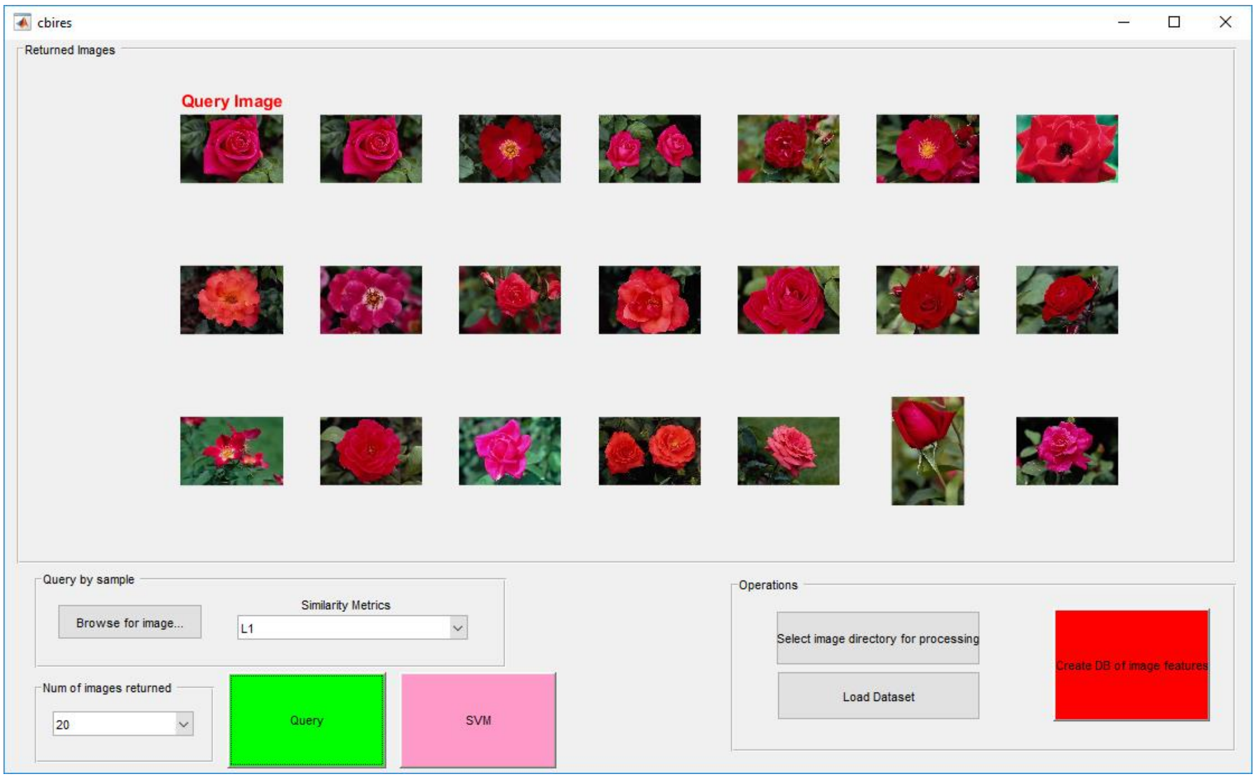


Figure 3.9: Retrieved images in existing system when query image is from the Flower class.

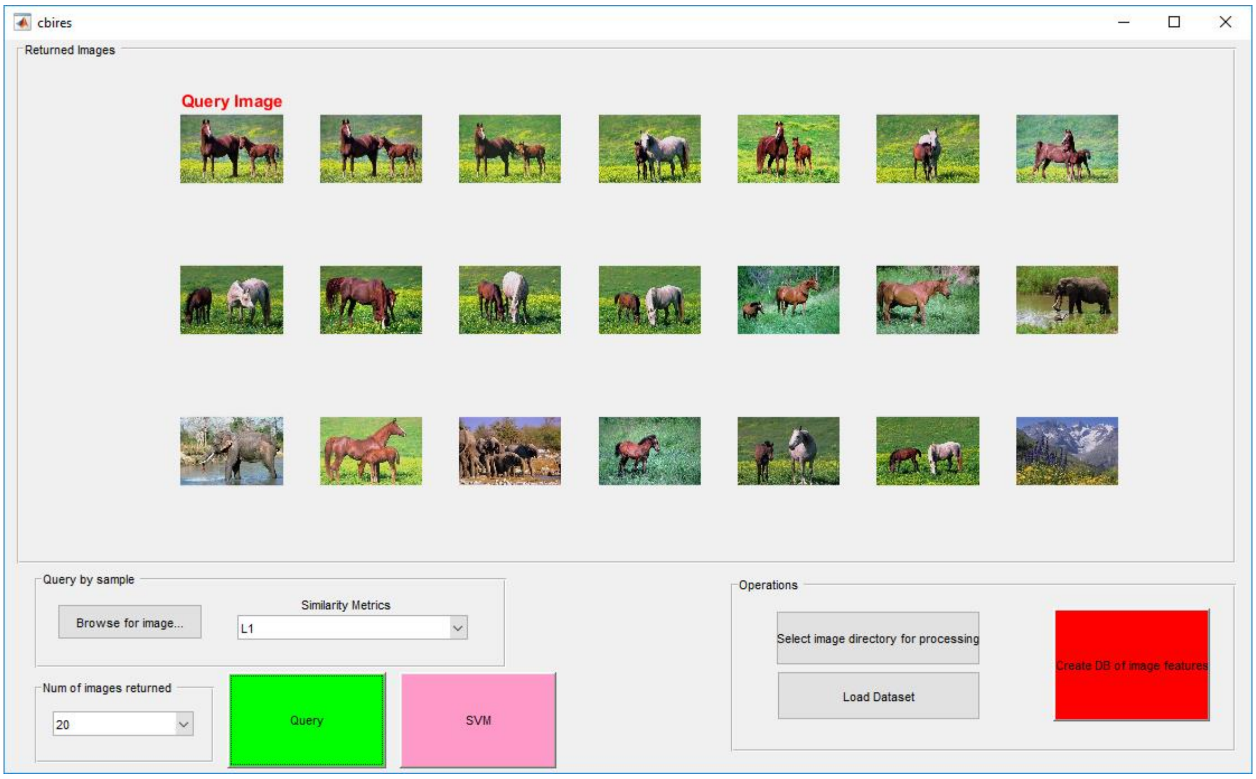


Figure 3.10: Retrieved images in existing system when query image is from the Horse class.

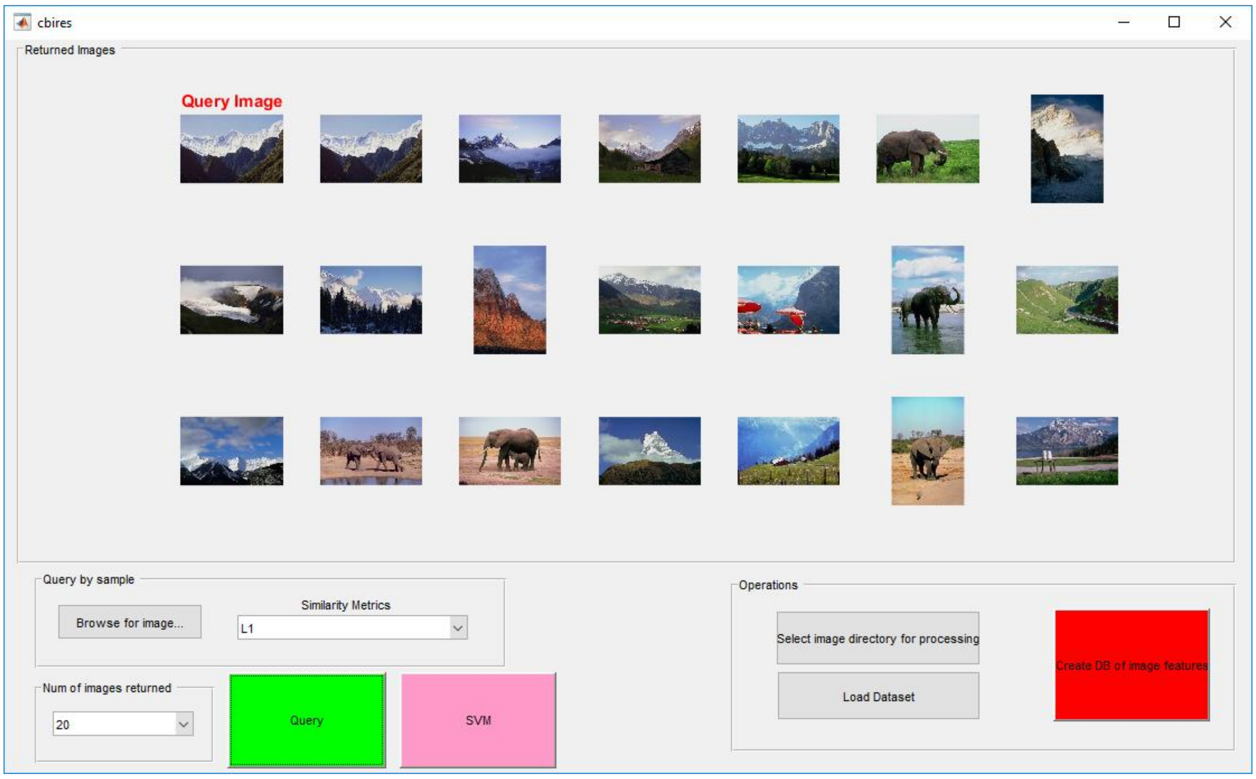


Figure 3.11: Retrieved images in existing system when query image is from the Mountain class.

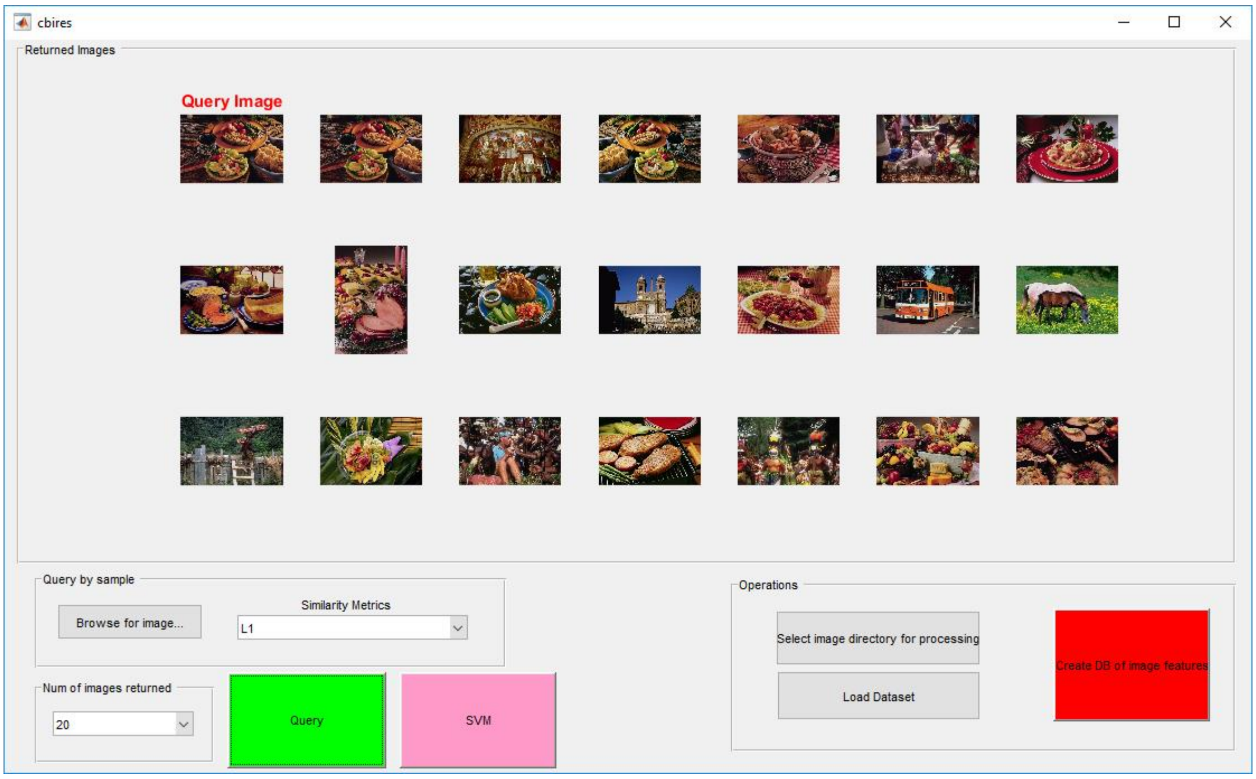


Figure 3.12: Retrieved images in existing system when query image is from the Food class.

3.5 DRAWBACKS OF EXISTING WORK

- Semantic gap exists between features extracted by computers and human perception.
- Inconsistent performance is observed.
- It requires empirical setting of parameters in the feature fusion step.
- Few retrieved images do not belong to the same class as that of the query image.

CHAPTER 4

PROPOSED SYSTEM

4.1 INTRODUCTION

The primary aim of this Content Based Image Retrieval (CBIR) is to have the exact result within the short period of time. Content-based image retrieval is based on the concept of using the depiction of characteristics which are retrieved from the images themselves. Maximum of the present CBIR systems works on querying-by-example, an approach in which a portion of the image or the whole image is chosen by the user in the form of query. The technique works by retrieving the feature of the image which is given as a query then going through the database for the image having similar features as the image in the query and extracts the appropriate images to the user which matches most appropriately to the query.

4.2 SYSTEM DESCRIPTION

The query image is selected from the database. HSV histogram, color auto-correlogram, color moments, Gabor wavelet and wavelet transform techniques are used to extract a total of 190 dimensions of the feature vectors. Thus features are extracted from the image and stored in database. Features are also extracted from every candidate image present in the database. Using SVM classification technique the query image is compared with the candidate images. The similar images are retrieved and displayed as result.

4.3 SYSTEM ARCHITECTURE

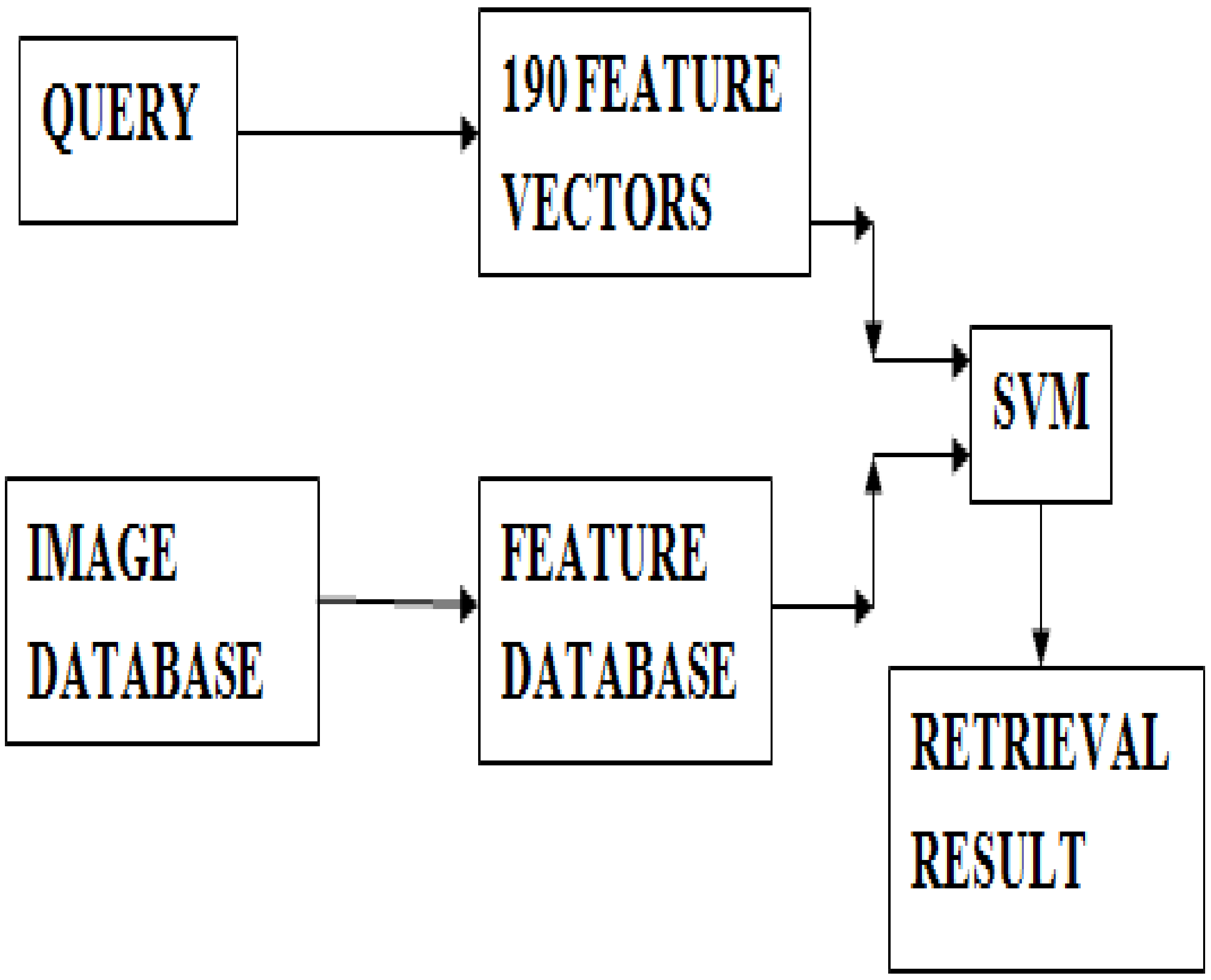
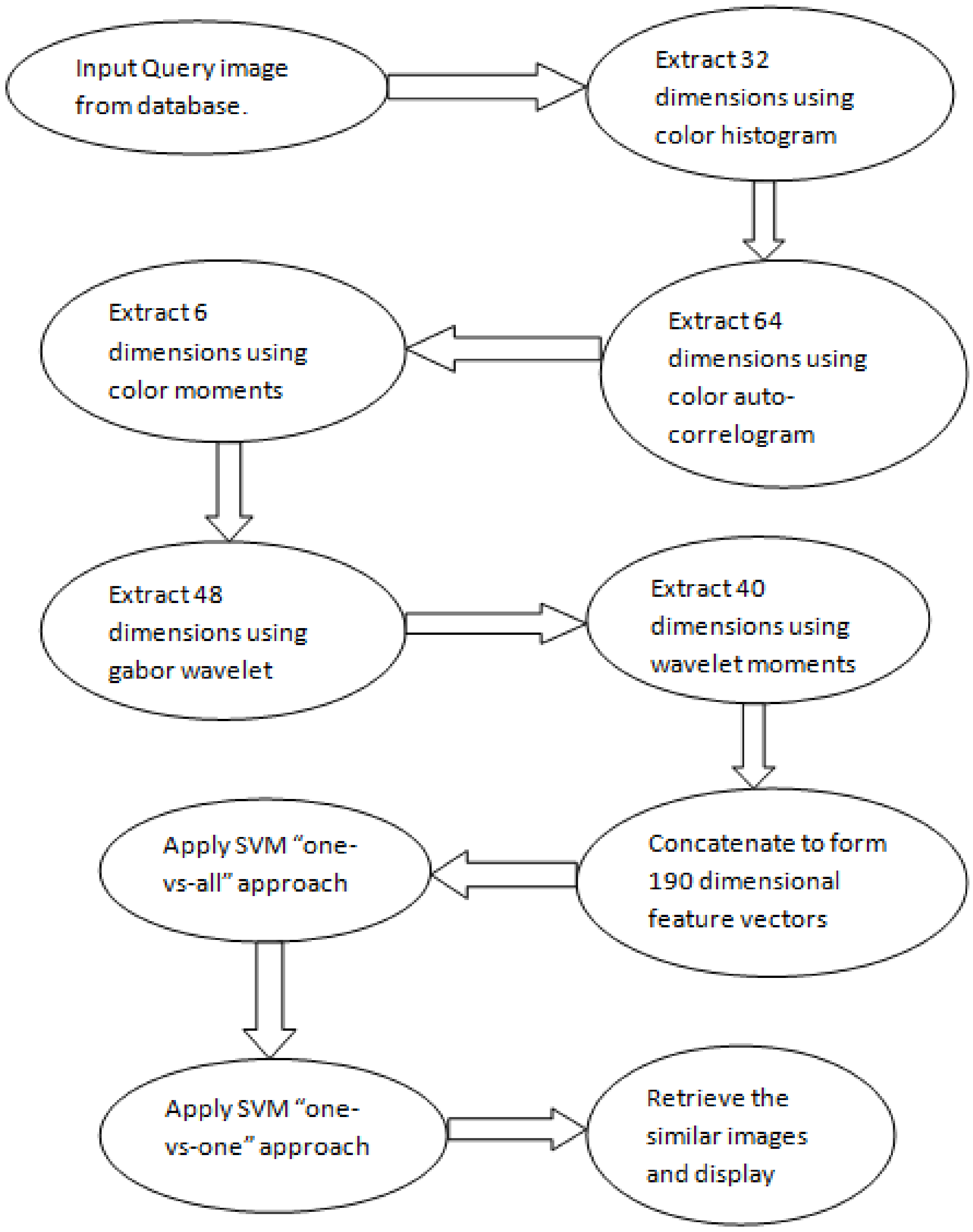


