CCBIR: A Cloud Based Implementation of Content Based Image Retrieval

R. MADANA MOHANA[1], Dr. A. RAMA MOHAN REDDY[2]

[1] Research Scholar, Department of Computer Science & Engineering, Sri Venkateswara University College of Engineering, Sri Venkateswara University, Tirupathi - 517 502, Andhra Pradesh, India.
rmmnaidu@gmail.com

[2] Professor, Department of Computer Science & Engineering, Sri Venkateswara University College of Engineering, Sri Venkateswara University, Tirupathi - 517 502, Andhra Pradesh, India.
ramamohansvu@yahoo.com

Abstract: - Content Based Image Retrieval (CBIR) is an efficient retrieval of relevant images from large databases based on features extracted from the image. This paper proposes a system that can be used for retrieving images related to a query image from a large set of distinct images. It follows an image segmentation based approach to extract the different features present in an image. The above features which can be stored in vectors called feature vectors and therefore these are compared to the feature vectors of query image and the image information is sorted in decreasing order of similarity. The processing of the same is done on cloud. The CBIR system is an application built on Windows Azure platform. It is a parallel processing problem where a large set of images have to be operated upon to rank them based on a similarity to a provided query image by the user. Numerous instances of the algorithm run on the virtual machines provided in the Microsoft data centers, which run Windows Azure. Windows Azure is the operating system for the cloud by Microsoft Incorporation.

Key-Words: - Content Based Image Retrieval (CBIR), Wavelet Ttransformation, Euclidean Algorithm, Windows Azure, Cloud Computing.

1 Introduction

In various areas of government, academia, hospitals and commerce large collections of digital images are produced. Many of these collections are the merchandise of digitizing existing collections of analogue drawings, photographs, paintings and prints. Generally the only way of searching these collections was by basically browsing or keyword indexing. Digital images databases however, open the way to content-based searching. [1]

Content Based Image Retrieval (CBIR) is concerned with the retrieval of images similar to a specified image, from an image repository. This paper aims at implementing a CBIR system in a cloud environment, to enhance the processing speed [22].

"Content Based" means that the search will analyze the actual contents of the image. The term "content" in this perspective might refer to shapes, textures, colors, or any extra information that can be resulting from the image itself. Without the ability to examine image content, searches have to rely on metadata such as keywords or captions, which may be difficult or expensive to produce. "Cloud Computing" is internet based computing, whereby shared resources, software and knowledge are provided to computers and different devices on demand.

Cloud Computing could be computing ability that gives an abstraction between the computing source and its fundamental technical design, enabling convenient, on demand network access to a mutual group of configurable computing assets that can be speedily provisioned and discharged with marginal management effort or service supplier interaction.

The organization of paper is follows: Section2 describes the overview of development of the system, section3 overall description, section4 related work, section5 implementation details, section6 experimental results and section7 conclusion and future work followed by references.

2 Overview of Development of System

The set of techniques used in developing this image retrieval system are purely mathematical. A mathematical tool called *Discrete Wavelet Transform (DWT)* has been used that provides an easy and effective way of extracting the variations in pattern present in an image [23]. Thus, the different features present in an image such as colour, pattern, shape and texture have

been extracted by using the above tool and have been represented by vectors, called feature vectors. The computer is then made to distinguish different regions present in an image with the help of *Statistical Clustering* of these feature vectors by using *K-Means Clustering Algorithm.*

These clusters are collection of feature vectors and within a cluster these feature vectors are almost similar to each other because a cluster represents one region of the image having similar content. The distance between the mean feature vectors of each of these clusters of one image to that of query image is calculated by simple Euclidean Distance formula for distance between two vectors [24]. The weighted sum of these distances according to the significance of the pair of clusters considered (all combinations from one image to the query image) is used as a measure of closeness or similarity between two images.

This paper implements the Euclidean Distance formula and is implemented on Cloud.

3 Overall Description

The CBIR system is an application built on Windows Azure platform. It is a parallel processing problem where a large set of images have to be operated upon to rank them based on a similarity to a provided query image by the user. Numerous instances of the algorithm run on the virtual machine provided in the Microsoft datacenters.

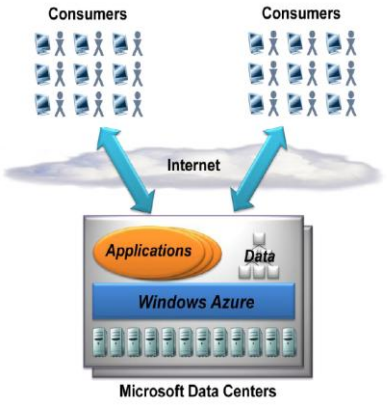


Fig. 1: Overall system design

Windows Azure could be a foundation for running Windows applications and storing data within the cloud. Windows Azure runs on machines in Microsoft data centers. Instead of providing software system that Microsoft customers will install and run themselves on their own computers. Windows Azure could be a service, customers use it to run applications and store information on Internet-accessible machines owned by Microsoft. Windows Azure is so a cloud services operating system that is the service management environment, improvement and service hosting for the Windows Azure Platform. Windows Azure provides you on-demand compute and storage to host, manages and scale web applications and services on the internet in Microsoft data centers. Those applications would possibly provide services to businesses, to customers, or both. The platform provides three basic services that integrate together to run the application on cloud. The Overall systems design is shown in the above Figure 1.

Compute Service:

The Windows Azure Compute service will run many various forms of applications. A primary goal of this platform is to sustain applications that have a very huge number of concurrent users. Windows Azure is meant to support applications that scale out, running multiple copies of an equivalent code across several commodity servers.

Two completely different instance types are available for developers to use: Worker role instances and Web role instances. Web role instance can accept incoming Hyper Text Transfer Protocol (HTTP) or HTTPS requests. To allow this, it runs in a VM that includes Internet Information Services (IIS) 7. Developers can create Web role instances using ASP.NET, Windows Communication Foundation (WCF), or another .NET technology that works with IIS. Worker role instances don't have IIS configured but they're executables in their own right. A developer will use only Web role instances, only Worker role instances, or a grouping of the two to create a Windows Azure application. [2], [3]. The computing service of azure is shown in Figure 2.