Figure 4.1: Proposed System Architecture

4.4 DATA FLOW DIAGRAM



Input Query image from database.
Extract 32 dimensions using color histogram
Extract 64 dimensions using color auto-correlogram
Extract 6 dimensions using color moments
Extract 48 dimensions using gabor wavelet
Extract 40 dimensions using wavelet moments
Concatenate to form 190 dimensional feature vectors
Apply SVM "one-vs-all" approach
Apply SVM "one-vs-one" approach
Retrieve the similar images and display

Figure 4.2: Proposed data flow diagram

4.5 MODULE DESCRIPTION

- ❖ Query image preprocessing
- ❖ Support Vector Machine

4.5.1 QUERY IMAGE PREPROCESSING

The raw pixel values are mapped into feature space. The below steps are applied to every image of our dataset to extract features of every image. The steps are following:

- ➢ Compute the color histogram for each image. In this case the HSV color space has been chosen and each H, S, V component is uniformly quantized into 8, 2 and 2 bins respectively. This produces a vector of 32 elements/values for each image.
- ➢ The next step is to compute the color auto-correlogram for each image, where the image is quantized into $4\times4\times4 = 64$ colors in the RGB space. This process produces a vector of 64 elements/values for each image.
- ➢ We extract the first two moments (i.e. mean and standard deviation) for each R,G,B color channel. This gives us a vector of 6 elements/values.
- ➢ Moving forward, we compute the mean and standard deviation of the Gabor wavelet coefficients, which produces a vector of 48 elements/values. This computation requires applying the Gabor wavelet filters for each image spanning across four scales: “0.05, 0.1, 0.2, 0.4” and six orientations: “$\theta_0 = 0$, $\theta n+1 = \theta n + 6/\pi$”.
- ➢ Apply the wavelet transform to each image with a 3-level decomposition. In this case the mean and standard deviation of the transform coefficients is utilized to form the feature vector of 40 elements/values for each image.

Finally, combine all the vectors 32 + 64 + 6 + 48 + 40 + 1. Each number indicates the dimensionality of the vectors from above steps that have been concatenated into the new vector.

4.5.2 SUPPORT VECTOR MACHINE

SVM technique is used for multiclass classification problem. It is used to classify the different sets of images present in our dataset. There are two approaches to solve this issue:

- One vs All approach

This involves training a single classifier per class as positive samples and all other samples as negatives. This strategy requires the base classifiers to produce a real-valued confidence score for its decision.

- One vs One approach

This trains n!/(n−k)!k! binary classifiers for a k-way multiclass problem. Each receives the samples of a pair of classes from the original training set, and must learn to distinguish these two classes. A voting scheme is applied at prediction time. All n!/(n−l)!k! classifiers are applied to an unseen sample. The class that gets the highest number of “+1” predictions gets predicted by the combined classifier. Thus by this method the candidate images are retrieved which are similar to the query image.

MATLAB R2016a is used to implement our approach for content based image retrieval. The following are the steps to be followed for implementation: An image directory is created for processing the image and a database is created which consists of the features of the image. Load the dataset. Browse the query image from the dataset and select one. Apply the above mentioned techniques and retrieve the images which are similar to the query image. The results for query images from every class is displayed below.

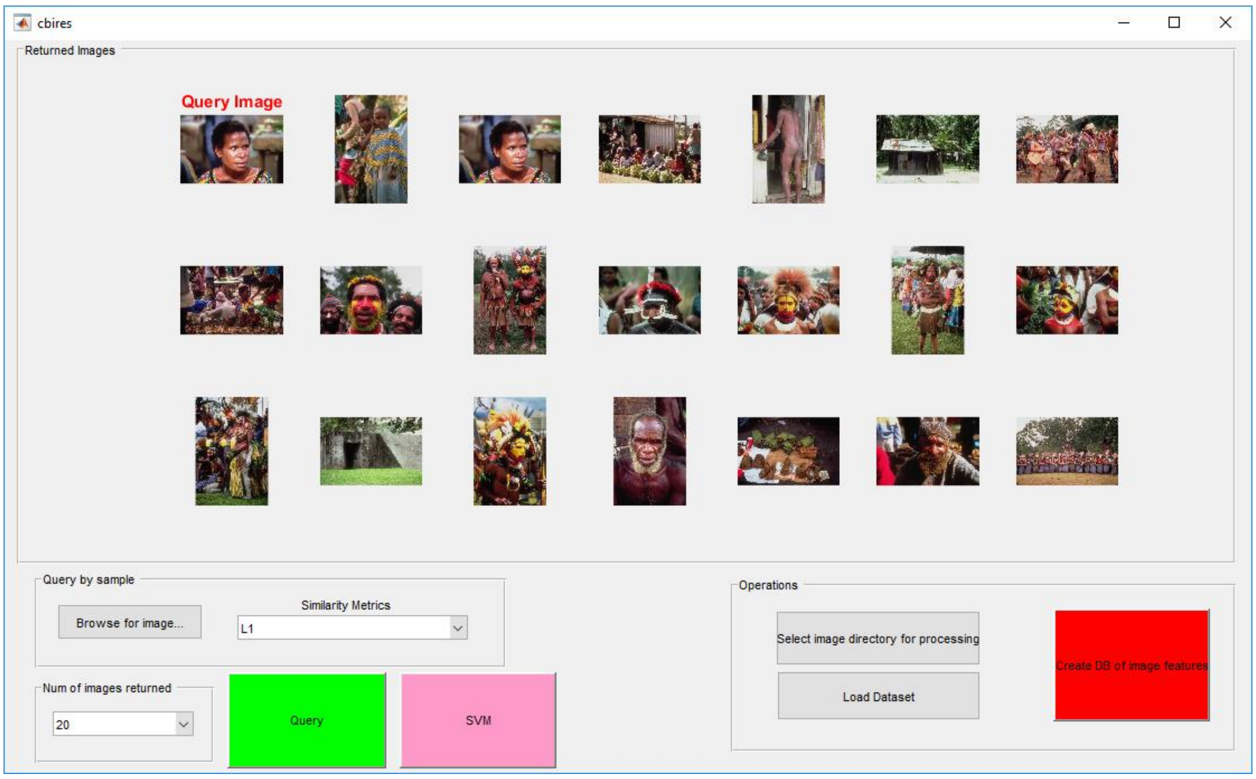


Figure 4.3: Retrieved images of proposed system when query image is from the Africa class.

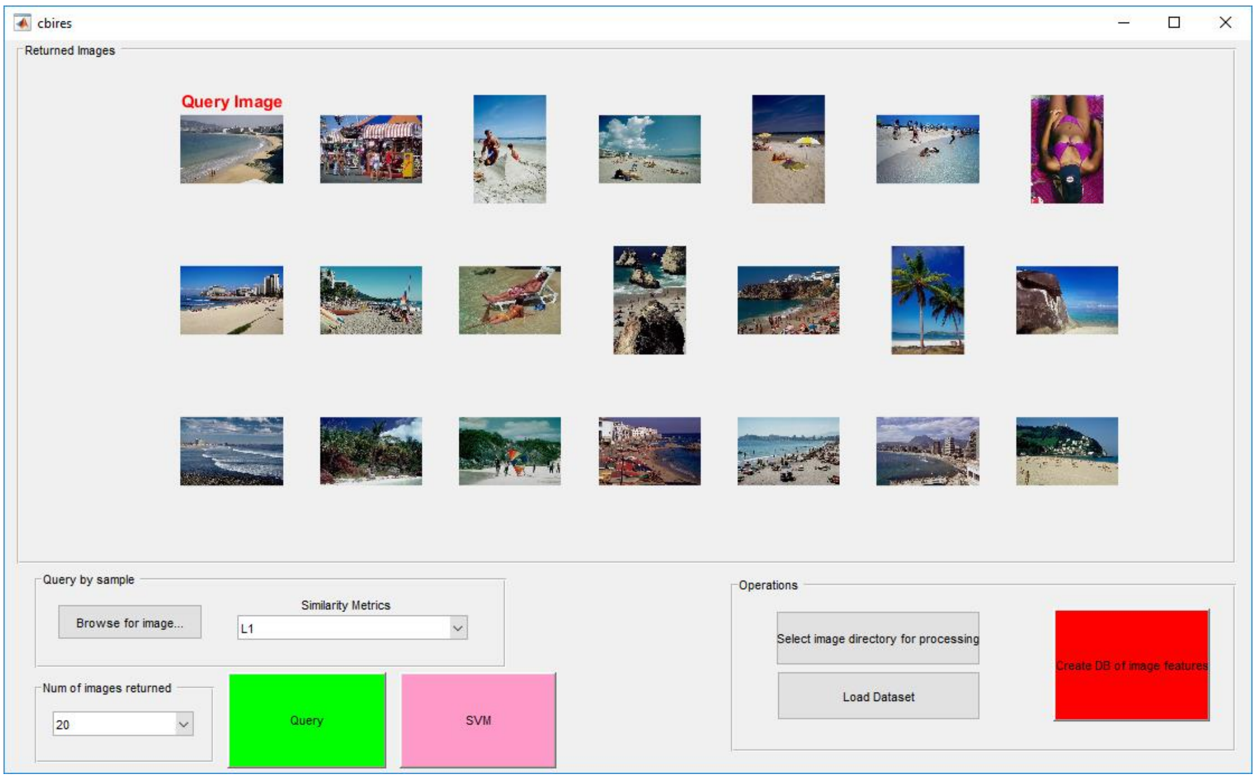


Figure 4.4: Retrieved images of proposed system when query image is from the Beach class.

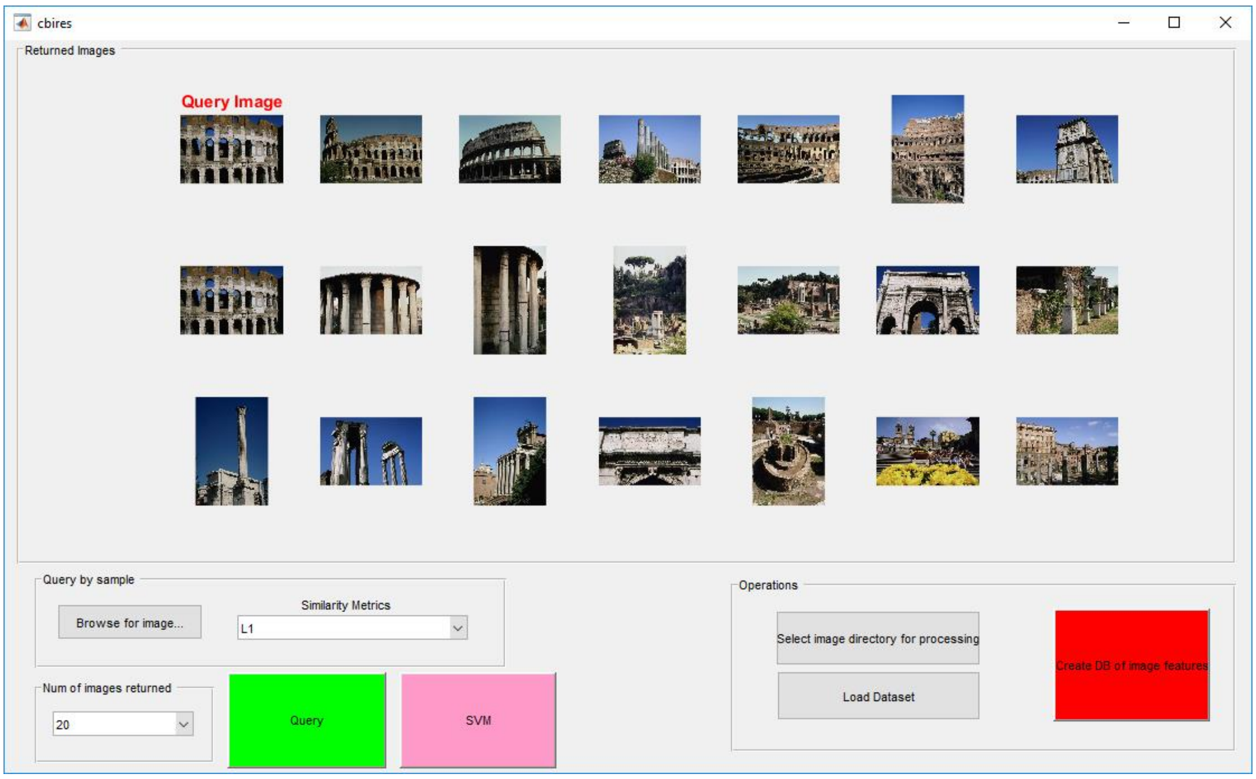


Figure 4.5: Retrieved images of proposed system when query image is from the Monument class.

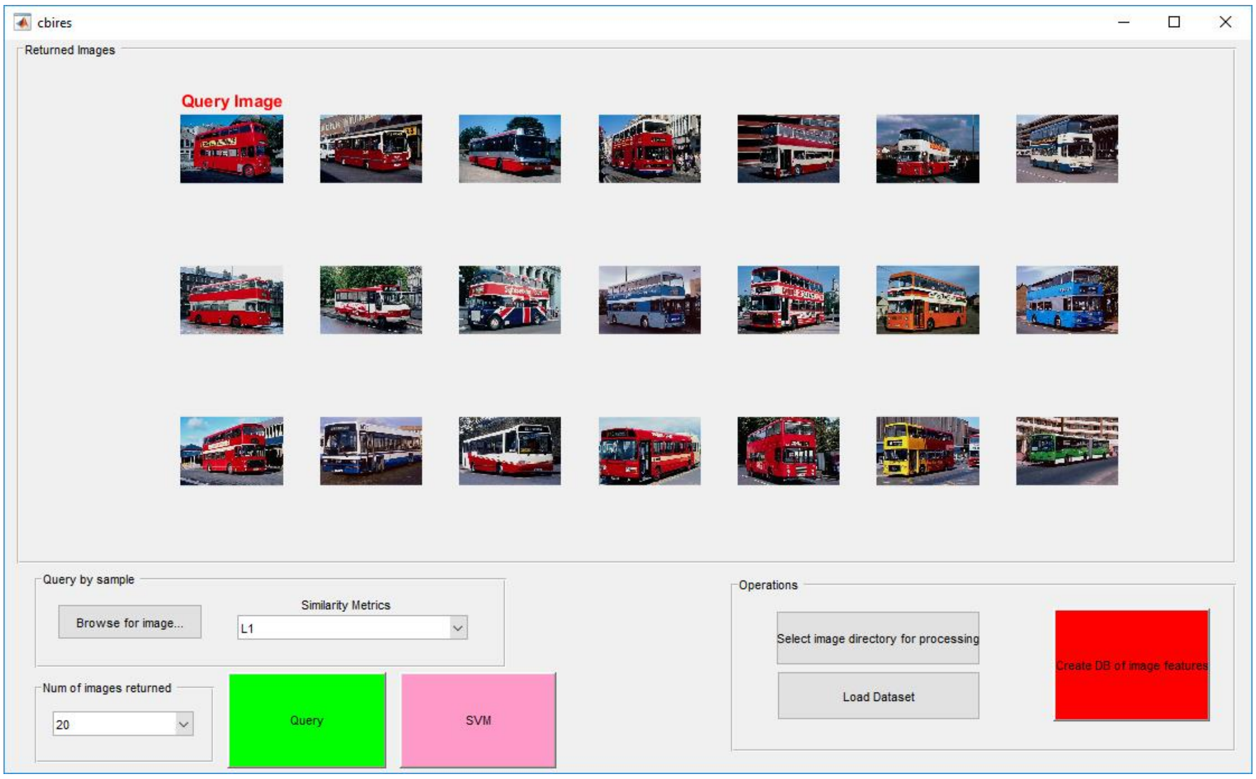


Figure 4.6: Retrieved images of proposed system when query image is from the Bus class.

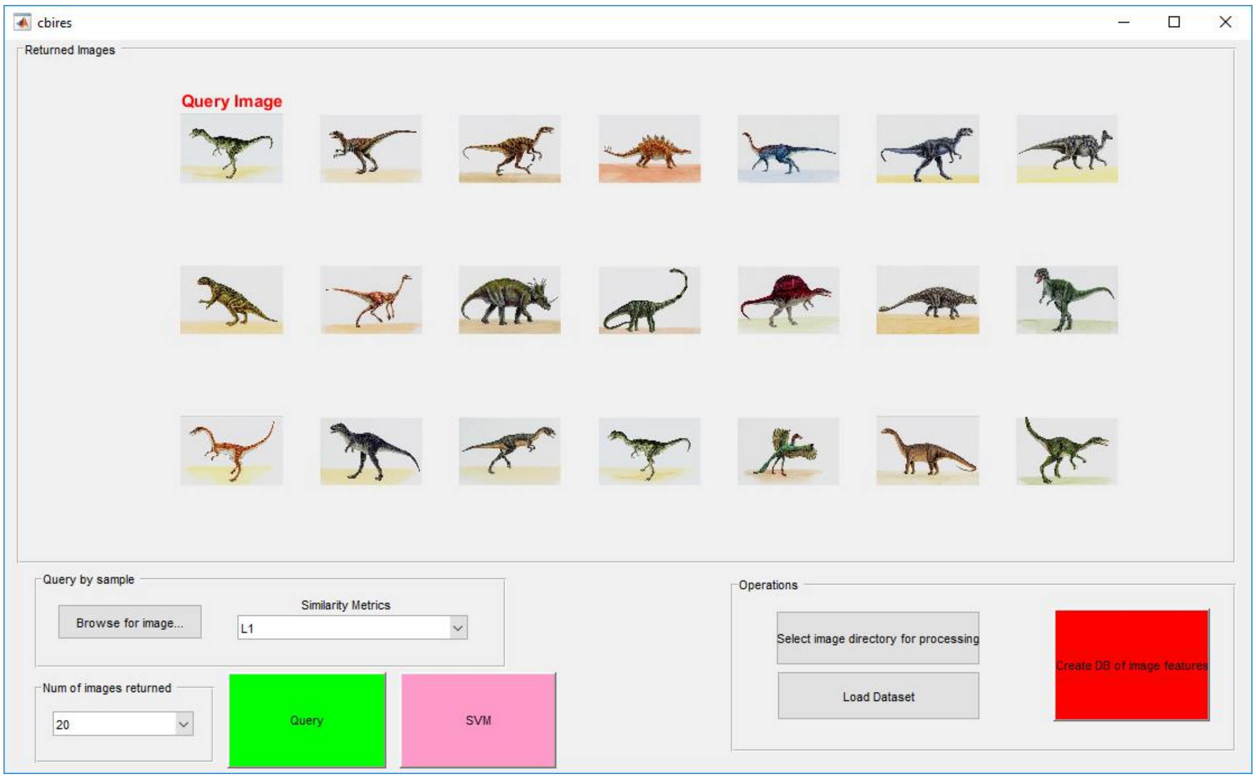


Figure 4.7: Retrieved images of proposed system when query image is from the Dinosaur class.