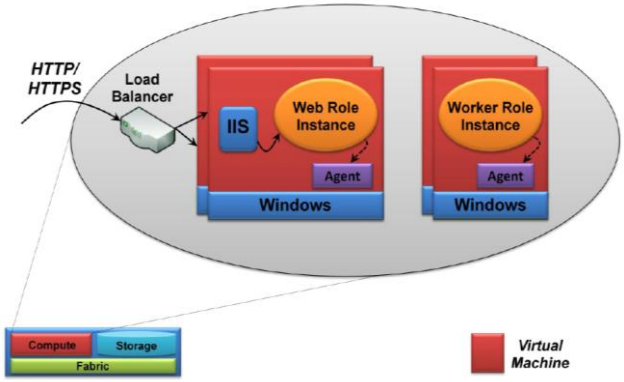


Fig. 2: Compute Service of Azure

Storage service:

Windows Azure Storage provides blobs, tables, and queues.

Blobs-A blob contains binary information, and there is an easy hierarchy: A storage account can have one or

several containers, every one of that holds one or plenty of blobs.
Tables-It permits operating with applications data in a more fine-grained way. The data every table holds is stored in a group of entities that contain properties. *Queues*-A primary function of queues is to afford some way for Web role instances to converse asynchronously with Worker role instances. [2] & [3]. The storages services provided by azure are shown in Figure 3.

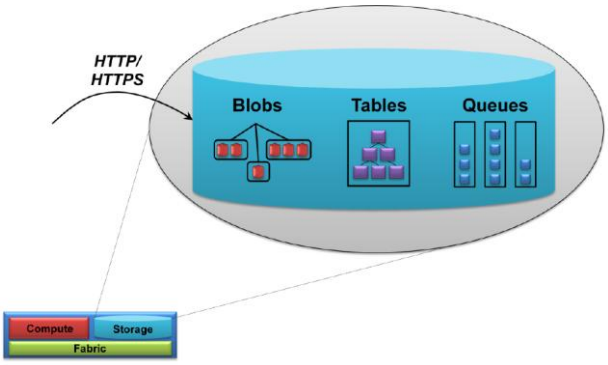


Fig. 3: Storage service provided by Azure

Fabric:

Windows Azure Fabric consists of a collection of machines, all of that are managed by software referred to as the fabric controller. The fabric controller is pretending transversely a cluster of five to seven machines, and it owns all of the possessions within the fabric. The fabric monitors all running applications. The fabric controller depends on a pattern file that is uploaded with all Windows Azure application. Every Windows Azure applications and all of the information in Windows Azure Storage are present in some Microsoft data center. In that data center, the set of machines devoted to Windows Azure is structured into a fabric. [2], [3]. The fabric service provided by azure is given in Figure 4.

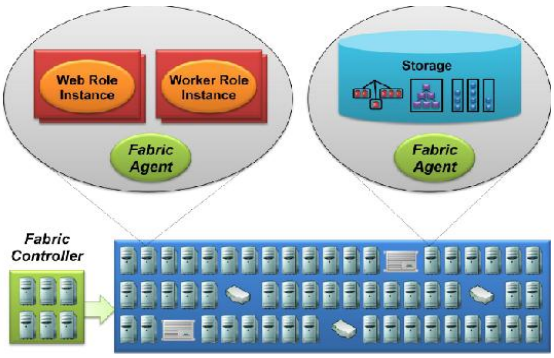


Fig. 4: Fabric Service provided by Azure

4 Related Work

A considerable number of image retrieval systems have been developed for commercial use and demonstration versions are in existence in the academic world. The commercial systems include IBM's QBIC, VIR image engine search from Virage Inc and Excalibur from Excalibur technologies. Several experimental systems exist in the academia, notable among them are Photo book system from Massachusetts institute of technology (MIT), Visual SEEK system of Columbia University and MARS of University of Illinois and NETRA of University of California, Santa Barbara. They are NETRA, RETIN (Recherche et Traque Interactive) developed by ENSA/University of cergy Pointoise, France, KIWI (Key-points Indexing Web Interface) of INSA Lyon, France, iPURE (Perceptual and User-friendly Retrieval of Images) developed by IBM India Research lab, New Delhi, India, and ImageMiner developed by Technologie-Zentrum Informatik, University of Bremen, Germany. Yang Di et al. [25] & [26] proposed the scalable image retrieval framework which can efficiently support content similarity search in the distributed environment. Its key idea is to integrate image fusion features into distributed hash tables (DHTs) by exploiting the property of the locality sensitive hashing (LSH). Choudhary R et al. [24] proposed a content based image retrieval integrated technique which extracts both the color and texture feature. To extract the color feature, color moment (CM) is used on color images and to extract the texture feature, local binary pattern (LBP) is performed on the grayscale image. Belloulata K et al. [23] proposed a new Region-Based Image Retrieval approach using a Shape Adaptive Discrete Cosine Transform (SA-DCT). In this retrieval system, an image has a prior segmentation alpha plane, which is defined exactly as in MPEG-4. Karande S. J and Maral V [22] proposed to use cloud computing as distributed computing environment to overcome the challenge bottleneck, the most of the recent research work in CBIR is focused on reduction of semantic gap.

NETRA

NETRA CBIR system use color, texture, shape and spatial location [13]. The descriptor for the color is color histogram obtained using training set of images. Texture feature descriptor is the normalized mean and standard deviation of the Gabor wavelet transform of the images. Shape feature descriptors are the curvature function of the contour, the centroid distance function of the contour and the complex coordinate function of the contour. Spatial location descriptor is bounding box [14]. The system allows query by example. Similarity matching is carried out in the Euclidean space [24].

Querying: There are 2,500 images from the Corel photo collection, organized in 25 categories, with 100 images in each category. You can select any one of them as the query image. All images in the database have been segmented into homogeneous regions [15]. You can click on one of the regions and select one of the four image attribute color, spatial location, texture, and shape. Instead of using an image example, you can also directly specify the color and spatial location. The spatial location querying tool utilizes two bounding boxes to define the area of interest. The inner box is used to define the preferred area, and the box outside is used to constrain the objects to be within this area. Thus, if the object has any its bodies exceeding this outside box, they will not be considered.

Result presentation: The matched images are linearly ordered

Applications: An initial prototype of NETRA is used in ADL (Alexandria Digital Library) to search on texture. [4], [5], [17].

RETIN

RETIN CBIR system use color and texture. The system unlike the NETRA system does not process image as a single entity. Rather for each image random pixels are selected of which color and texture feature are computed. The descriptor for color is color histogram while the texture descriptor is the output of Gabor transformation of the images. The system allow query by example. However the system is flexible enough to allow the user to locate region of interest within a query image and use it for search.

Similarity matching is carried out using weighted Minkowski distance and cross matching between bins.

Querying: The user give a query image. Searching is either done on the whole image, or on a region of interest that is drawn by the user. From the whole image or the region of interest, the distribution of feature classes is computed, and compared with those of the database images.

Matching: Two distances between two histogram vectors H_1, H_2 of length k are used. One is the weighted Minkowski distance of order p, the other allows cross matching between bins.

Result presentation: The retrieved images are shown in linear order with respect to the matching score.

Relevance feedback: The weights between the feature distances u_i and within the distances w_i and w_{ij}, are set uniformly to start with. The user indicates which of the retrieved images are relevant, and which are not. The system then adjusts the weights using a least mean square error rule. [6]

KIWI

KIWI system like the RETIN system detects key points in an image rather than the entire image. It uses color and shape. The color descriptor is color histogram while the shape descriptors are computed with the aid of Gabor filters. The system allows Query By Example (QBE). Similarity matching is carried out in the Euclidean space.

Querying: The user can specify a query image by selecting an image from the database, or by providing an URL of an image.

Matching: The color feature vectors are compared with the Euclidean distance. After normalization, the 2D histograms [11] are compared using the Bhattacharyya distance given in equation (1):

$$d(H_1, H_2) = -\log \sum_{i,j} \sqrt{H_1(i,j) H_2(i,j)} \qquad (1)$$

Where H_1 and H_2 are two histograms.

After a normalization of the distribution of distance values from the individual features, the similarity values are sorted and then weighted averaged.

Result presentation: The retrieved images are presented in linear order with respect to the final score. [7], [8].

iPURE

iPURE system operates using color, texture, shape and spatial location. The system segments the image into regions before deriving the features. Color descriptor is the average color in CIE's (Commission internationale de l'éclairage) Luv color space. Texture descriptors are by means of word decomposition [18]. Shape descriptors are size, orientation axes and Fourier descriptor. The spatial location descriptors are centroid, and bounding box. The system accept query by example. Similarity matching s carried out in the Euclidean space.

Querying: The user starts the retrieval session with a query image which is sent to the server, which sents back a segmented image. The user can select one or more segments.

Matching: Individual segments are matched by computing a weighted Euclidean distance on the feature vectors. Then the contiguity amongst the retrieved segments are matched against those of the query segments.

Relevance feedback: The user can give relevance feedback modified versions of the retrieved images. The modifications are change of color (bit flipping in RGB domain), texture (Wold frequencies), shape (changes of

size and the highest Fourier coefficients), and spatial layout (position and orientation). If the user finds the modified images also relevant, it means that the particular feature has low relevance. The new query is defined as the mean of the feature vectors of the relevant images [9].

ImageMiner

ImageMiner system use color, texture and shape. Color descriptors is color histogram, texture description is grey level co-occurrence matrix. Shape description is carried out using image contour size, centroids and boundary coordinates. The system feature description has the capability to classify scenes and generate description of the scene with keywords and values. The system allows query by example. Special module within the system carries out similarity matching.

Querying: SQL-like queries with keywords and values can be posed.

Matching and Indexing: Retrieval is performed by the IBM Search Manager Text searcher.

Applications: Videos are made available for retrieval with ImageMiner in two steps. First, shots are extracted from the video by using a histogram based method. More precisely, a new shot is detected when the difference in the intensity histogram of two consecutive frames exceeds a given threshold. Then, a mosaicing-technique is used to generate a single still image for each shot detected in the video sequence. [10], [17].

REVIEW

CBIR system effectiveness is measured by its precision and recall [13]. Precision is the ratio of relevant images to the total number of images retrieved. Recall is percentage of relevant images among all possible relevant images. Though manufacturers of CBIR systems give good figures about its precision and recall, it is difficult to evaluate how successful content-based image retrieval systems are in terms of effectiveness, efficiency and flexibility. For example if a database is narrow with only airplanes the system will only return airplanes as similar images even if the query image is something other than airplane. Also if the database is diverse but contains only a single chicken, the best that can be achieved is return of only one chicken, in response to a chicken query. Thus the more diverse and larger the collection of images in the database the more chance that it contains images similar to the query image. This review points to the fundamental issues in the design of CBIR including its image database, which are

- Efficient selection of images which satisfy a user's query.
- Data modeling.
- Feature extraction.
- Selecting appropriate features for content representation.
- Query languages, and Indexing techniques.

5 Implementation Details

For CBIR, only the foremost segments are measured for feature extraction namely texture features, color histogram features and image density features. Then a single feature vector is constructed and stored in the feature database. When a query image is submitted by the user, the similar task is done as represented above to get its feature vector. For similarity comparison between the database image and the query image, the Euclidean Distance technique is used. Using a proper threshold, images that are semantically closer are extracted from the database and displayed as a thumbnail.

CBIR Architecture of our system is shown in Figure 5.