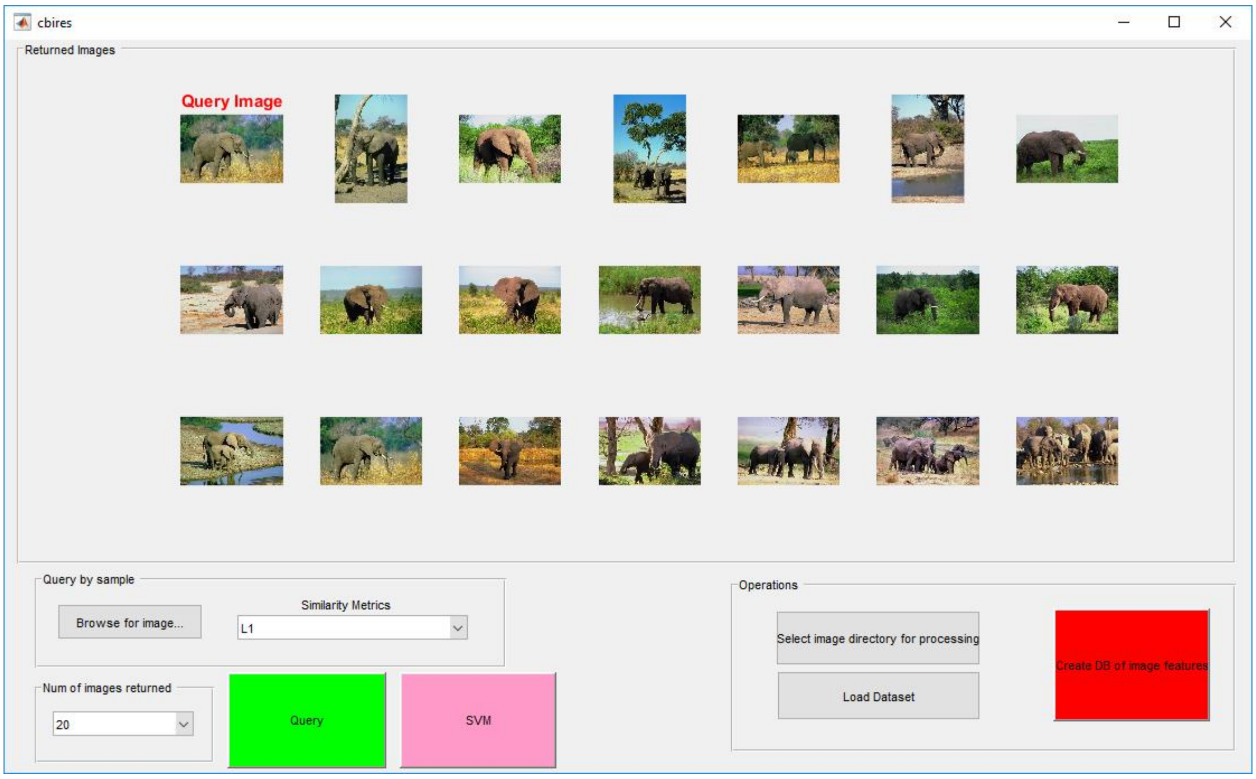


Figure 4.8: Retrieved images of proposed system when query image is from the Elephant class.

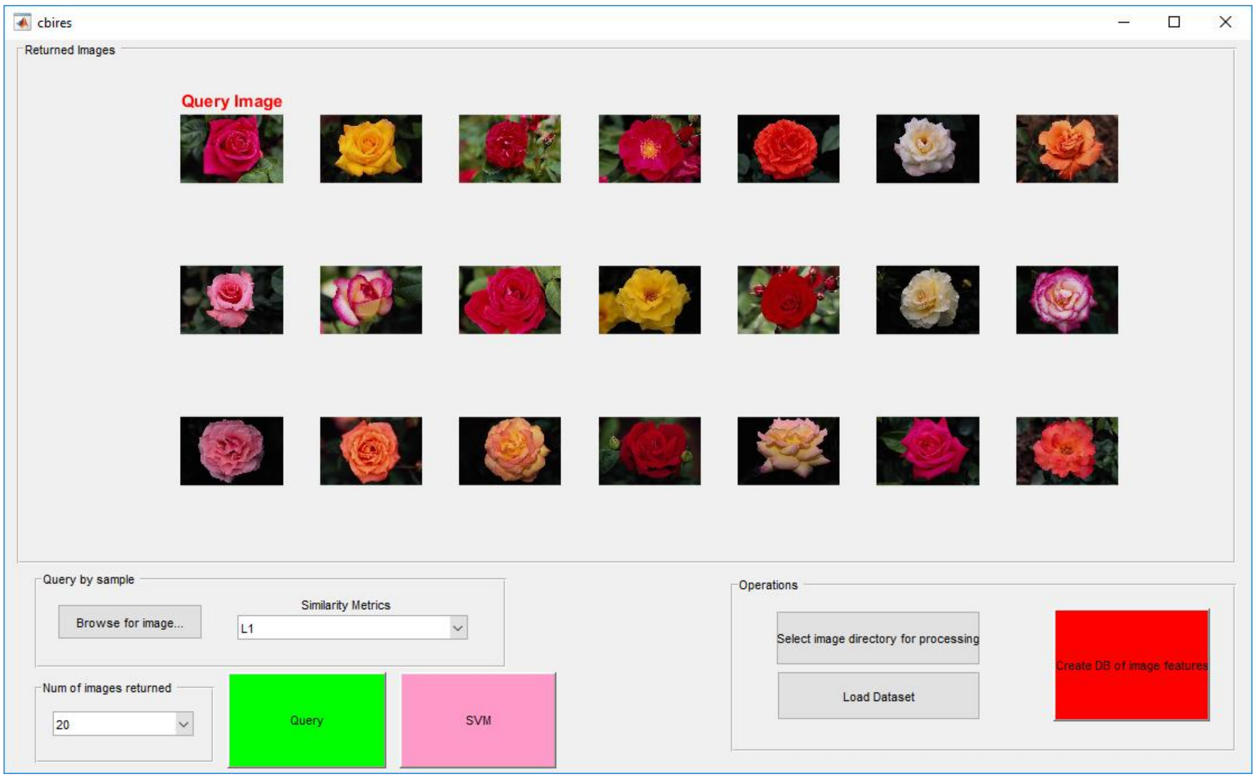


Figure 4.9: Retrieved images of proposed system when query image is from the Flower class.

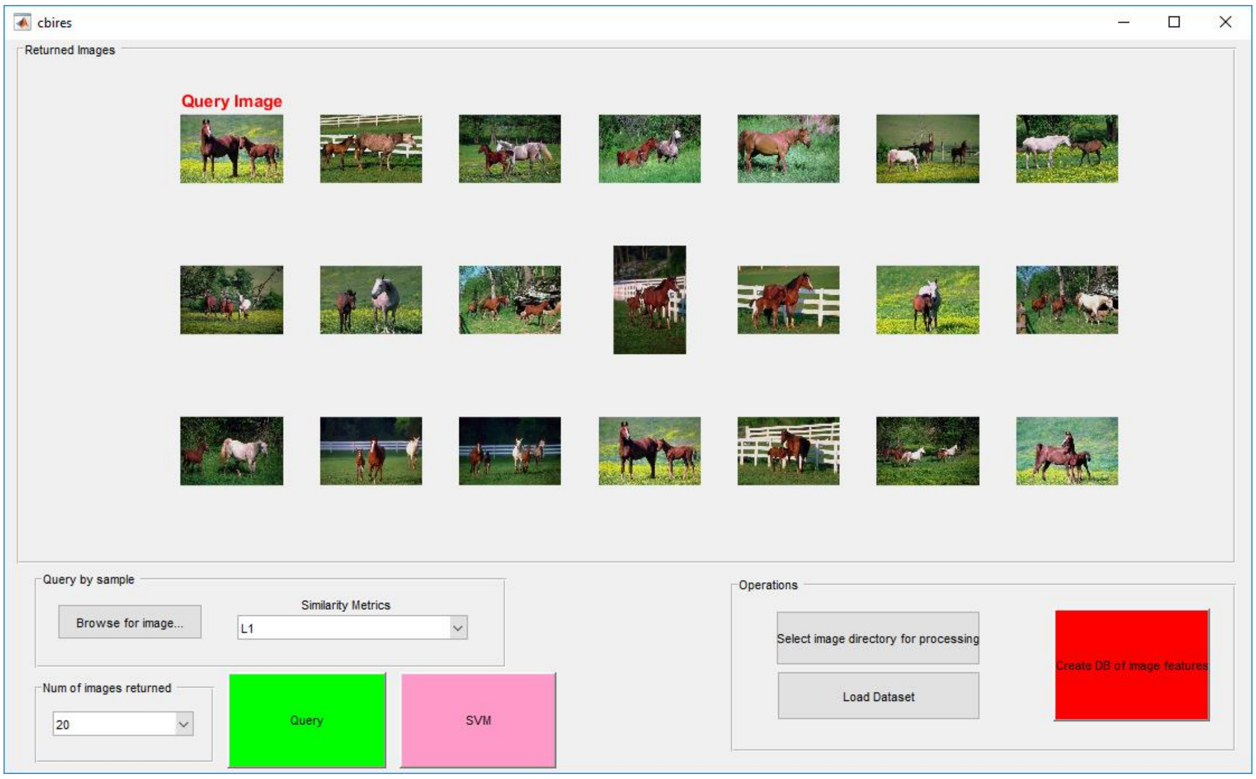


Figure 4.10: Retrieved images of proposed system when query image is from the Horse class.

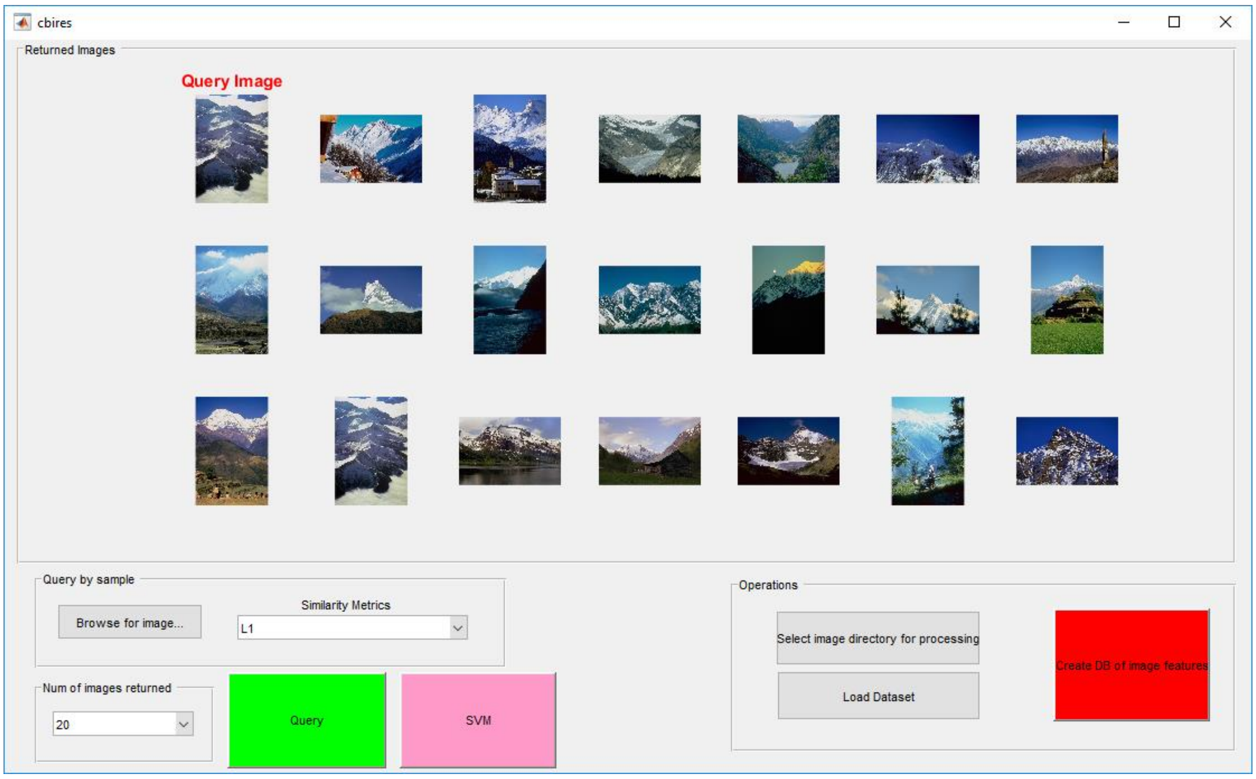


Figure 4.11: Retrieved images of proposed system when query image is from the Mountains class.

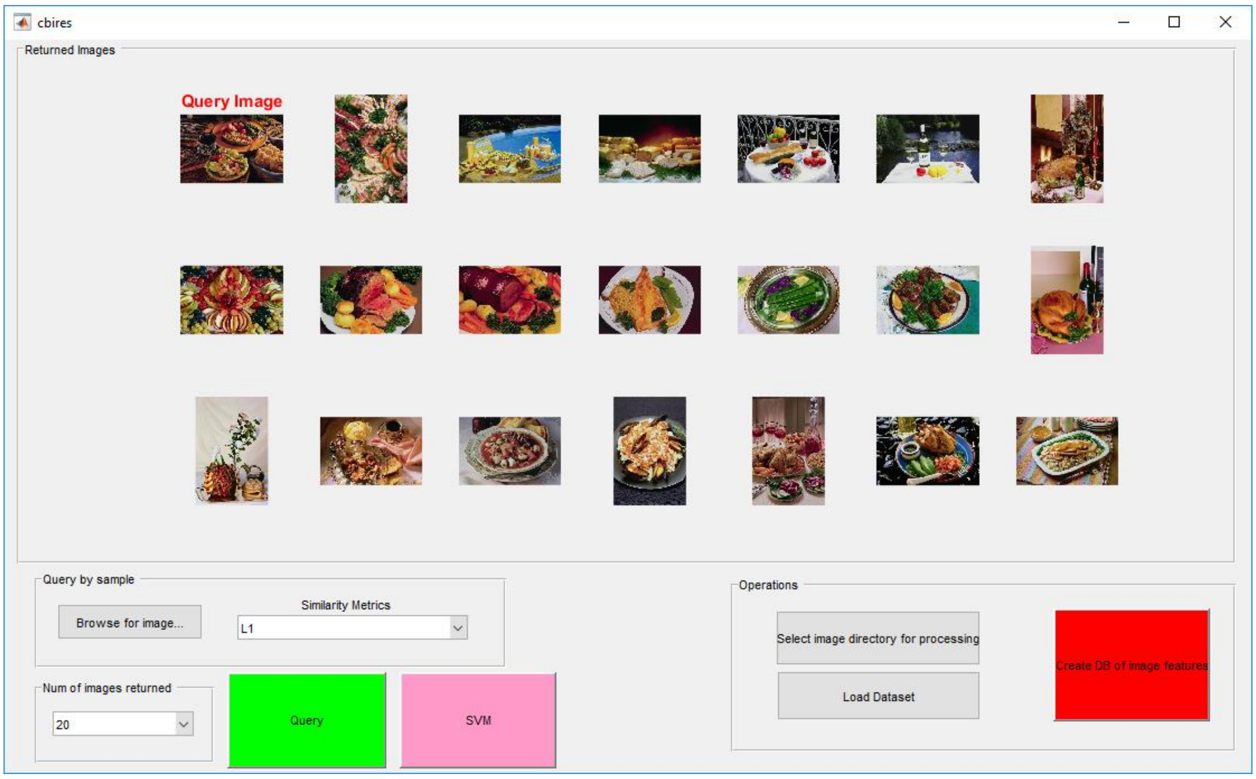


Figure 4.12: Retrieved images of proposed system when query image is from the Food class.

Evaluation of retrieval performance is a crucial problem in Content-Based Image Retrieval. Many different methods for measuring the performance of a system have been created and used by researchers. We have used the most common evaluation methods namely, Precision, Recall and F-Measure usually presented as a Precision, Recall and F-measure graph. Precision and recall alone contain insufficient information. We can always make recall value 1 just by retrieving all images. In a similar way precision value can be kept in a higher value by retrieving only few images or precision and recall should either be used together or the number of images retrieved should be specified. With this, the following formulae are used for finding Precision, Recall and F-measure values.

- **Precision:**

The precision is defined as, the precision parameter is used to measure the number of relevant images retrieved obtained by CBIR system, over the total number of images requested. The Precision is represented by P. The following formulae describe how precision can be calculated:

P = no. of relevant images retrieved / number of images requested

- **Recall:**

The recall is defined as; the recall parameter measures the no. of correct images obtained by CBIR system over the total no. of images requested. Thus, recall can be calculated as,

R= no. of images retrieved / number of images requested

- **F-measure:**

The F-measure is used to represents the harmonic mean of precision and recall i.e.

F=2RP/R+P

The F-measure (also F-score) is a score to measure the accuracy of the system. It considers both the precision P and the recall R of the system to calculate average score.

CHAPTER 5

EXPERIMENT ANALYSIS AND RESULT

5.1 DATABASE

The dataset being used is called as Wang database. It consists of 1000 images from 10 different categories where 100 images correspond to each category. The 10 categories or classes are: Africa, Beach, Monument, Bus, Dinosaur, Elephant, Flower, Horse, Mountain and Food.

$$\mathcal{D}_{10-\text{classes}} = \begin{cases} \text{Africa} \\ \text{Monum.} \\ \text{Animals} \\ \text{People} \\ \quad\vdots \end{cases} \leftarrow \left\{ \begin{array}{|c|c|c|c|c|} \hline \text{Africa} & \text{Monuments} & \text{Animals} & \text{People} & \cdots \\ \hline image_1 & image_1 & image_1 & image_1 & \vdots \\ image_2 & image_2 & image_2 & image_2 & \\ image_3 & image_3 & image_3 & image_3 & \\ \vdots & \vdots & \vdots & \vdots & \\ image_{100} & image_{100} & image_{100} & image_{100} & \\ \hline \end{array} \right.$$

Figure 5.1: Depicts the classes present in dataset with number of images.

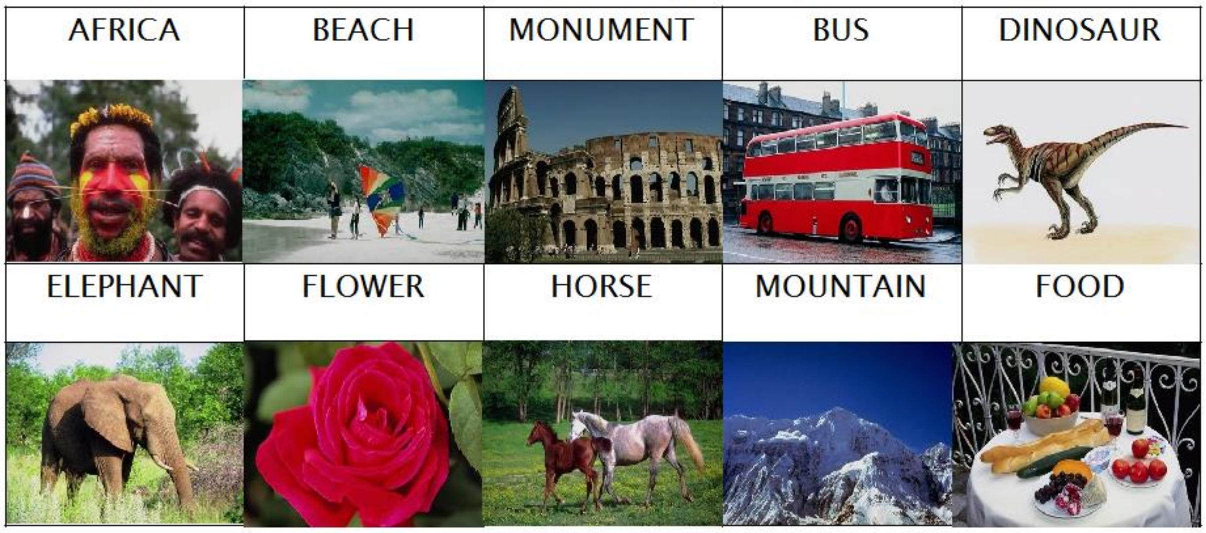


Figure 5.2: Sample images from every dataset.

5.2 PERFORMANCE EVALUATION FOR EXISTING SYSTEM

The table shows the Precision and F-score results for the results obtained for the query image from every class using similarity based ranking.

CLASSES	PRECISION (%)	F-SCORE (%)
AFRICA	65	78.8
BEACH	70	82.35
MONUMENT	45	62
BUS	75	85.71
DINOSAUR	100	100
ELEPHANT	85	91.89
FLOWER	100	100
HORSE	80	88.8
MOUNTAIN	75	85.71
FOOD	75	85.71

Table 5.1: Comparison of precision and F-score for query image from every class of the dataset using existing approach.

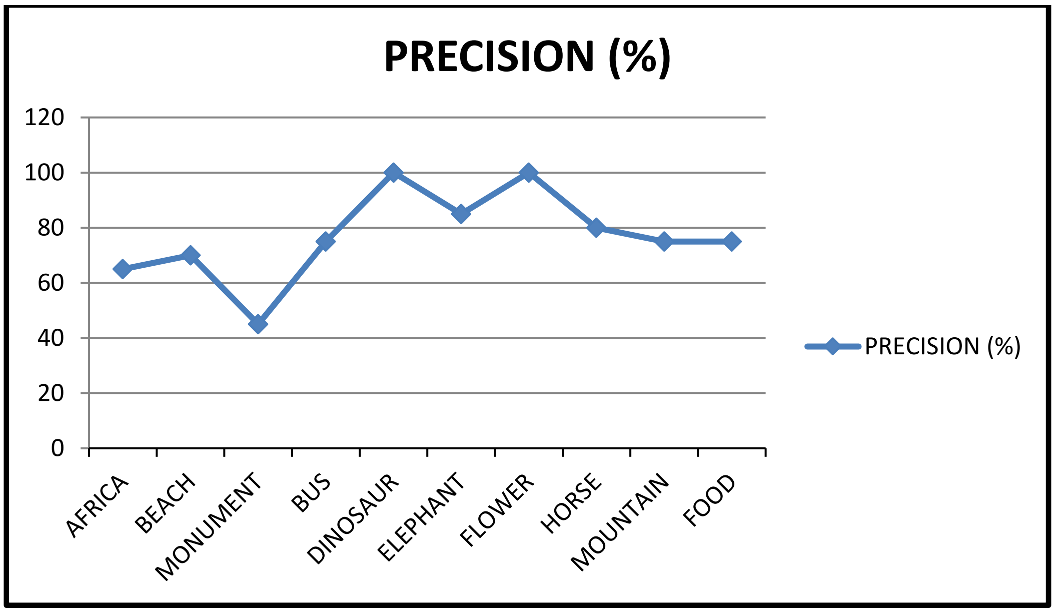


Figure 5.3: Comparison of datasets for precision using existing approach.

The values of precision are graphed across y axis for the classes present in the dataset along x axis. The precision score for few query images have gone very low and the performance in inconsistent when similarity based ranking is used for image retrieval.

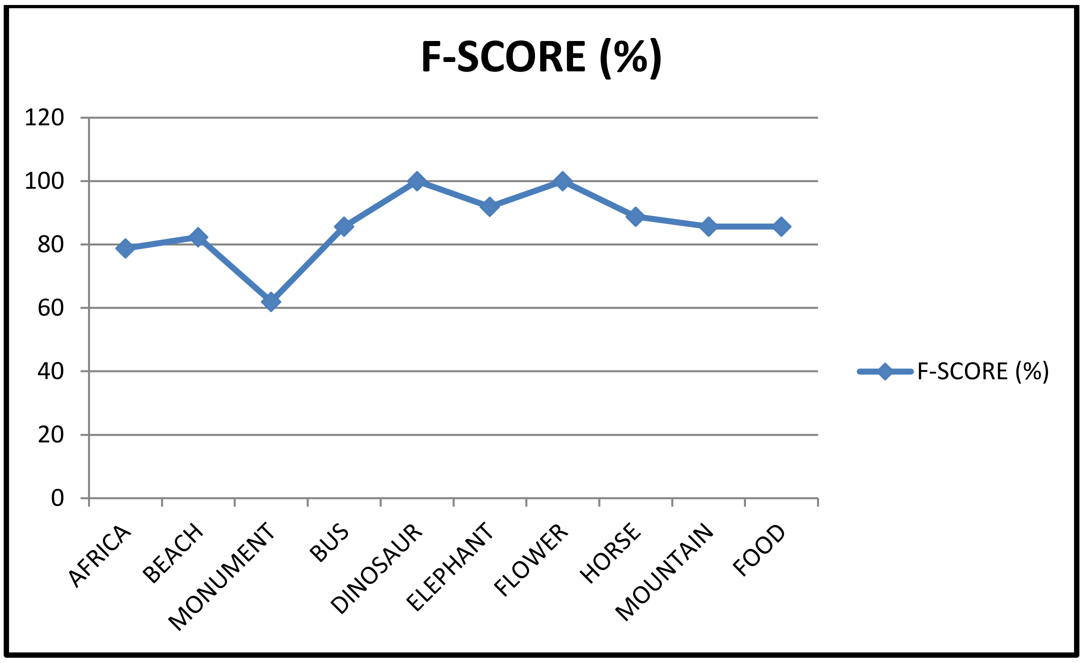


Figure 5.4: Comparison of datasets for F-score using existing approach.

The values of F-score are graphed across y axis for the classes present in the dataset along x axis.

5.2 PERFORMANCE EVALUATION FOR PROPOSED SYSTEM

The table shows the Precision and F-score results for the results obtained for the query image from every class using Support Vector Machine.

CLASSES	PRECISION (%)	F-SCORE (%)
AFRICA	95	97.43
BEACH	100	100
MONUMENT	100	100
BUS	100	100
DINOSAUR	100	100
ELEPHANT	100	100
FLOWER	100	100
HORSE	100	100
MOUNTAIN	100	100
FOOD	100	100

Table 5.2: Comparison of precision and F-score for every class of dataset using SVM.

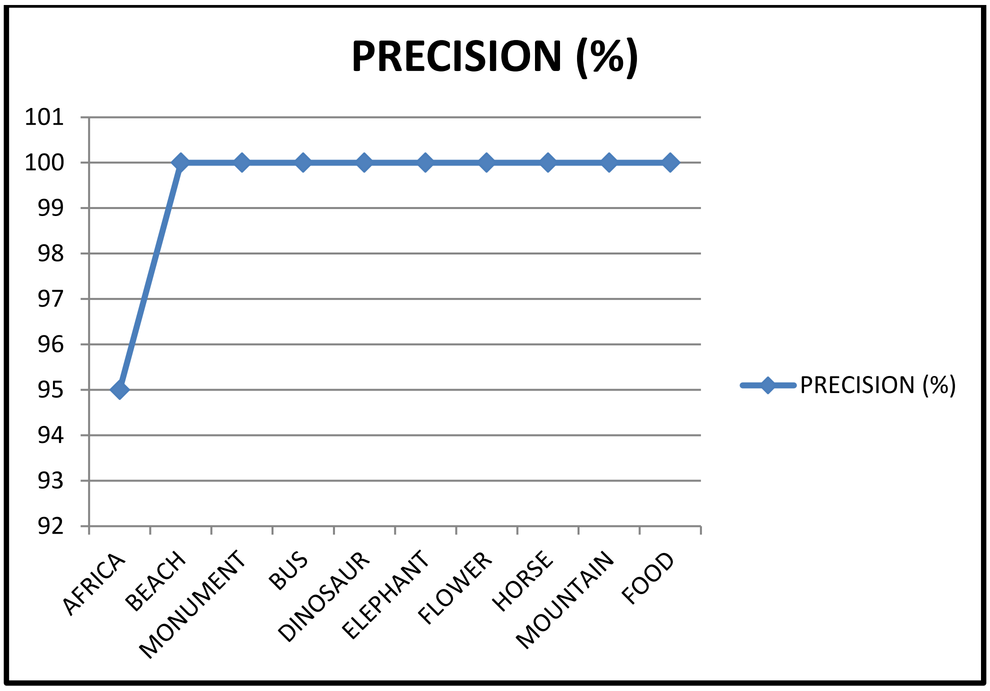


Figure 5.5: Comparison of datasets for precision using proposed method.

The values of precision are graphed across y axis for the classes present in the dataset along x axis. Using SVM the performance obtained is consistent. The variation in precision value is minimum.

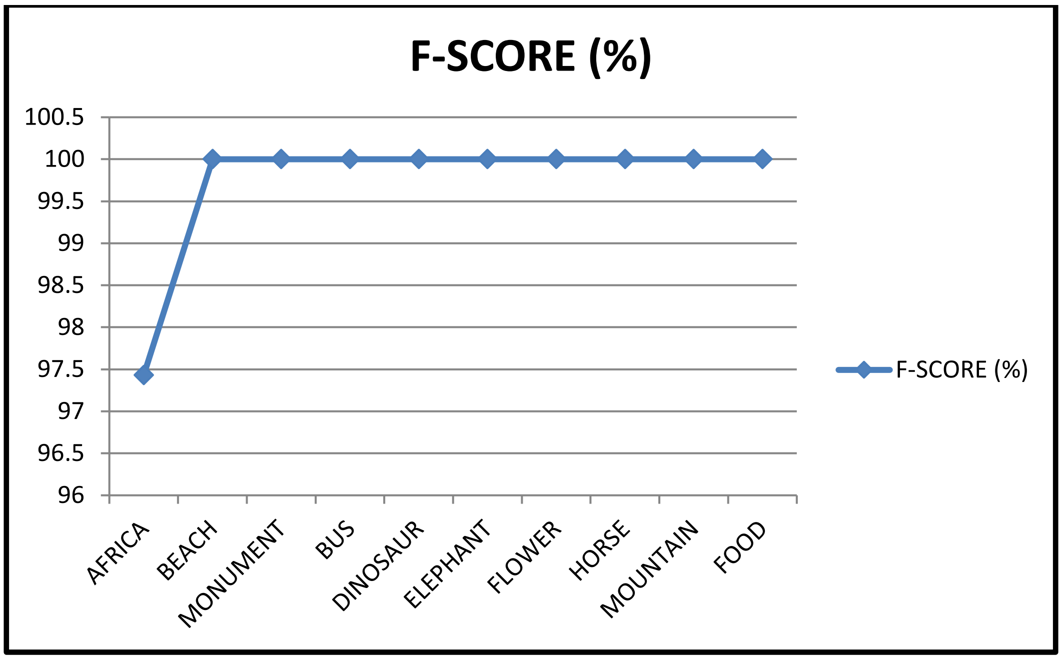


Figure 5.6: Comparison of datasets for F-score using proposed method.

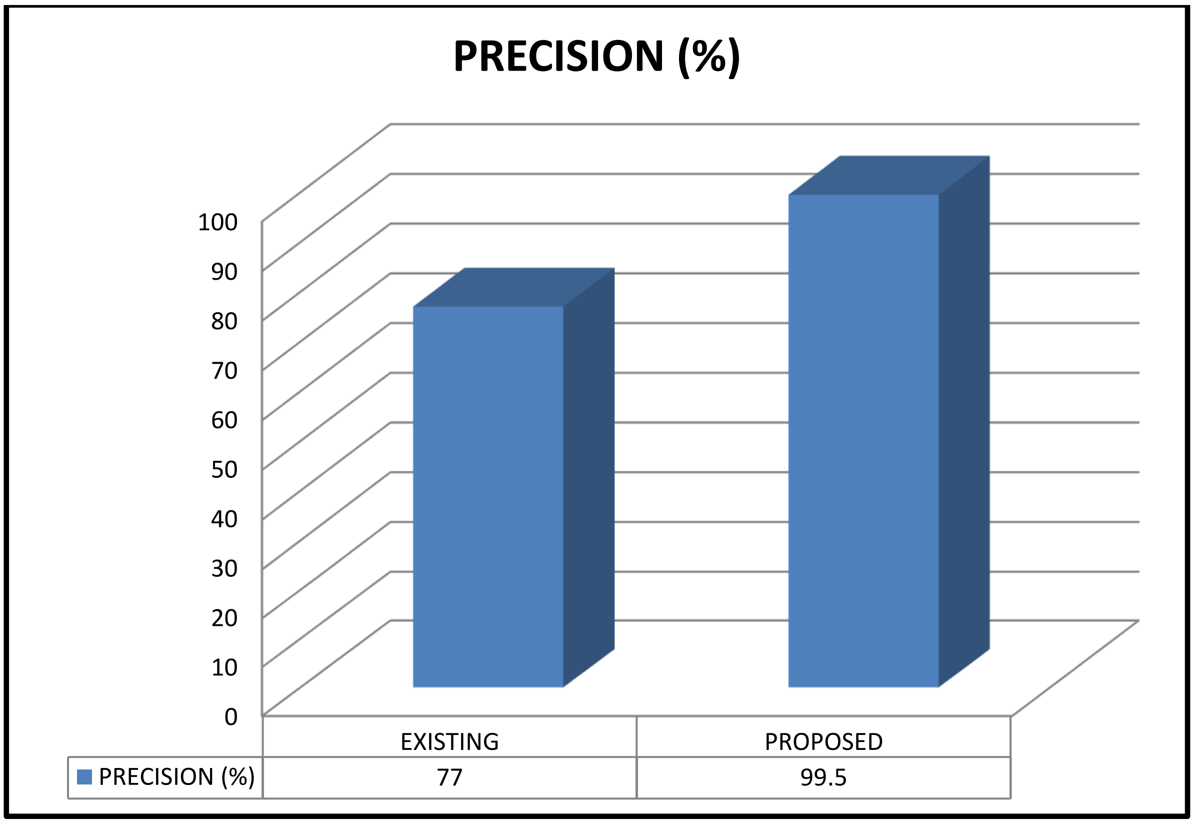
The values of F-score are graphed across y axis for the classes present in the dataset along x axis.

5.3 COMPARISON BETWEEN EXISTING AND PROPOSED

The below table shows the comparison of average Precision and average F-score values between the existing and the proposed approach.

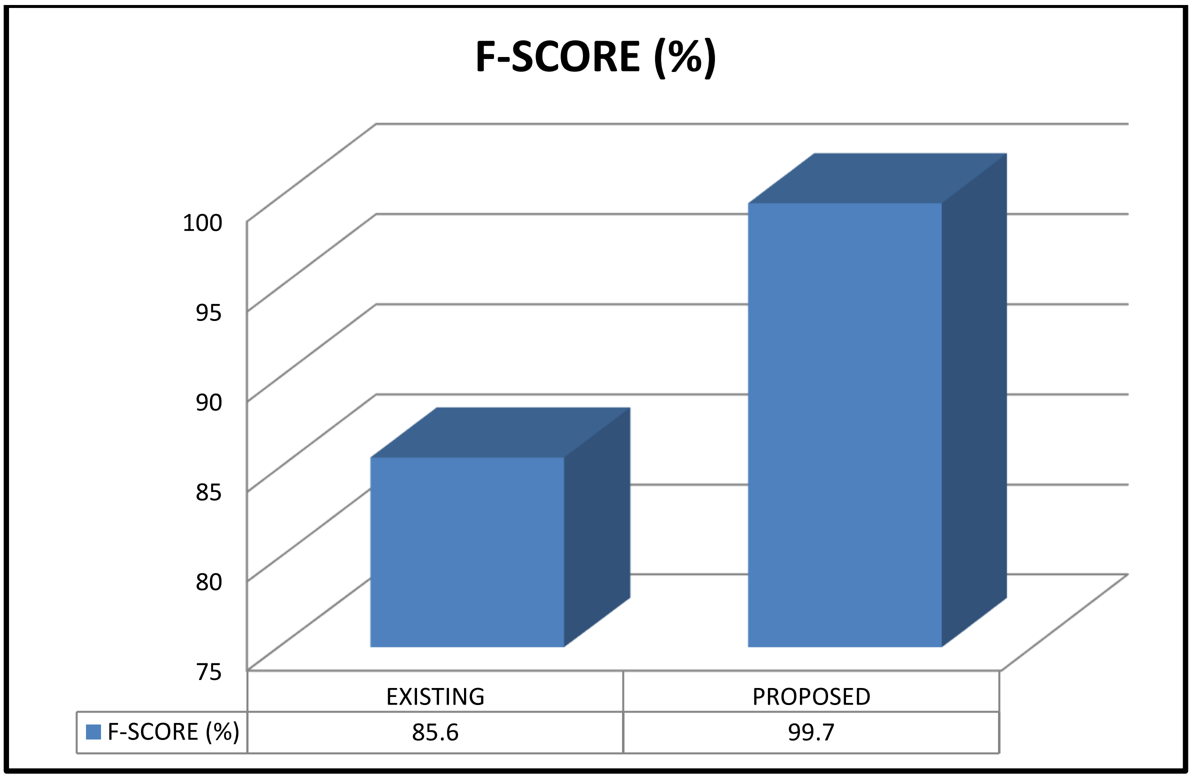
AVERAGE	SIMILARITY BASED RANKING	SUPPORT VECTOR MACHINE
PRECISION (%)	77	99.5
F-SCORE (%)	85.6	99.7

Table 5.3: Comparison of the average precision and average recall of existing and proposed approach.



	EXISTING	PROPOSED
■ PRECISION (%)	77	99.5

Figure 5.7: Comparison of average precision of existing and proposed approach.



	EXISTING	PROPOSED
■ F-SCORE (%)	85.6	99.7

Figure 5.8: Comparison of average F-score of existing and proposed approach.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

In the proposed system the aim is to retrieve similar images from the database for the given query image using the deep learning method of support vectors. The feature extraction was seen as the binary classification problem and SVM (Support Vector Machine) was used for solving this problem. SVM is used as the classifier which is performing the task of classifying the image and this process of classification is given to all the traits of the image which are extracted after the feature extraction process. It is used for estimating the highest margin hyper planes within the feature space which is also a high dimensional feature space. The mathematical description of Support Vector Machine is also explained. Compared to the existing technique precision and F-Measure scores have been improved. The time complexity is reduced.

6.2 FUTURE ENHANCEMENT

If the SVM algorithm predicts the wrong class label for a query image, then the retrieved image will be from a wrong class since the wrong label was predicted. This is hoped to be worked in the future.