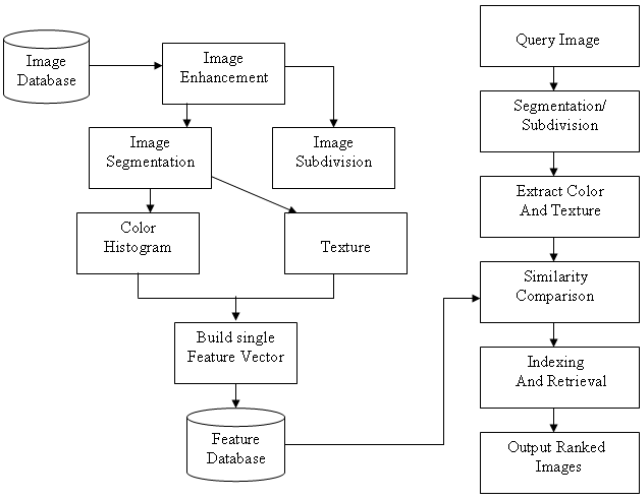


Fig. 5: Architecture for CBIR implementation

In image retrieval systems color histogram is the most commonly used feature. The main reason is that it is independent of image size and orientation. Also color histogram is one of the most straight forward features utilized by humans for visual recognition and discrimination. Statistically, it denotes the common probability of the intensities of the 3 color channels. Once the image is segmented, from every region the color histogram is extracted. The major statistical data that are extracted from the color histogram are the number pixels having similar intensity corresponding to the three basic colors (that are Red, Blue and Green)and this is done on a per-segment basis for all the images

(query and repository images alike) and features per segment are obtained. All the segments need not be considered, but only segments that are dominant may be considered, because this would speed up the calculation and may not significantly affect the end result.

Local Colour Histograms (LCH):

One feature of colour histograms that can be both an advantage and a disadvantage is their lack of spatial information [12]. This can be an advantage as a given image's global histogram will remain the same when rotated or flipped. This does however also mean that two perceptually very different images with similar colour distribution will be deemed similar by a colour histogram-based retrieval system as illustrated in Figure 6. To alleviate this problem several methods of introducing some spatial information have been suggested.

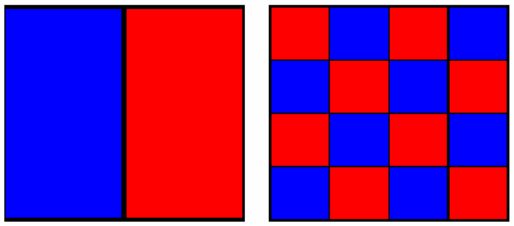


Fig. 6: Two perceptually different images with equal colour distribution

The simplest and most obvious way of introducing such spatial information is the method used by Gong and others [16]. Furthermore they divided the images into nine equal parts and considered a histogram for each of these. This gives some spatial sensitivity, but increases the computing power and storage needed. One also looses the insensitivity to rotation we have in global colour histograms. Shengjiu Wang [19] proposes a rotation-insensitive variant of *Local Color Histograms* (LCH) Shengjiu Wang calls the Harbin approach in his publication *A Robust CBIR method Using LCH*. This approach furthermore divides the images into a number of evenly sized regions and computes their LCH. The difference from Gongs approach lies in the method for comparing the images. His Harbin approach uses a system of weighted bipartite graphs to calculate the minimum cost distance between two images. With this method each part of one mage is compared to all the parts of the comparison image. This makes the method less sensitive to rotation compared to Gong and others' method described above.

An approach to LCH not using equally sized regions of the image was proposed by Lu, Phillips and Harman [20]. They propose a system where two histograms are created, one for the foreground and another for the background. This was meant to alleviate the problems of background masking. Background masking occurs when an image for instance contains a small car on green lawn. The green lawn would take up more of the image area, and green would be weighted heavily in global colour histograms. This could be a disadvantage if what the user wanted to find was more images containing the small car in the foreground. Defined as the minimum rectangle containing the most important objects of an image, the foreground would be found automatically by determining vertical and horizontal pixel value transitions or set manually by the user.

Minimum Probability of Error Retrieval (MPER):

MPER system is developed by SVCL's 2007 MPE systems [21]. In this approach system decomposes images into subsystems (also known as bags of local features) that measure properties such as texture, color, shape etc. Then the Gaussian Mixture Model (GMM) is developed from each bag. Each bag will correspond to a pixel in the GMM. For each database, subsystems will form GMM and stores it as the signature along with each image. When the query image is given, system forms its GMM, checks it against the GMM's in the database and retrieves the images having the maximum similarity.

Supervised Multiclass Labeling (SML) Algorithm:

SML Algorithms are a set of simple, yet powerful algorithms. The annotation of the semantic retrieval system includes set of operations as:

i. Train the system with a set of image databases.
ii. Find the probability of each part of subimage (subpart) in a full image. The system itself automatically annotates the image according to the probability of the content of the image.

Fig. 7: Cat, Tiger, Plants, Leaf, Grass

The algorithm is known as Supervised because user trains the image labelling system to identify classes of objects and it allows the system to differentiate between similar concepts. For example SML system can identify a polar bear and gizzly bear as bears. It is known as multiclass because training process can be repeated for many visual concepts. I.e. same system can be used for

the identification of different concrete objects as Tiger, sky, man, etc shown in Figure 7. The entire algorithm is known as SML as it is the method of connecting particular features contained by images straightforwardly to words that illustrate these features.

The execution of SML algorithm is done through different steps as:

- System splits each image into 8x8 pixel squares.
- It extracts some information from these squares called as logical feature. This localized features for an image is collectively known as a "bag of features".
- It finds the probability of each feature as a vector of features by using some equations and gives the names of annotations in that order.
- At annotation time, all trained classes directly compete for the image.
- One way to test SML system is to ask it to annotate images in the database and then retrieve images based on test queries.