APPENDIX 1

CODING

cbires.m file:

```
function varargout = cbires(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @cbires_OpeningFcn, ...
    'gui_OutputFcn',  @cbires_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function cbires_OpeningFcn(hObject, eventdata, handles, varargin)
% Choose default command line output for cbires
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
function varargout = cbires_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function btn_BrowseImage_Callback(hObject, eventdata, handles)
[query_fname, query_pathname] = uigetfile('*.jpg; *.png; *.bmp', 'Select query
image');

if (query_fname ~= 0)
    query_fullpath = strcat(query_pathname, query_fname);
    imgInfo = imfinfo(query_fullpath);
    [pathstr, name, ext] = fileparts(query_fullpath); % fiparts returns char type

    if ( strcmp(lower(ext), '.jpg') == 1 || strcmp(lower(ext), '.png') == 1 ...
            || strcmp(lower(ext), '.bmp') == 1 )

        queryImage = imread( fullfile( pathstr, strcat(name, ext) ) );
```

```
%        handles.queryImage = queryImage;
%        guidata(hObject, handles);

    % extract query image features
    queryImage = imresize(queryImage, [384 256]);
    if (strcmp(imgInfo.ColorType, 'truecolor') == 1)
        hsvHist = hsvHistogram(queryImage);
        autoCorrelogram = colorAutoCorrelogram(queryImage);
        color_moments = colorMoments(queryImage);
        % for gabor filters we need gary scale image
        img = double(rgb2gray(queryImage))/255;
        [meanAmplitude, msEnergy] = gaborWavelet(img, 4, 6); % 4 = number
of scales, 6 = number of orientations
        wavelet_moments = waveletTransform(queryImage,
imgInfo.ColorType);
        % construct the queryImage feature vector
        queryImageFeature = [hsvHist autoCorrelogram color_moments
meanAmplitude msEnergy wavelet_moments str2num(name)];
    elseif (strcmp(imgInfo.ColorType, 'grayscale') == 1)
        grayHist = imhist(queryImage);
        grayHist = grayHist/sum(grayHist);
        grayHist = grayHist(:)';
        color_moments = [mean(mean(queryImage))
std(std(double(queryImage)))];
        [meanAmplitude, msEnergy] = gaborWavelet(queryImage, 4, 6); % 4 =
number of scales, 6 = number of orientations
        wavelet_moments = waveletTransform(queryImage,
imgInfo.ColorType);
        % construct the queryImage feature vector
        queryImageFeature = [grayHist color_moments meanAmplitude
msEnergy wavelet_moments str2num(name)];
    end

    % update handles
    handles.queryImageFeature = queryImageFeature;
    handles.img_ext = ext;
    handles.folder_name = pathstr;
    guidata(hObject, handles);
    helpdlg('Proceed with the query by executing the button!');

    % Clear workspace
    clear('query_fname', 'query_pathname', 'query_fullpath', 'pathstr', ...
        'name', 'ext', 'queryImage', 'hsvHist', 'autoCorrelogram', ...
        'color_moments', 'img', 'meanAmplitude', 'msEnergy', ...
```

```
            'wavelet_moments', 'queryImageFeature', 'imgInfo');
    else
        errordlg('You have not selected the correct file type');
    end
else
    return;
end

function popupmenu_DistanceFunctions_Callback(hObject, eventdata, handles)
handles.DistanceFunctions = get(handles.popupmenu_DistanceFunctions,
'Value');
guidata(hObject, handles);

function popupmenu_DistanceFunctions_CreateFcn(hObject, eventdata,
handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function popupmenu_NumOfReturnedImages_Callback(hObject, eventdata,
handles)
handles.numOfReturnedImages =
get(handles.popupmenu_NumOfReturnedImages, 'Value');
guidata(hObject, handles);
function popupmenu_NumOfReturnedImages_CreateFcn(hObject, eventdata,
handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function btnExecuteQuery_Callback(hObject, eventdata, handles)
if (~isfield(handles, 'queryImageFeature'))
    errordlg('Please select an image first, then choose your similarity metric and
num of returned images!');
    return;
end

% check for dataset existence
if (~isfield(handles, 'imageDataset'))
    errordlg('Please load a dataset first. If you dont have one then you should
consider creating one!');
    return;
end
```

```
% set variables
if (~isfield(handles, 'DistanceFunctions') && ~isfield(handles,
'numOfReturnedImages'))
   metric = get(handles.popupmenu_DistanceFunctions, 'Value');
   numOfReturnedImgs = get(handles.popupmenu_NumOfReturnedImages,
'Value');
elseif (~isfield(handles, 'DistanceFunctions') || ~isfield(handles,
'numOfReturnedImages'))
   if (~isfield(handles, 'DistanceFunctions'))
      metric = get(handles.popupmenu_DistanceFunctions, 'Value');
      numOfReturnedImgs = handles.numOfReturnedImages;
   else
      metric = handles.DistanceFunctions;
      numOfReturnedImgs = get(handles.popupmenu_NumOfReturnedImages,
'Value');
   end
else
   metric = handles.DistanceFunctions;
   numOfReturnedImgs = handles.numOfReturnedImages;
end

if (metric == 1)
   L1(numOfReturnedImgs, handles.queryImageFeature,
handles.imageDataset.dataset, handles.folder_name, handles.img_ext);
elseif (metric == 2 || metric == 3 || metric == 4 || metric == 5 || metric == 6  ||
metric == 7 || metric == 8 || metric == 9 || metric == 10 || metric == 11)
   L2(numOfReturnedImgs, handles.queryImageFeature,
handles.imageDataset.dataset, metric, handles.folder_name, handles.img_ext);
else
   relativeDeviation(numOfReturnedImgs, handles.queryImageFeature,
handles.imageDataset.dataset, handles.folder_name, handles.img_ext);
end


% --- Executes on button press in btnExecuteSVM.
function btnExecuteSVM_Callback(hObject, eventdata, handles)
% hObject    handle to btnExecuteSVM (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% check for image query
if (~isfield(handles, 'queryImageFeature'))
   errordlg('Please select an image first!');
   return;
```

```
end

% check for dataset existence
if (~isfield(handles, 'imageDataset'))
    errordlg('Please load a dataset first. If you dont have one then you should
consider creating one!');
    return;
end

numOfReturnedImgs = get(handles.popupmenu_NumOfReturnedImages,
'Value');
metric = get(handles.popupmenu_DistanceFunctions, 'Value');

% call svm function passing as parameters the numOfReturnedImgs,
queryImage and the dataset
[~, ~, cmat] = svm(numOfReturnedImgs, handles.imageDataset.dataset,
handles.queryImageFeature, metric, handles.folder_name, handles.img_ext);

% plot confusion matrix
opt = confMatPlot('defaultOpt');
opt.className = {
    'Africa', 'Beach', 'Monuments', ...
    'Buses', 'Dinosaurs', 'Elephants', ...
    'Flowers', 'Horses', 'Mountains', ...
    'Food'
    };
opt.mode = 'both';
figure('Name', 'Confusion Matrix');
confMatPlot(cmat, opt);
xlabel('Confusion Matrix');

function btnPlotPrecisionRecall_Callback(hObject, eventdata, handles)
if (~isfield(handles, 'imageDataset'))
    errordlg('Please select a dataset first!');
    return;
end


% set variables
numOfReturnedImgs = 20;
database = handles.imageDataset.dataset;
metric =  get(handles.popupmenu_DistanceFunctions, 'Value');

precAndRecall = zeros(2, 10);
```

```
for k = 1:15
   randImgName = randi([0 999], 1);
   randStrName = int2str(randImgName);
   randStrName = strcat('images\', randStrName, '.jpg');
   randQueryImg = imread(randStrName);

   % extract query image features
   queryImage = imresize(randQueryImg, [384 256]);
   hsvHist = hsvHistogram(queryImage);
   autoCorrelogram = colorAutoCorrelogram(queryImage);
   color_moments = colorMoments(queryImage);
   % for gabor filters we need gary scale image
   img = double(rgb2gray(queryImage))/255;
   [meanAmplitude, msEnergy] = gaborWavelet(img, 4, 6); % 4 = number of
scales, 6 = number of orientations
   wavelet_moments = waveletTransform(queryImage, imgInfo.ColorType);
   % construct the queryImage feature vector
   queryImageFeature = [hsvHist autoCorrelogram color_moments
meanAmplitude msEnergy wavelet_moments randImgName];

   disp(['Random Image = ', num2str(randImgName), '.jpg']);
   [precision, recall] = svm(numOfReturnedImgs, database, queryImageFeature,
metric);
   precAndRecall(1, k) = precision;
   precAndRecall(2, k) = recall;
end

figure;
plot(precAndRecall(2, :), precAndRecall(1, :), '--mo');
xlabel('Recall'), ylabel('Precision');
title('Precision and Recall');
legend('Recall & Precision', 'Location', 'NorthWest');
function btnSelectImageDirectory_Callback(hObject, eventdata, handles)

folder_name = uigetdir(pwd, 'Select the directory of images');
if ( folder_name ~= 0 )
   handles.folder_name = folder_name;
   guidata(hObject, handles);
else
   return;
end

function btnCreateDB_Callback(hObject, eventdata, handles)
```

```
if (~isfield(handles, 'folder_name'))
    errordlg('Please select an image directory first!');
    return;
end

% construct folder name foreach image type
pngImagesDir = fullfile(handles.folder_name, '*.png');
jpgImagesDir = fullfile(handles.folder_name, '*.jpg');
bmpImagesDir = fullfile(handles.folder_name, '*.bmp');

% calculate total number of images
num_of_png_images = numel( dir(pngImagesDir) );
num_of_jpg_images = numel( dir(jpgImagesDir) );
num_of_bmp_images = numel( dir(bmpImagesDir) );
totalImages = num_of_png_images + num_of_jpg_images +
num_of_bmp_images;

jpg_files = dir(jpgImagesDir);
png_files = dir(pngImagesDir);
bmp_files = dir(bmpImagesDir);

if ( ~isempty( jpg_files ) || ~isempty( png_files ) || ~isempty( bmp_files ) )
    % read jpg images from stored folder name
    % directory and construct the feature dataset
    jpg_counter = 0;
    png_counter = 0;
    bmp_counter = 0;
    for k = 1:totalImages

        if ( (num_of_jpg_images - jpg_counter) > 0)
            imgInfoJPG = imfinfo( fullfile( handles.folder_name,
jpg_files(jpg_counter+1).name ) );
            if ( strcmp( lower(imgInfoJPG.Format), 'jpg') == 1 )
                % read images
                sprintf('%s \n', jpg_files(jpg_counter+1).name)
                % extract features
                image = imread( fullfile( handles.folder_name,
jpg_files(jpg_counter+1).name ) );
                [pathstr, name, ext] = fileparts( fullfile( handles.folder_name,
jpg_files(jpg_counter+1).name ) );
                image = imresize(image, [384 256]);
            end

            jpg_counter = jpg_counter + 1;
        elseif ( (num_of_png_images - png_counter) > 0)
            imgInfoPNG = imfinfo( fullfile( handles.folder_name,
png_files(png_counter+1).name ) );
            if ( strcmp( lower(imgInfoPNG.Format), 'png') == 1 )
                % read images
                sprintf('%s \n', png_files(png_counter+1).name)
                % extract features
                image = imread( fullfile( handles.folder_name,
png_files(png_counter+1).name ) );
                [pathstr, name, ext] = fileparts( fullfile( handles.folder_name,
png_files(png_counter+1).name ) );
                image = imresize(image, [384 256]);
            end

            png_counter = png_counter + 1;

        elseif ( (num_of_bmp_images - bmp_counter) > 0)
            imgInfoBMP = imfinfo( fullfile( handles.folder_name,
bmp_files(bmp_counter+1).name ) );
            if ( strcmp( lower(imgInfoBMP.Format), 'bmp') == 1 )
                % read images
                sprintf('%s \n', bmp_files(bmp_counter+1).name)
                % extract features
                image = imread( fullfile( handles.folder_name,
bmp_files(bmp_counter+1).name ) );
                handle = image(image);
                imgmodel = imagemodel(handle);
                str = getImageType(imgmodel);
                disp([str])
                return;

                [pathstr, name, ext] = fileparts( fullfile( handles.folder_name,
bmp_files(bmp_counter+1).name ) );
                image = imresize(image, [384 256]);
            end

            bmp_counter = bmp_counter + 1;

        end

        switch (ext)
            case '.jpg'
                imgInfo = imgInfoJPG;
```

```
          case '.png'
             imgInfo = imgInfoPNG;
          case '.bmp'
             imgInfo = imgInfoBMP;
       end

       if (strcmp(imgInfo.ColorType, 'grayscale') == 1)
          grayHist = imhist(image);
          grayHist = grayHist/sum(grayHist);
          grayHist = grayHist(:)';
          color_moments = [mean(mean(image)) std(std(double(image)))];
          [meanAmplitude, msEnergy] = gaborWavelet(image, 4, 6); % 4 =
number of scales, 6 = number of orientations
          wavelet_moments = waveletTransform(image, imgInfo.ColorType);
          % construct the dataset
          set = [grayHist color_moments meanAmplitude msEnergy
wavelet_moments];
       elseif (strcmp(imgInfo.ColorType, 'truecolor') == 1)
          hsvHist = hsvHistogram(image);
          autoCorrelogram = colorAutoCorrelogram(image);
          color_moments = colorMoments(image);
          % for gabor filters we need gray scale image
          img = double(rgb2gray(image))/255;
          [meanAmplitude, msEnergy] = gaborWavelet(img, 4, 6); % 4 = number
of scales, 6 = number of orientations
          wavelet_moments = waveletTransform(image, imgInfo.ColorType);
          % construct the dataset
          set = [hsvHist autoCorrelogram color_moments meanAmplitude
msEnergy wavelet_moments];
       end

       % add to the last column the name of image file we are processing at
       % the moment
       dataset(k, :) = [set str2num(name)];

       % clear workspace
       clear('image', 'img', 'hsvHist', 'autoCorrelogram', 'color_moments', ...
          'gabor_wavelet', 'wavelet_moments', 'set', 'imgInfoJPG', 'imgInfoPNG',
...
          'imgInfoGIF', 'imgInfo');
    end

    % prompt to save dataset
    uisave('dataset', 'dataset1');
    % save('dataset.mat', 'dataset', '-mat');
    clear('dataset', 'jpg_counter', 'png_counter', 'bmp_counter');
end


% --- Executes on button press in btn_LoadDataset.
function btn_LoadDataset_Callback(hObject, eventdata, handles)
% hObject    handle to btn_LoadDataset (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[fname, pthname] = uigetfile('*.mat', 'Select the Dataset');
if (fname ~= 0)
    dataset_fullpath = strcat(pthname, fname);
    [pathstr, name, ext] = fileparts(dataset_fullpath);
    if ( strcmp(lower(ext), '.mat') == 1)
        filename = fullfile( pathstr, strcat(name, ext) );
        handles.imageDataset = load(filename);
        guidata(hObject, handles);
        % make dataset visible from workspace
        % assignin('base', 'database', handles.imageDataset.dataset);
        helpdlg('Dataset loaded successfuly!');
    else
        errordlg('You have not selected the correct file type');
    end
else
    return;
end
```

svm.m file:

```
function [precision, recall, cmat] = svm(numOfReturnedImgs, dataset,
queryImageFeatureVector, metric, folder_name, img_ext)
img_names = dataset(:, end);
dataset(:, end) = [];

% extract image name from queryImageFeatureVector
query_img_name = queryImageFeatureVector(:, end);
queryImageFeatureVector(:, end) = [];

% construct labels
lbls = zeros(length(dataset), 1);
for k = 0:length(lbls)-1
    if (img_names(k+1) >= 0 && img_names(k+1) <= 99)
        lbls(k+1) = 1;
    elseif (img_names(k+1) > 99 && img_names(k+1) <= 199)
        lbls(k+1) = 2;
    elseif (img_names(k+1) > 199 && img_names(k+1) <= 299)
        lbls(k+1) = 3;
    elseif (img_names(k+1) > 299 && img_names(k+1) <= 399)
        lbls(k+1) = 4;
    elseif (img_names(k+1) > 399 && img_names(k+1) <= 499)
        lbls(k+1) = 5;
    elseif (img_names(k+1) > 499 && img_names(k+1) <= 599)
        lbls(k+1) = 6;
    elseif (img_names(k+1) > 599 && img_names(k+1) <= 699)
        lbls(k+1) = 7;
    elseif (img_names(k+1) > 699 && img_names(k+1) <= 799)
        lbls(k+1) = 8;
    elseif (img_names(k+1) > 799 && img_names(k+1) <= 899)
        lbls(k+1) = 9;
    elseif (img_names(k+1) > 899 && img_names(k+1) <= 999)
        lbls(k+1) = 10;
    end
end

[g gn] = grp2idx(lbls);                     %# nominal class to numeric

%# split training/testing sets
[trainIdx testIdx] = crossvalind('HoldOut', lbls, 1/2); % split the train and test
labels 50%-50%

pairwise = nchoosek(1:size(gn, 1), 2);           %# 1-vs-1 pairwise models
svmModel = cell(size(pairwise, 1), 1);           %# store binary-classifers
predTest = zeros(sum(testIdx), numel(svmModel)); %# store binary predictions

%# classify using one-against-one approach, SVM with 3rd degree poly kernel
for k=1:numel(svmModel)
    %# get only training instances belonging to this pair
    idx = trainIdx & any( bsxfun(@eq, g, pairwise(k,:)) , 2 );

    %# train
%     svmModel{k} = svmtrain(dataset(idx,:), g(idx), ...
%         'BoxConstraint',2e-1, 'Kernel_Function','polynomial', 'Polyorder',3);
    svmModel{k} = svmtrain(dataset(idx,:), g(idx), ...
        'BoxConstraint', Inf, 'Kernel_Function', 'rbf', 'rbf_sigma', 14.51);

    %# test
    predTest(:,k) = svmclassify(svmModel{k}, dataset(testIdx,:)); % matlab
native svm function
end
pred = mode(predTest, 2);   %# voting: clasify as the class receiving most votes

%# performance
cmat = confusionmat(g(testIdx), pred); %# g(testIdx) == targets, pred ==
outputs
final_acc = 100*sum(diag(cmat))./sum(cmat(:));
fprintf('SVM (1-against-1):\naccuracy = %.2f%%\n', final_acc);
fprintf('Confusion Matrix:\n'), disp(cmat)
% assignin('base', 'cmatrix', cmat);

% Precision and recall
% 1st class
precision = zeros(size(gn, 1), 1);
recall = zeros(size(gn, 1), 1);

precision = cmat(1, 1)/sum(cmat(:, 1)); % tp/tp+fp, where tp = true positive, fp
= false positive
recall = cmat(1, 1)/sum(cmat(1, :)); % tp/tp+fn, where fn = false negatives
% % 2nd class and forward
for c = 2:size(gn, 1)
    precision(c) = cmat(c, c)/sum(cmat(c:end, c));
    recall(c) = cmat(c, c)/sum(cmat(c, c:end));
end

% verify predictions
% dataset = [dataset img_names lbls];
% testData = dataset(testIdx, :);
% classesInTestData = sort(testData(:, end)); % 500 samples from 10 classes
% predictedImgs = dataset(pred, :);
% dataset(:, end) = [];

for k = 1:numel(svmModel)
    %# test
    predQueryImg(:, k) = svmclassify(svmModel{k},
queryImageFeatureVector); % queryImage = x.jpg, row=x from dataset
end
predFinalQueryImg = mode(predQueryImg, 2); % predicted final image in class
x using voting
fprintf('Predicted Query Image Belongs to Class = %d\n', predFinalQueryImg);

% take all images from dataset that belong to class x
```

```
dataset = [dataset img_names lbls];
imgsInClassX = dataset( find( dataset(:, end) == predFinalQueryImg ), : );

% Perform knn with queryImage and imgsInClassX
imgsInClassXWithoutLbls = imgsInClassX;
imgsInClassXWithoutLbls(:, end) = [];
% imgsInClassXWithoutLbls(:, end) = [];

L2(numOfReturnedImgs, [queryImageFeatureVector query_img_name],
imgsInClassXWithoutLbls, metric, folder_name, img_ext);

end
```

APPENDIX 2

SCREENSHOTS

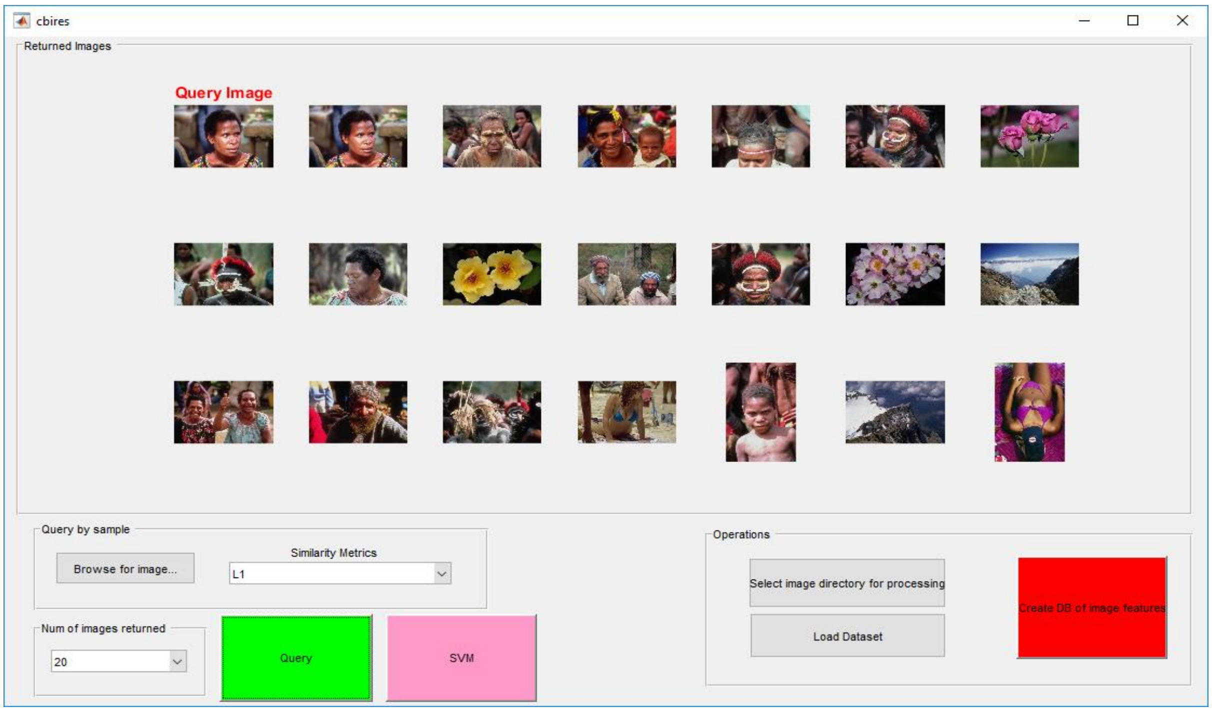


Figure A.1: Retrieved images in existing system when query image is from the Africa class.