In this method automated segmentation is the harder part in the case of computer vision. Advantages of Semantic Retrieval over Query by Visual Example are: it gives the higher level of abstraction, supports natural language queries and avoids semantic gaps. But performance of semantic retrieval system is confined to particular semantic classes, can not do retrieval or matching by standing outside the semantic domain and it is not possible when limited by vocabulary.

5.1 CBIR Architectural Design:

The architectural design of CBIR given in Figure 8 and description of the components are given below:

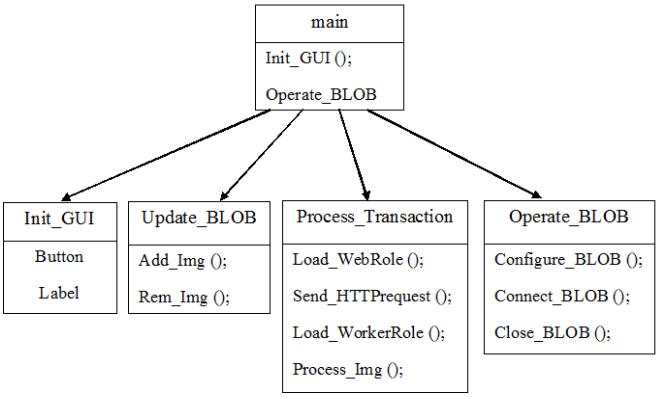


Fig. 8: Data flow diagram

Init_GUI:
Type: Function
Description: This component initializes the graphical user interface required for the user to perform his/her transactions .This is also an initialization process at the front end of the system. This is automatically activated when the application is started by the user.
Flow of events:
1) The user starts the system.
2) Main function calls the GUI component functions.
3) The functions initialize GUI by providing necessary buttons and labels.

Update_BLOB:
Type: Function
Description: This component helps to update the BLOB to its present values. It contains "Add Images" button and on click of this button, the BLOB data structure provided by Windows Azure is updated.
Flow of events:
1) The user clicks the button to update the BLOB data structure.
2) The updation of the BLOB data structure is tried.
3) The success/failure of the updation is reported back.

Process Transaction:
Type: Function
Description: The Web Role instance accepts HTTP requests from the user which are scheduled into a queue by it. The Worker Role instances perform the computations by processing these requests. It uses the BLOB data structure provided by Windows Azure to get the data.
Flow of Events:
1) The Web Role instance accepts HTTP requests from the user.
2) It then schedules these requests into a queue.
3) The Worker Role instances perform the computations by using the data stored in the BLOB data structure.

Operate_BLOB:
Type: Function
Description: The configuration file is set to indicate the URL of the storage account and the data to be accessed. A connection is established to the BLOB data structure provided by Windows Azure. After the interaction is complete the connection is closed.

5.2 CBIR Implementation:

For CBIR, simply the foremost segments are measured for feature extraction i.e. texture features, color histogram features and image density features. Then a single feature vector is constructed and stored in the feature database. When a query image is submitted by the client, the similar exertion is made as described over to get its feature vector. For similarity evaluation among the database image and the query image, the Euclidean Distance method is used. By means of a

suitable threshold, images that are semantically nearer are retrieved from the database and displayed as a thumbnail. The system use case realization is shown in Figure 9 and the CCBIR procedure is given below.

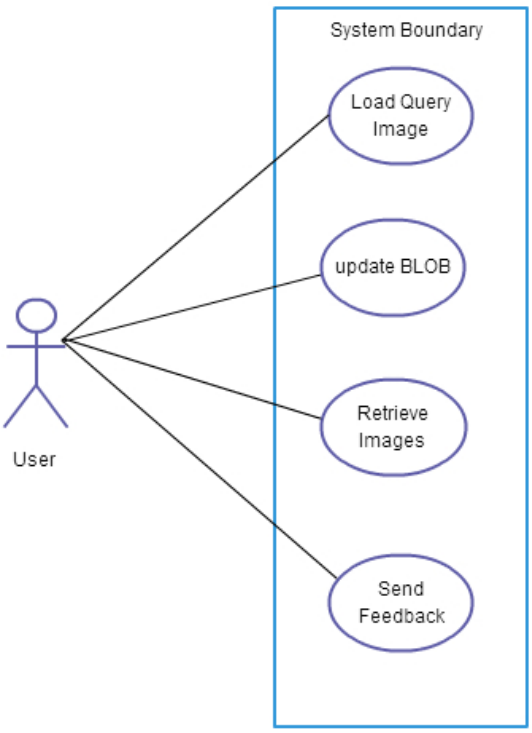


Fig. 9: System Use Case realization

Procedure for CCBIR Implementation:

1. Send the query image through the web role handling the application, for computation, to a worker node.
2. Worker node sends back the structure containing the image features of the query image.
3. Send the request for computations on the images stored in the BLOBs to worker nodes, with query image features as parameter.
4. Worker nodes send back a structure containing the image name and the similarity index for each image on which they operate (making use of the procedure to compute image features for these images).
5. Rank images based on the similarity index and create thumbnails for the images and display

Procedure to compute features of an image:

SEGMENTATION

1. Segment the image such that it becomes a 6*6 matrix.
2. Compare each segment of the query image with the images in the repository until a match is found.
3. The matched segment is flagged to avoid future computations with it.
4. This is repeated for all the segments of the image and the image with >30 segments, is tagged as one of the similar images.

The Algorithms for Segmentation and Segment comparison are shown in Table1 and Table2 respectively:

Table 1
ALGORITHM FOR SEGMENTATION

Algorithm: Segmentation

Get the Boundary of the image along Y-axis
Get the Boundary of the image along X-axis
//Segment the bitmap image into a 6*6 matrix
for every boundary of the divided image along Y-axis
for every boundary of the divided image along X-axis
Get the segment boundary
Copy segment
end for
end for
return Segmented_Images;

Table 2
ALGORITHM FOR SEGMENT COMPARISON

Algorithm: Segment Comparison

Initialize a flag array of size 36
for every segment of query image
for every segment from repository
if(the flag is set AND the segments are similar)
Increment SimilarityCount;
end if
end for
end for
if(SimilarityCount >30)
Tag image to be similar
end if

FEATURE COMPUTATION

1. For each region compute histogram on the segments and select features of those segments that are most frequently appearing inside that region (this is done by defining a threshold to indicate similarity).
2. Extract the texture features for the selected fragments in the last step.

3. Include the information in a structure that represents the particular region.
4. Combine information from all regions into a structure to represent the features for the entire image.

The Algorithm for Feature Computation is shown in Table 3:

Table 3
ALGORITHM FOR FEATURE COMPUTATION

Algorithm: Feature Computation
for all the pixels in the image Get a pointer to each pixel of the image Store the RED, GREEN, BLUE in an array **end for** Sort the RED, GREEN, BLUE array in ascending order **for** first 64 values of each array Feature= Feature+ Number of pixels/(1+Intensity of pixel); Feature= Feature/ 64 **end for** **return** Feature;

FEATURE COMPARISON

1. The features are often treated as vectors, so the difference turns to be the distance between these two vectors. It's naturally to define the distance in terms of Euclidean norms.
2. Given two images, the difference between the two features measures the similarity of these two images. The features are often treated as vectors, so the difference turns to be the distance between these two vectors. It's naturally to define the distance in terms of Euclidean norms. But the absolute distance is not quite suit for this task. For example, we have two pairs of images (a, a'), (b, b'). The feature of these images: f (a) = 1000, f (a') =1050, f (b) =100, f (b') =150. The absolute distances are same for both cases, but it's obviously that the difference in the second case is more significant.
3. So we use the relative measure of distance given in equation (2):

$$d_{(r,s)} = \frac{|r-s|}{1+r+s} \tag{2}$$

Where r and s are the features to be compared and d is the distance between them. The distance is used as the similarity index that is used for ranking the images (lesser the distance more similar are the images being compared).

The Algorithm for Feature Comparison is shown in Table 4:

Table 4
ALGORITHM FOR FEATURE COMPARISON

Algorithm : **Feature Comparison**
Get the Feature of Query image
Get the Feature of Image from repository
Calculate the difference between the above features

6 Experimental Results

Database:

The dataset consisting of the classes mentioned in Table1 and some assorted images. This is considered as our test environment on cloud. The training was on less than half the testing data. The Table5 shows the general performance of the system on image database where images are classified by humans roughly based on objects present in them. It can be observed that the performance of the system depends widely on the kind and variety of objects in the image. Totally, we have considered one thousand assorted images, randomly taken from the web. So, from the image database of one thousand images, we have calculated precision and recall for varied image queries.

The recall is defined as the proportion of relevant documents retrieved and precision is defined as the proportion of retrieved documents that is relevant. The accuracy is defined as the difference between the proportion of retrieved documents that are relevant and the proportion of retrieved documents that are irrelevant. The recall, precision and the accuracy for the database is shown in Table5.

The common evaluation measures used in CBIR systems are precision and recall defined as equation (3) & (4) respectively.

$$\text{Precision}(P) = \frac{r}{n} \tag{3}$$

$$\text{Recall}(R) = \frac{r}{R} \tag{4}$$

Where: r: number of relevant documents retrieved
n: number of documents retrieved
R: total number of relevant documents.

The average precision is calculated as the ratio of difference between the number of relevant pictures retrieved (r_i) and the number of irrelevant pictures (ir_i) to the total number of relevant pictures in the database (tr). The average is calculated over query of all the images in the class and is given in equation (5).

$$\text{Average Precision} = \frac{(r_i - ir_i)}{tr} \quad (5)$$

The accuracy is calculated by sum of precision and recall and then divided by two and is given in equation (6).

$$\text{Accuracy} = \frac{(\text{Precision} + \text{Recall})}{2} \quad (6)$$

Canberra and Euclidean distance given as equation (7) & (8), both measures were used for finding similar images. Euclidean distance is used between the standard deviations of the query image and the images in the database.

$$\text{Euclidean Distance} = \sqrt{\sum (a_i - b_i)^2} \quad (7)$$

$$\text{Canberra Distance} = \sum \frac{|a_i - b_i|}{|a_i| + |b_i|} \quad (8)$$

The performance of CCBIR system with respect to average precision, precision, recall and accuracy using Euclidean Distance (Method1) is given in Figure 10.

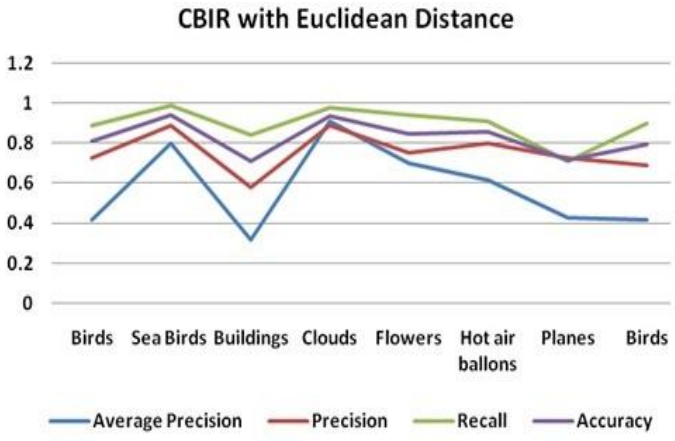


Fig. 10: Performance of the CBIR system using Euclidean Distance (Method1)

The performance of normal CBIR system using for Each Category of Images for Canberra Distance with respect to average precision, precision, recall and accuracy is given in table 5.