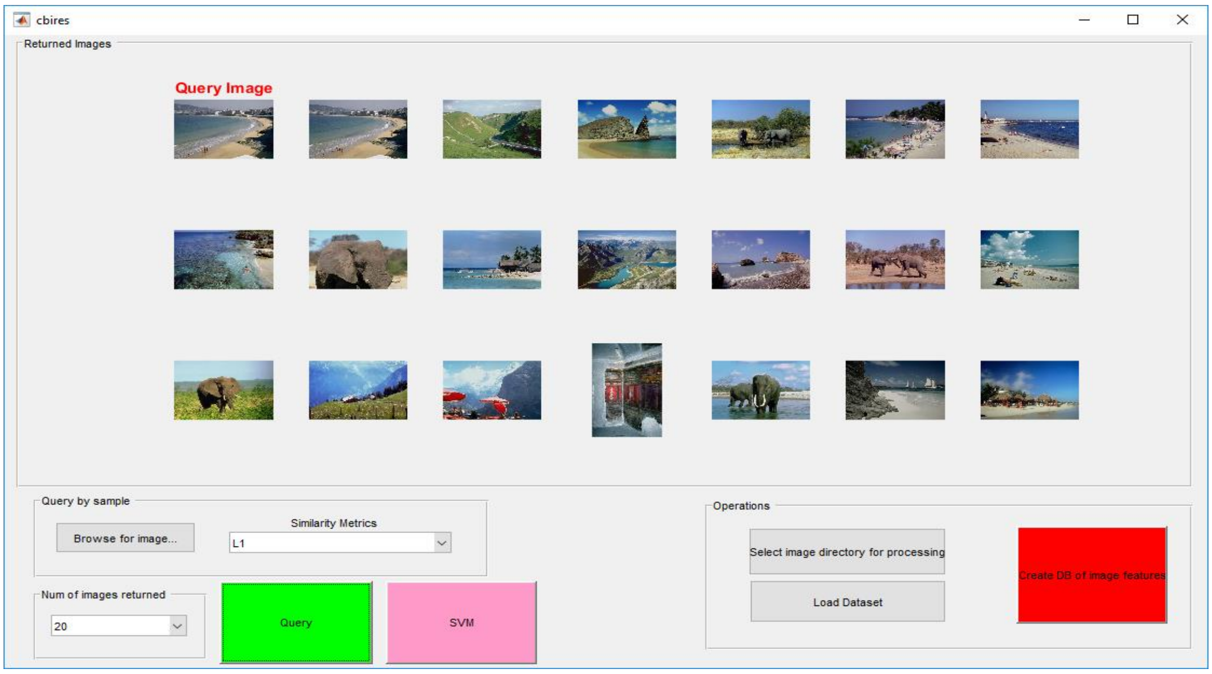


Figure A.2: Retrieved images in existing system when query image is from the Beach class.

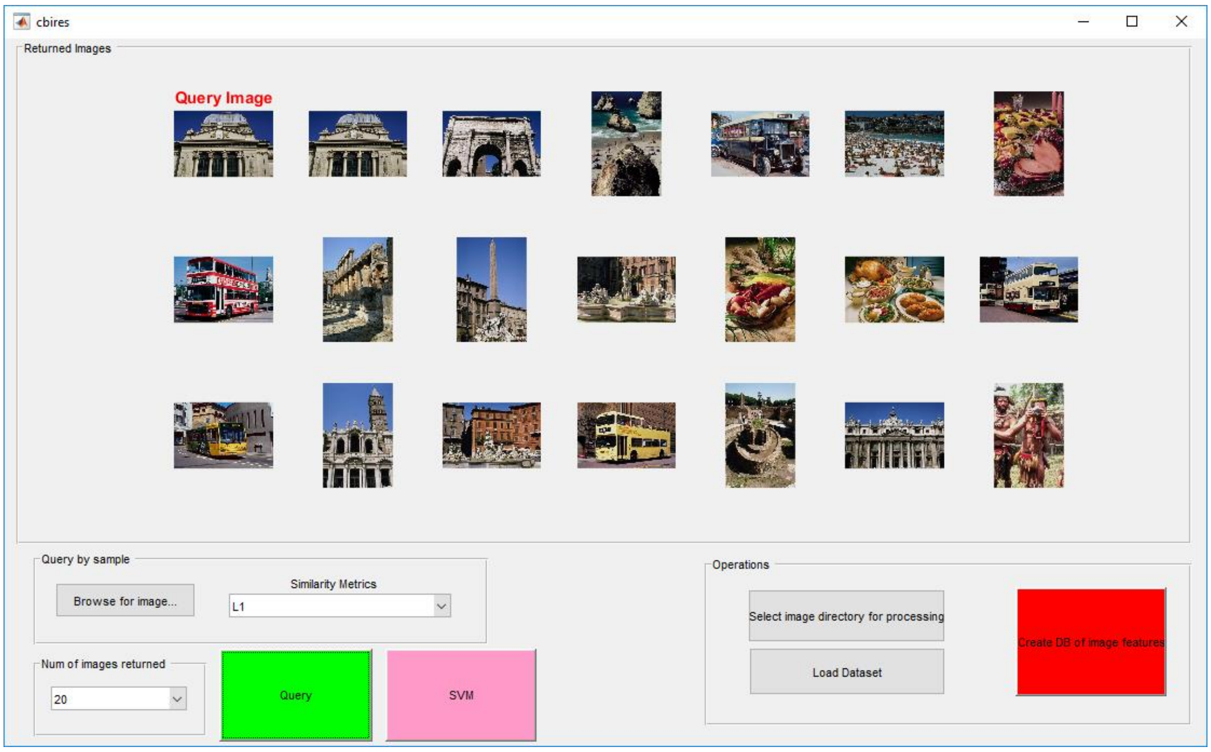


Figure A.3: Retrieved images in existing system when query image is from the Monument class.

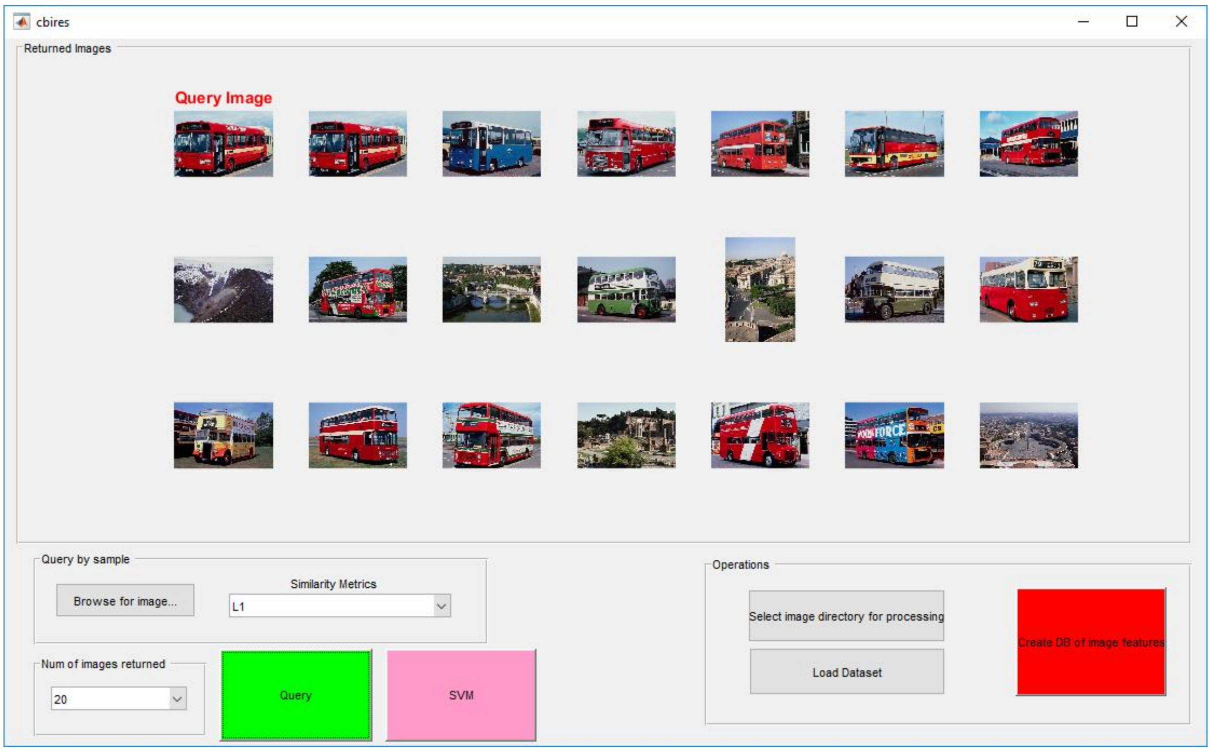


Figure A.4: Retrieved images in existing system when query image is from the Bus class.

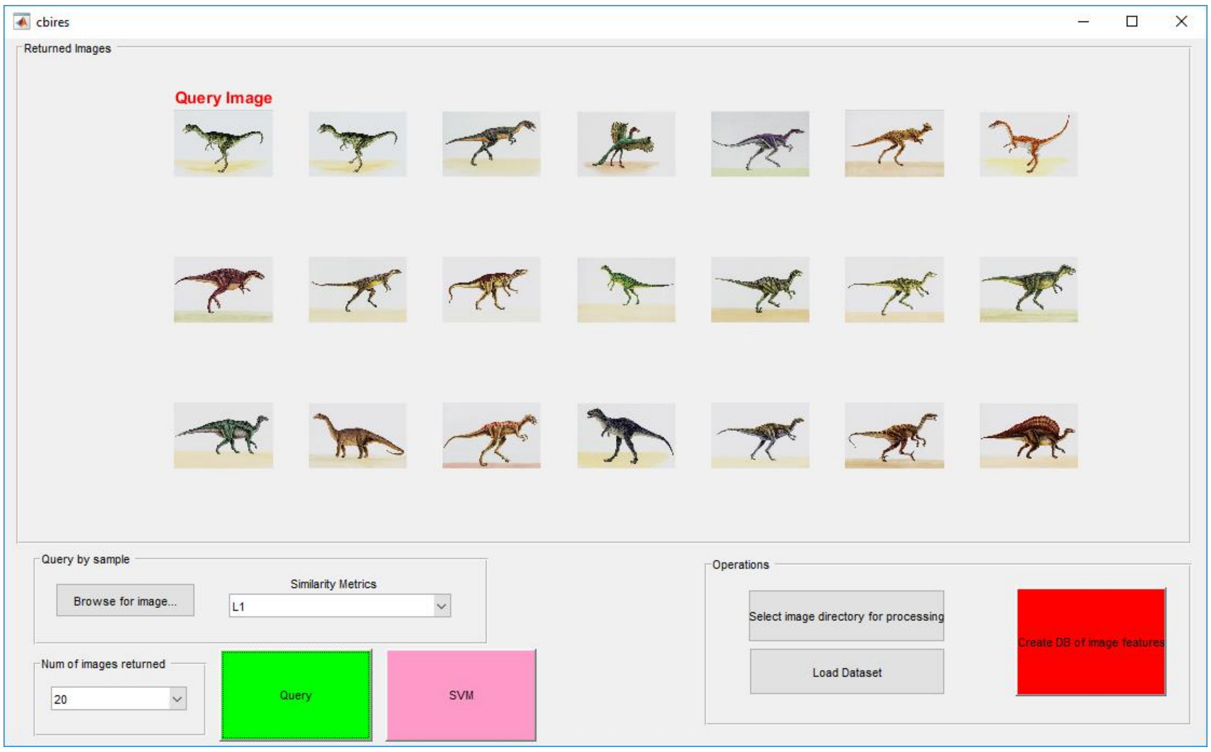


Figure A.5: Retrieved images in existing system when query image is from the Dinosaur class.

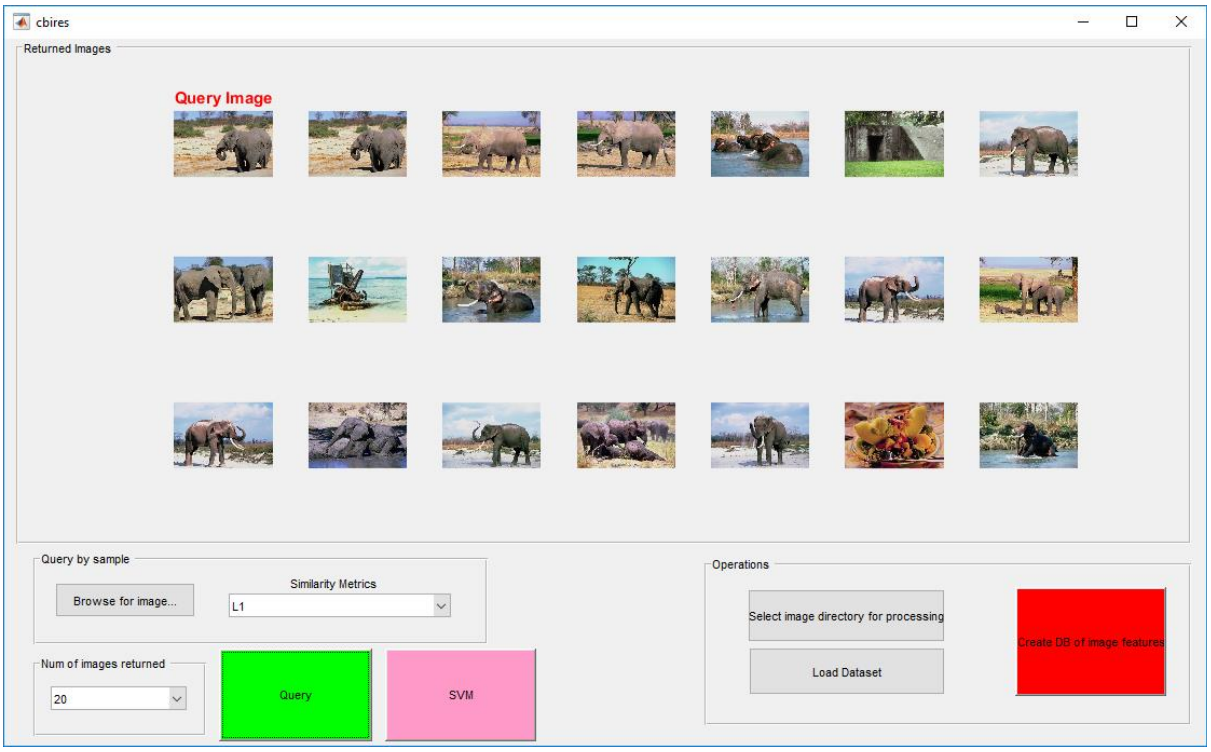


Figure A.6: Retrieved images in existing system when query image is from the Elephant class.

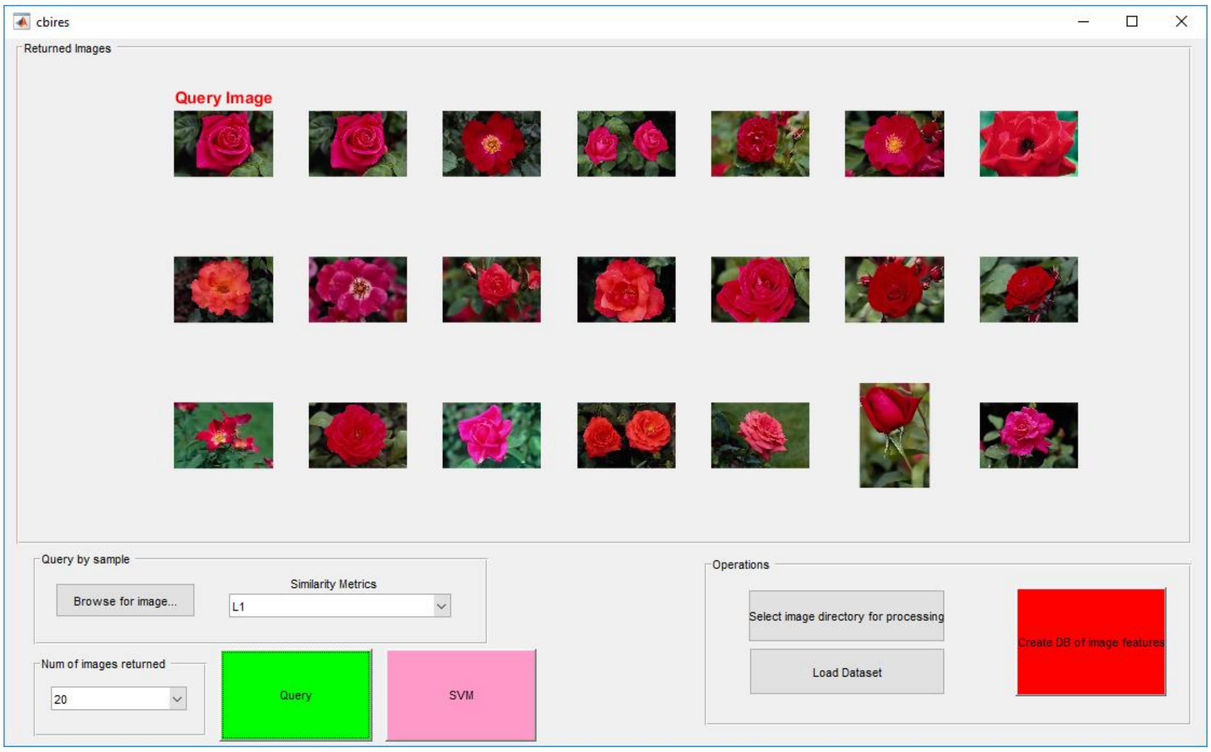


Figure A.7: Retrieved images in existing system when query image is from the Flower class.