Table 5
PERCENTAGE OF PERFORMANCE OF THE CBIR SYSTEM USING CANBERRA DISTANCE (METHOD2)

S. No.	Category	Average Precision (%)	Precision (%)	Recall (%)	Accuracy (%)
1.	Birds	0.468	0.734	0.93	0.601
2.	Sea Birds	0.88	0.94	1	0.91
3.	Buildings	0.36	0.68	0.86	0.52
4.	Clouds	0.94	0.97	1	0.955
5.	Flowers	0.716	0.858	0.97	0.787
6.	Hot air ballons	0.63	0.815	1	0.7225
7.	Planes	0.467	0.7335	0.75	0.60025
8.	Birds	0.342	0.671	0.71	0.5065

The average performance of accuracy of CBIR with Euclidean Distance and Canberra Distance is given in Figure 11.

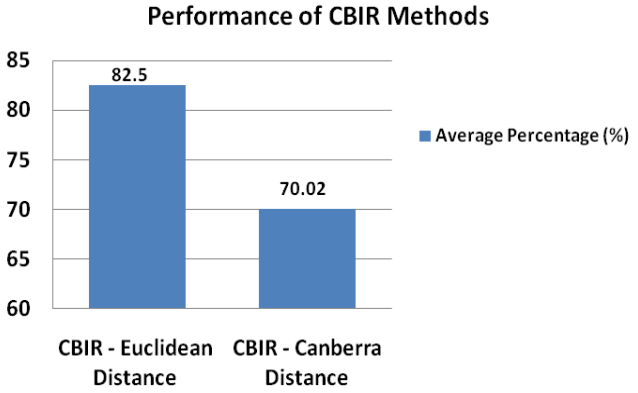


Fig. 11 Average Percentage of Accuracy of all CBIR methods

Performance:

Since the proposed approach is implemented on the cloud platform there is a considerable increase in the speed of computation. Azure provides a set of Worker roles that can perform parallel computations thus decreasing the overall processing speed. And also the following non-functional features are provided.

Fault tolerance

To an application developer, Windows Azure consists of the Compute service and the Storage service. However neither one might function exclusive of the Windows Azure Fabric. By means of disapproval simultaneously a data center filled of machines into an

articulate whole, the Fabric provides a basis for the whole thing else.

The fabric controller owns all resources in a particular Windows Azure data center. It's also conscientious for handing over instances of both applications and storage to physical machines. Doing this intelligently is important. A developer requests for a number of Web role instances and Worker role instances for his/her application. A naive assignment might situate every one of these instances on machines in the same rack serviced by the identical network switch. If either the rack or the switch failed, the entire application would no longer be available. Known the high accessibility goals of Windows Azure, building an application dependent on single points of failure like these would not end up achieving high accessibility. To avoid this, the fabric controller groups the machines it owns into an amount of liability domains. Every fault domain is a part of the data center where a single failure can shut down access to everything in that domain.

The application portrayed in the figure above is running just two Web role instances, and the data center is divided into two fault domains. While the fabric controller deploys this application, it places one Web role instance in each of the fault domains. This arrangement means that a single hardware failure in the data center can't take down the entire application. The fabric controller sees Windows Azure Storage as just another application - the controller doesn't handle data replication. Instead, the Storage application does this itself, making sure those replicas of any blobs, tables, and queues used by this application are placed in different fault domains.

Scalability

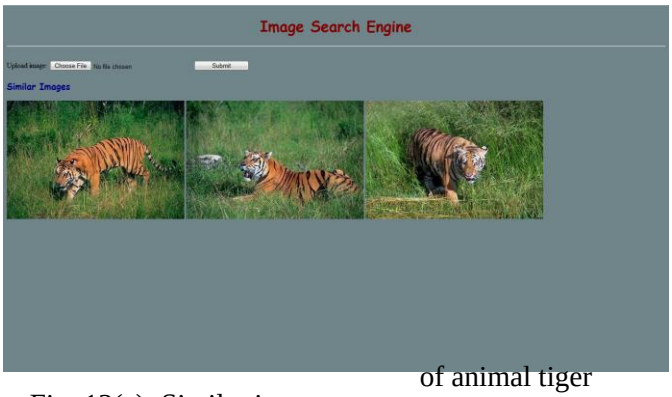
Scalability can be achieved in two ways: Computing scalability and Data storage scalability. Computing scalability is achieved by dynamically scaling out the number of the Worker role instances as and when the requirement arises. Data storage scalability is achieved by the usage of BLOBs for storing the images which are scalable by nature.

Security

To use Windows Azure Storage a user must build a storage account. This can be done via the Windows Azure portal internet interface. The user can receive a 256-bit secret key once the account is created. Further the above secret key is used to substantiate user requests to the storage system. Specifically, a HMAC SHA256 signature for the request is produced by using the above secret key. The signature is accepted among every client requirements by request to authenticate the verifying the HMAC signature.

Snapshots of CCBIR System:

Snapshot is that the state of a system at a selected purpose in time. Snapshot can refer to a genuine copy of the state of a system or to a potential provided by certain systems. These snapshots output a set of similar images to a query image specified. The following Figures 12(a) to 12(h) shows various results of different image cases of CCBIR system.