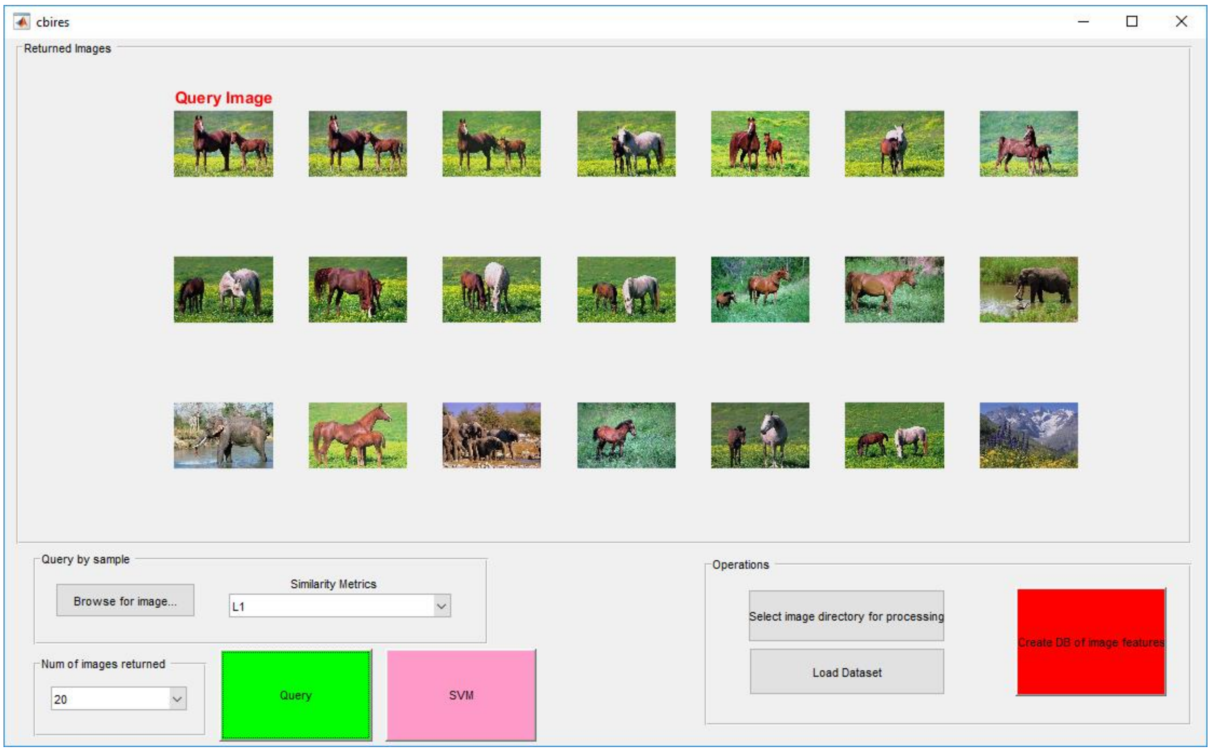


Figure A.8: Retrieved images in existing system when query image is from the Horse class.

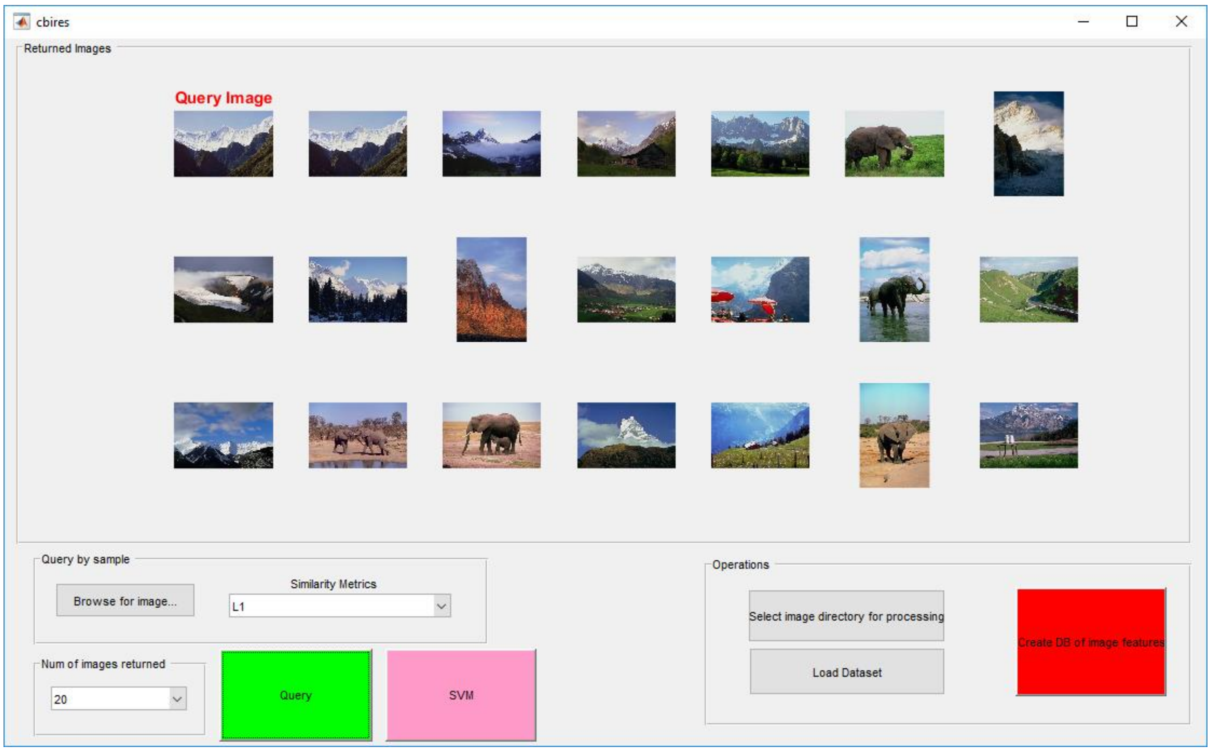


Figure A.9: Retrieved images in existing system when query image is from the Mountain class.

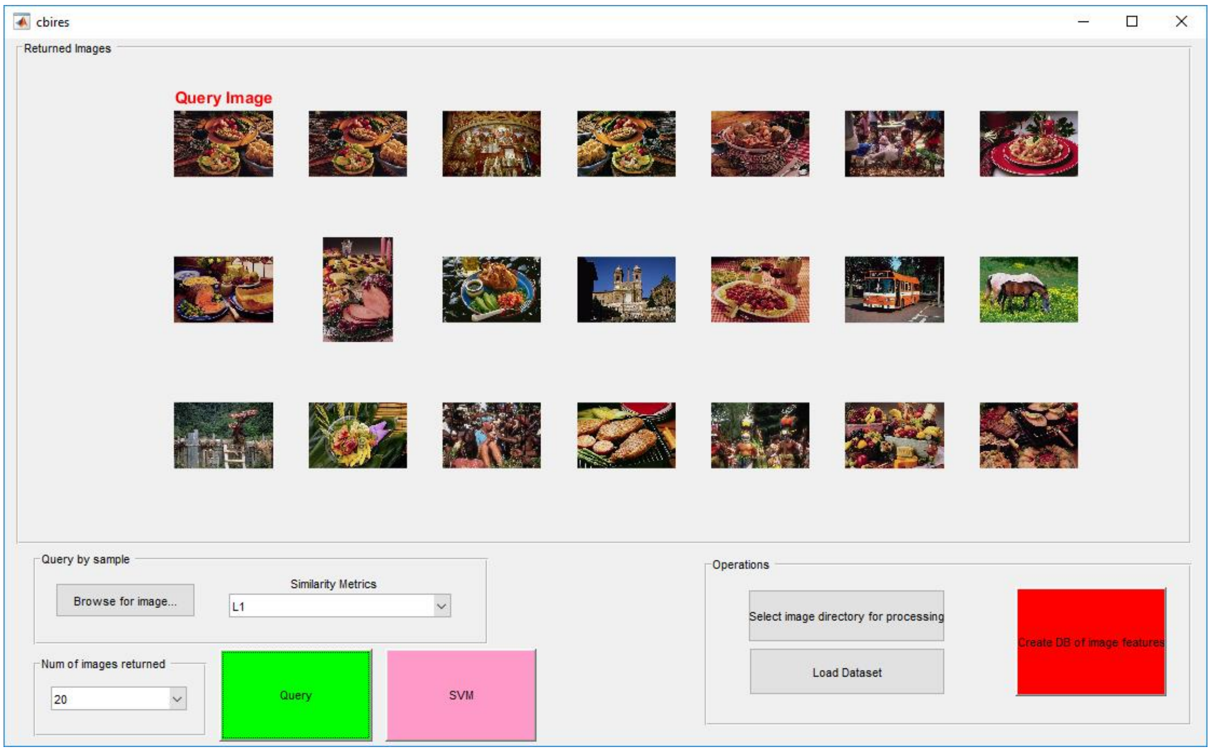


Figure A.10: Retrieved images in existing system when query image is from the Food class.

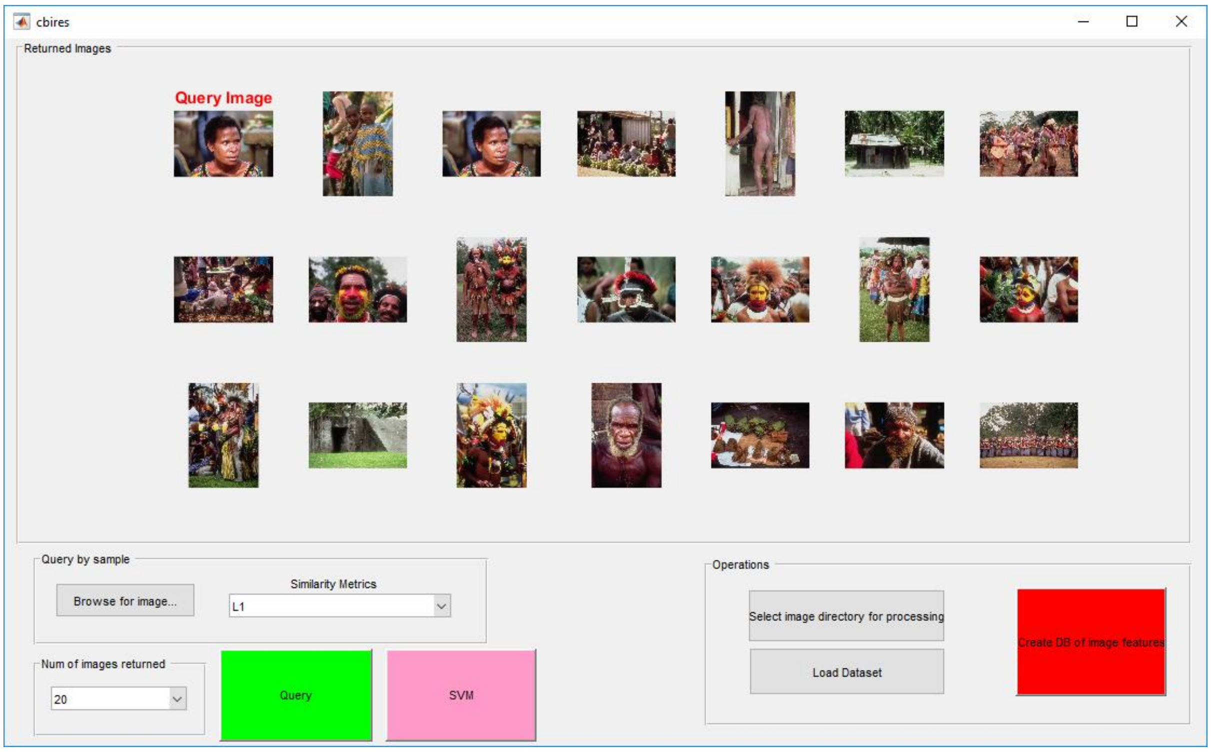


Figure A.11: Retrieved images of proposed system when query image is from the Africa class.

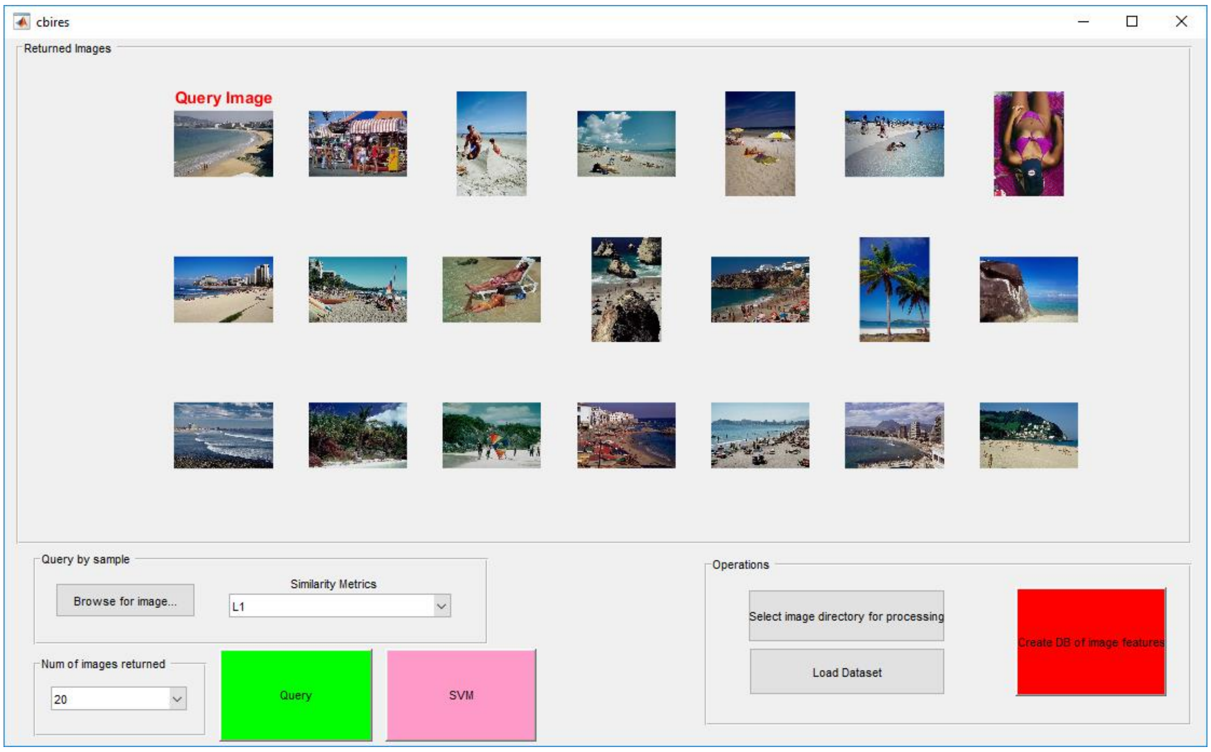


Figure A.12: Retrieved images of proposed system when query image is from the Beach class.

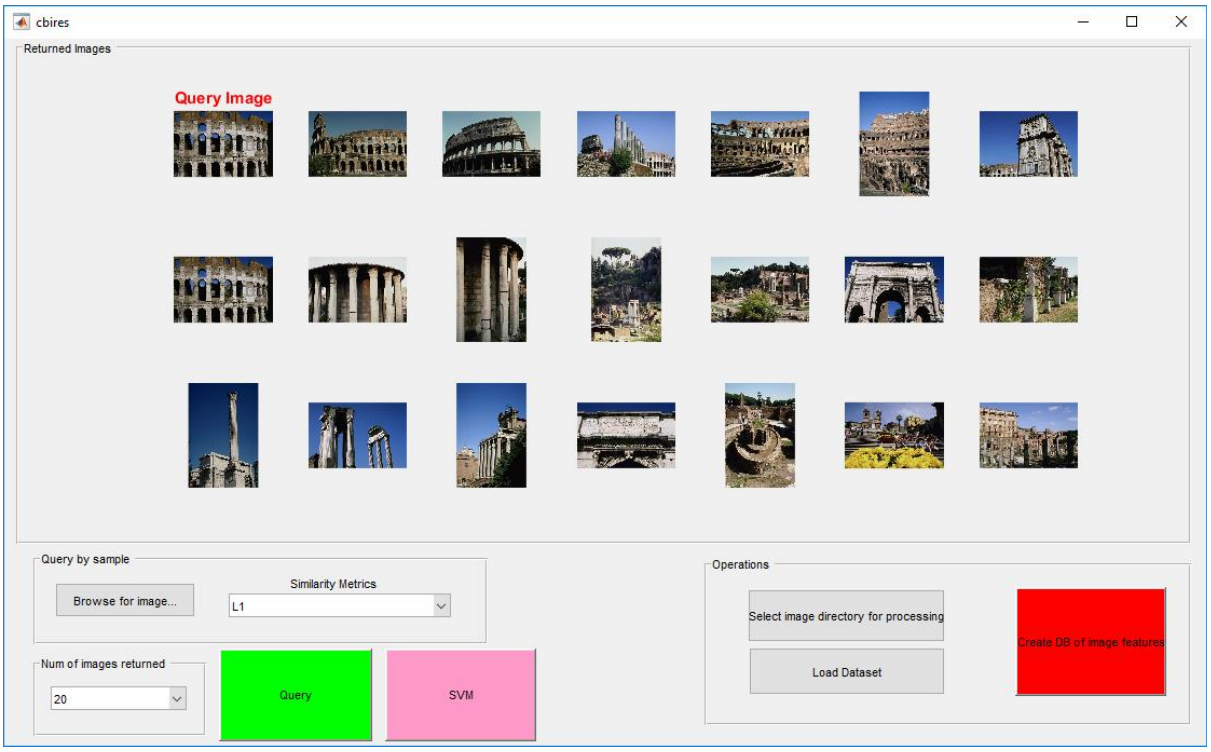


Figure A.13: Retrieved images of proposed system when query image is from the Monument class.

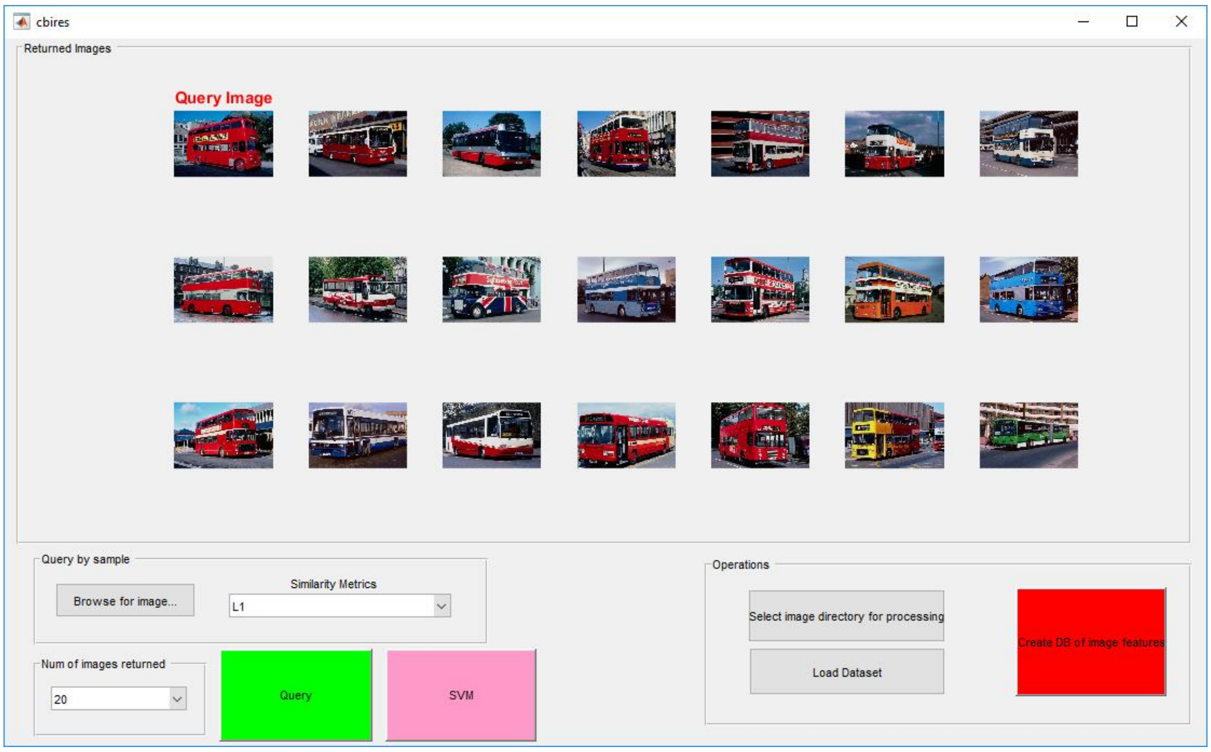


Figure A.14: Retrieved images of proposed system when query image is from the Bus class.

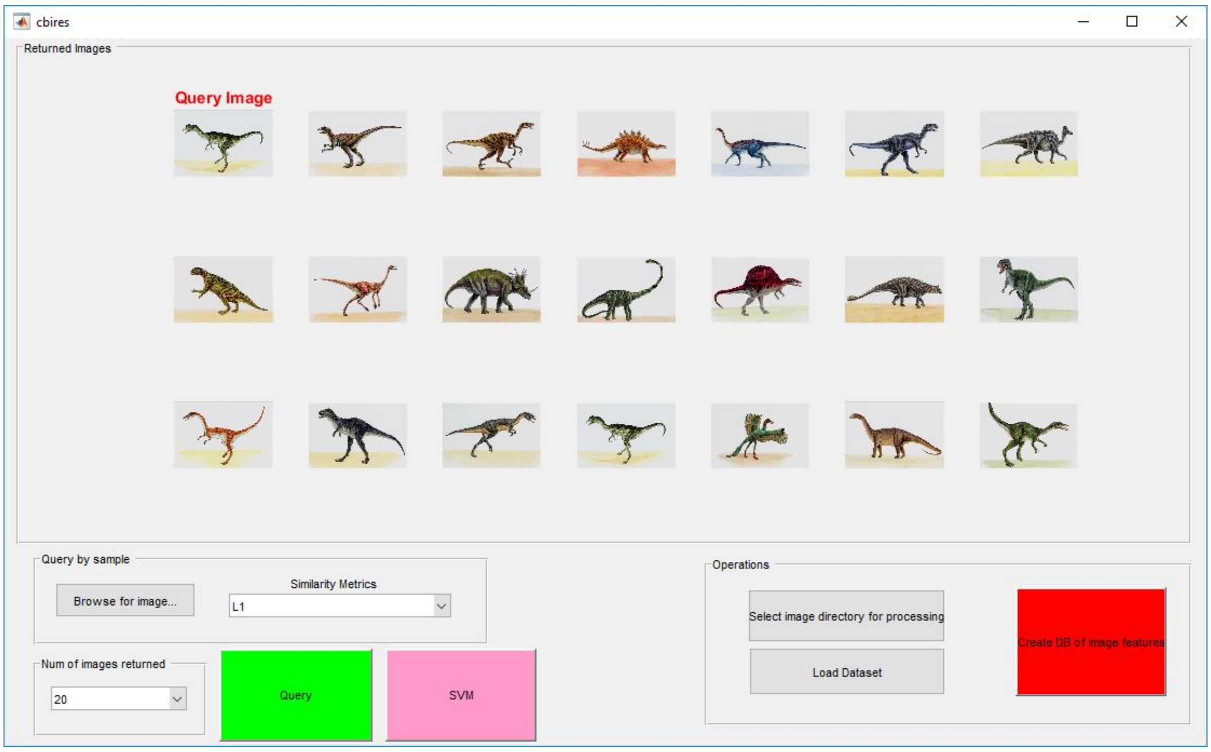


Figure A.15: Retrieved images of proposed system when query image is from the Dinosaur class.

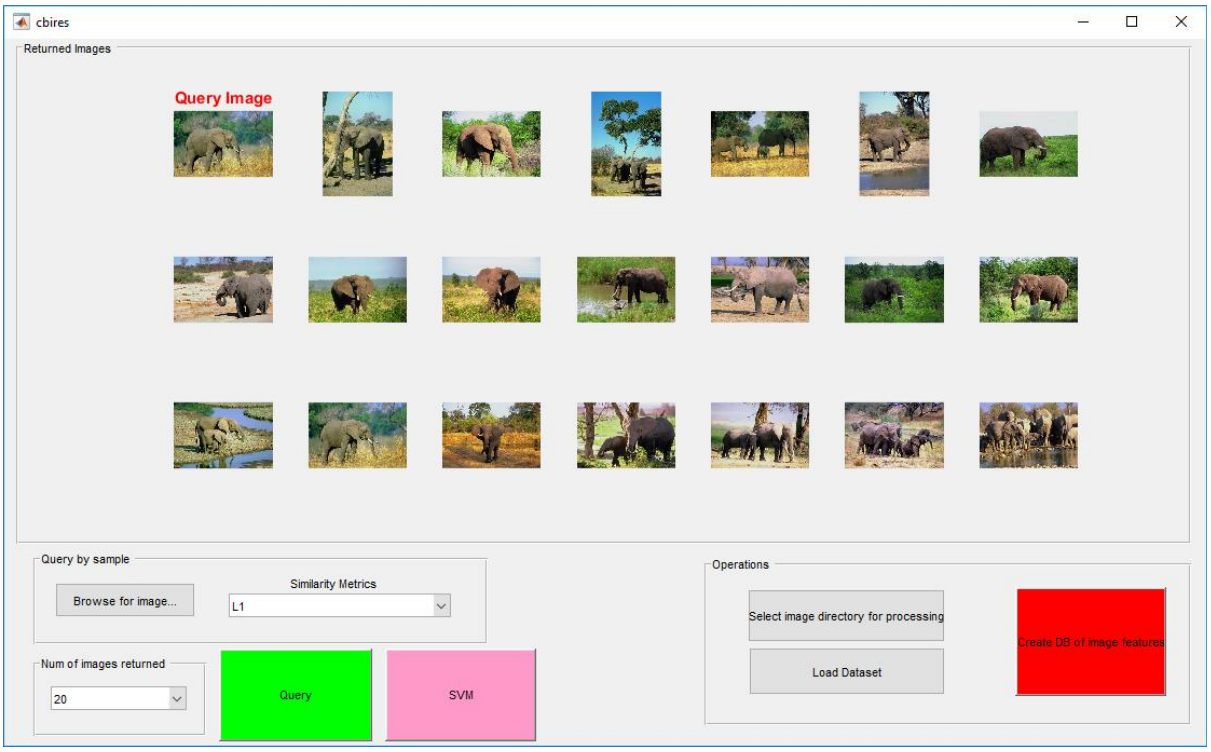


Figure A.16: Retrieved images of proposed system when query image is from the Elephant class.

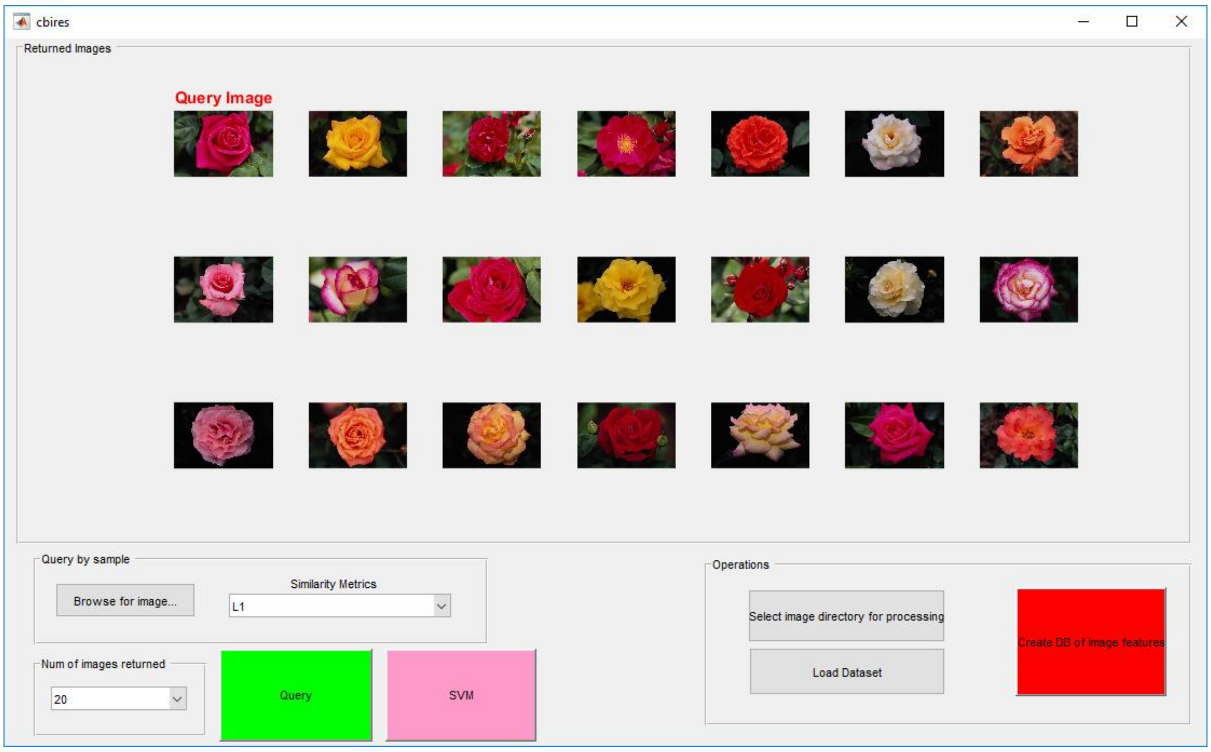


Figure A.17: Retrieved images of proposed system when query image is from the Flower class.

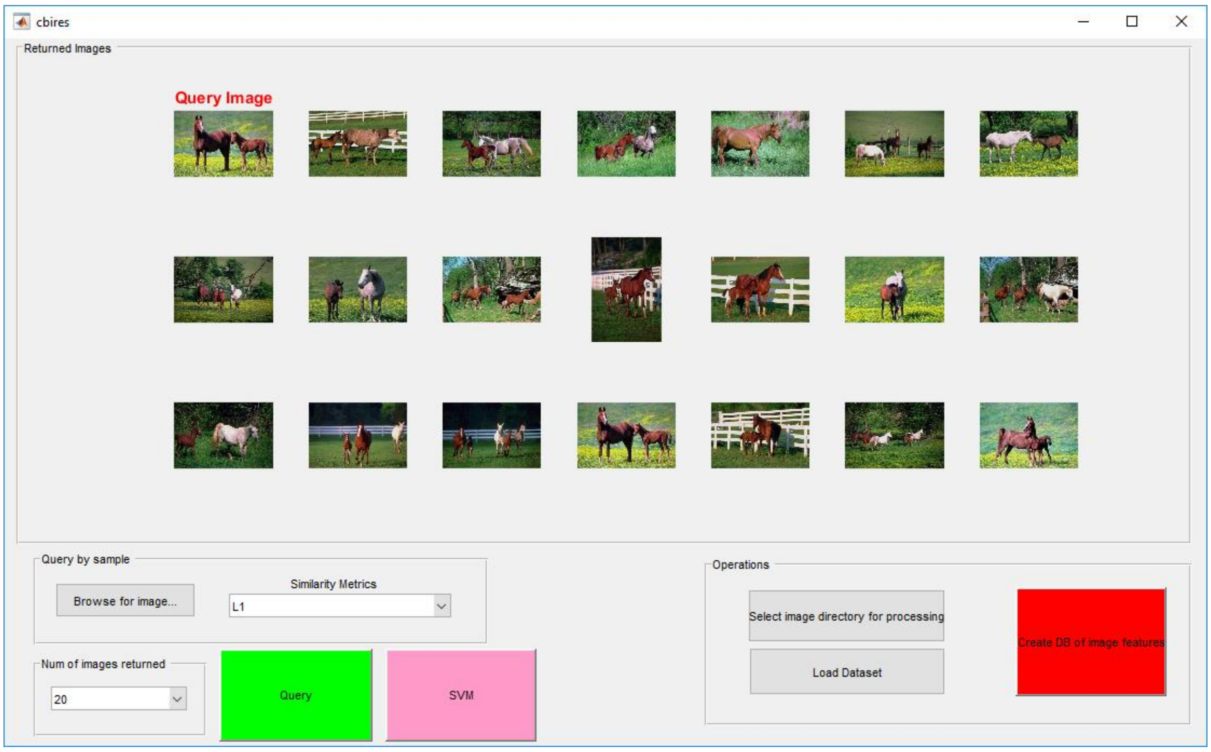


Figure A.18: Retrieved images of proposed system when query image is from the Horse class.

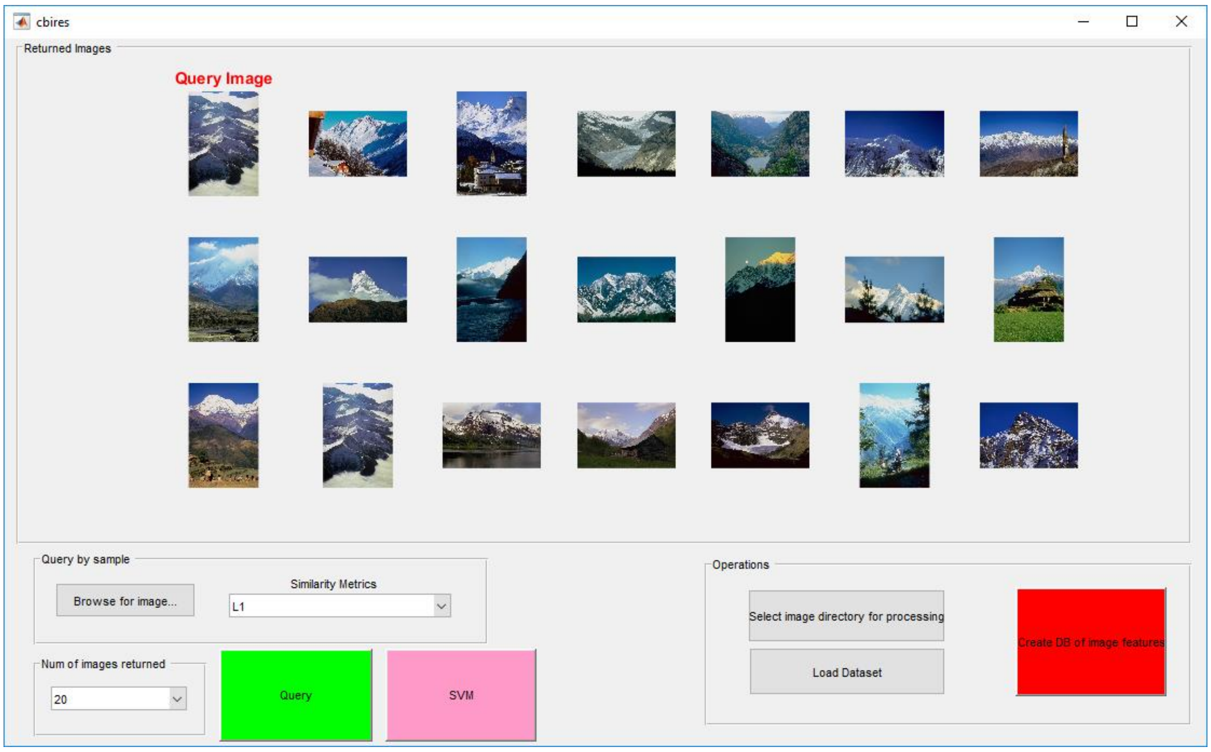


Figure A.19: Retrieved images of proposed system when query image is from the Mountain class.

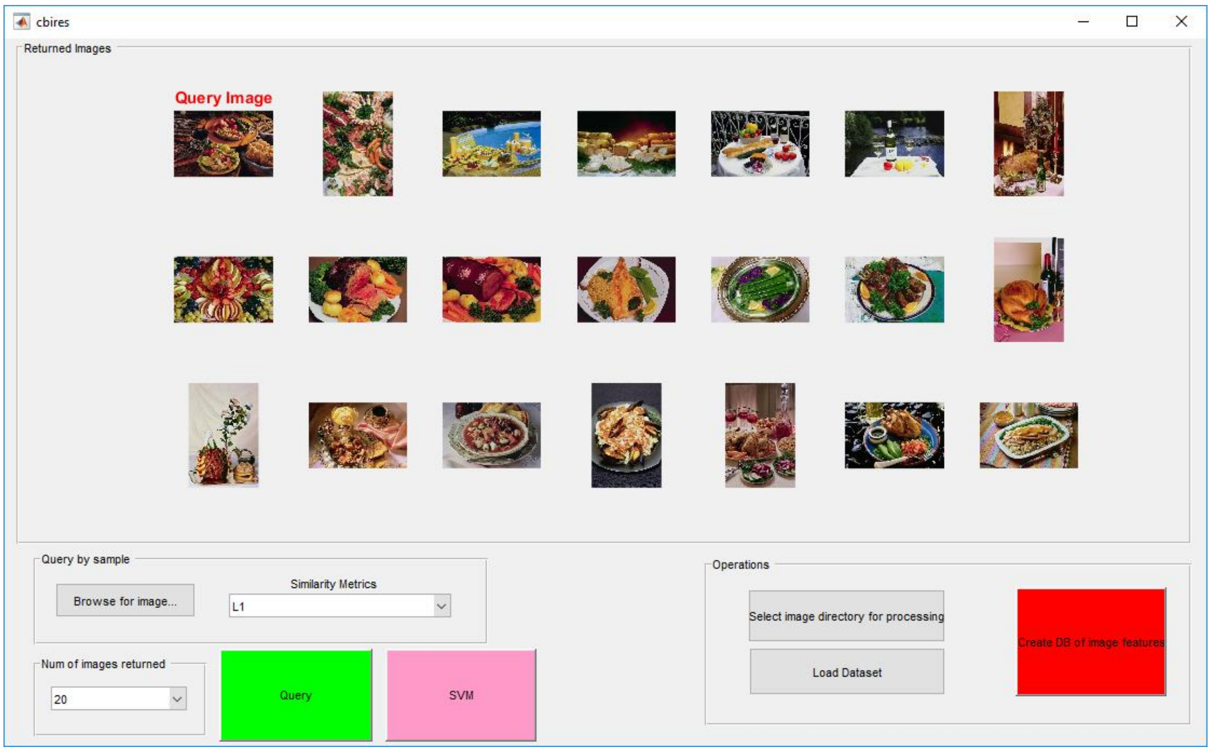


Figure A.20: Retrieved images of proposed system when query image is from the Food class.

REFERENCES

[1] Yan Zhou, Fan-Zhi Zeng, Hui-min Zha, Paul Murray, Jinchang Ren, ‘Hierarchical Visual Perception and Two-Dimensional Compressive Sensing for Effective Content-Based Color Image Retrieval.’ Springer, 2016.

[2] Wan J et al., ‘Deep learning for content-based image retrieval: a comprehensive study.’ In: Proceedings of the 22nd ACM international conference on multimedia, 2014; pp. 157–66.

[3] Wang H, Nie FP, Huang H et al., ‘Heterogeneous visual features fusion via sparse multimodal machine.’ CVPR, 2013; pp. 3097–102.

[4] Zheng L, Wang S, Zhou W et al., ‘Bayes merging of multiple vocabularies for scalable image retrieval.’ CVPR, 2014; pp. 1963–70, in Columbus, Ohio, USA, June 2014.

[5] Zheng L, Wang S, Liu Z et al., ‘Packing and padding: coupled multi-index for accurate image retrieval.’ CVPR, 2014; pp. 1947–54, in Columbus, Ohio, USA, June 2014.

[6] Sadegh Fadaei, Rassoul Amirfattahi, Mohammad Reza Ahmadzadeh, ‘A New Content-Based Image Retrieval System Based on Optimized Integration of DCD, Wavelet and Curvelet Features.’ IET Image Processing, 2016.

[7] Ray-I Chang, Shu-Yu Lin, Jan-Ming Ho, Chi-Wen Fann and Yu-Chun Wang, ‘A Novel Content Based Image Retrieval System using K-means/KNN with Feature Extraction.’ ComSIS Vol. 9, No. 4, Special Issue, December 2012.

[8] Priyanka Sharma, ‘Content Based Image Retrieval Using SVM.’ International Journal of Computer Science Trends and Technology (IJCST), Volume 4 Issue 4, Jul - Aug 2016.

[9] Sumiti Bansal, Er.Rishamjot Kaur, ‘Content Based Image Retrieval Using C-SVM Technique.’ International Journal of scientific research and management (IJSRM), volume 3 issue 4, April 2015.