of animal tiger

Fig. 12(a): Similar image output

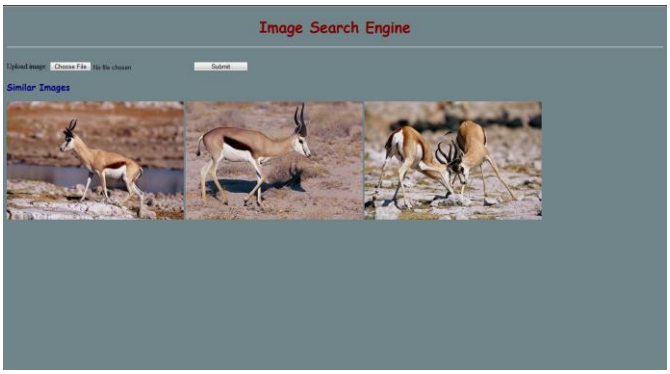


Fig. 12(b): Similar image output of animal deer

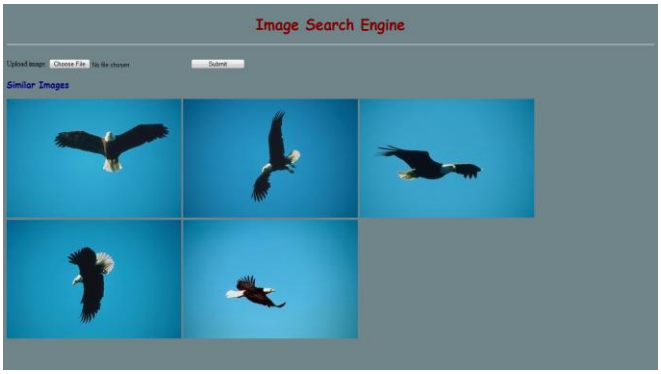


Fig. 12(c): Similar image output of birds

Fig. 12(d): Similar image output of flowers

Fig. 12(e): Similar image output of planes

Fig. 12(f): Similar image output of clouds

Fig. 12(g): Similar image output of combination of images

Fig. 12(h): Similar image output of combination of images

7 Conclusion

The need to discover a desired image from a group is shared by several professional teams, including design engineers, journalists and art historians. Whereas the necessities of image users will vary significantly, these are often useful to characterize image queries into three levels of abstraction: primitive features such as shape or color, logical features such as the individuality of objects given away and abstract attributes such as the importance of the scenes depicted. Whereas CBIR systems presently function efficiently simply at the lowest of these levels, the majority users require higher levels of retrieval.

Users need to retrieve images from a set come from a diversity of domains, including medicine, architecture, crime prevention, publishing and fashion. Extremely minute has yet been available on the approach such users seek for and use images, though makes attempts are being created to classify users' behavior within the hope that this can permit their requirements to be better met within the future.

Existing indexing carry out for images depends mostly on classification codes or text descriptors supported in several cases by text retrieval correspondence intended or made to order especially to handle images. Yet again, extremely tiny substantiation on the efficiency of such systems has been published. Client fulfillment with such systems appears to differ significantly.

CBIR operates on a completely different principle from keyword indexing. Primitive features characterize image content, such as color, textures, and shape, are computed for both stored and query images, and used to identify the twenty stored images most strongly corresponding the query. Semantic features comparable to the kind of object present within the image more durable to extract, although this remains an active research topic. Video retrieval may be a topic of increasing importance. CBIR techniques are moreover used to fragment extensive videos into entity shots, extract still key frames shortening the content of every shot, and explore video clips containing particular types of movement.

The effectiveness of all current CBIR systems is intrinsically restricted by the fact that they can operate only at the primitive feature level. None of them will search effectively for, say, a photograph of a dog, although some semantic queries may be handled by specifying them in terms of primitives. A beach scene, for instance, is retrieved by specifying huge areas of blue at the top of the image, and yellow at the bottom. There is substantiation that combining primitive image features with hyperlinks or text keywords will conquer variety of these troubles, however modest is known with relation to however such features will best be combined for retrieval.

References:

[1] Remco C. Veltkamp, Mirela Tanase(2002). "Content-Based Image Retrieval Systems: A Survey", *UU-CS*-2000-34, 2002.

[2] "Windows Azure Platform Home Page", *http://www.microsoft.com/windowsazure*

[3] David Chappell. "Introducing the Windows Azure Platform". *http://go.microsoft.com/fwlink/?LinkId=158011*.

[4] W. Y. Ma (1997). "NETRA: A Toolbox for Navigating Large Image Databases". *Ph.D thesis, Dept. of Electrical and Computer Engineering, University of California* at Santa Barbara, June 1997.

[5] Wei-Ying Ma and B. S. Manjunath (1999). "Netra: A toolbox for navigating large image databases". *Multimedia Systems*, 7(3):184–198, 1999.

[6] J. Fournier, M. Cord, and S. Philipp-Foliguet (2001). "Retin: A content-based image indexing and retrieval system". *Pattern Analysis and Applications*, 4(2/3):153–173, 2001.

[7] E. Loupias and S. Bres (2001). "Key point-based indexing for pre-attentive similarities: The kiwi system". *Pattern Analysis and Applications*, 4(2/3):200-214, 2001.

[8] Etienne Loupias and Nicu Sebe (2000). "Wavelet-based salient points: Applications to image retrieval using color and texture features. In Advances in Visual Information Systems" . *Proceedings of the 4th International Conference, VISUAL 2000, Lecture Notes in Computer Science* 1929, pages 223–232. Springer, 2000.

[9] Gaurav Aggarwal, Pradeep Dubey, Sugata Ghosal, Ashutosh Kulshreshtha, and Abhinanda Sarkar (2000). "iPure: Perceptual and user-friendly retrieval of images". *In Proceedings of IEEE Conference on Multimedia and Exposition (ICME 2000)*, July 2000.

[10] J. Kreyss, M. R¨oper, P. Alshuth, Th. Hermes, and O. Herzog (19997). "Video retrieval by still image analysis with ImageMiner". *In Proceedings of IS&T/SPIE's Symposium on Electronic Imaging: Science & Technologie,* 8-14 Feb. '97, San Jose, CA, 1997.

[11] Sameer Antani, L. Rodney Long, George R. Thoma (2004). "Content-Based Image Retrieval for Large Biomedical Image Archives" *MEDINFO 2004*.

[12] S. Nandagopalan, Dr. B. S. Adiga, and N. Deepak (2008). "A Universal Model for Content-Based Image Retrieval". *World Academy of Science, Engineering and Technology*, Vol.2, 2008.

[13] Michael Eziashi Osadebey (2006). "Integrated Content Based Image Retrieval using texture, shape, spatial information", *Master Thesis Report in Media Signal Processing*, Department of Applied Physics and Electronics, Umea University, Umea, Sweden, February 2006.

[14] Chang, S K et al (1988). "An intelligent image database system". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(5), 681-688.

[15] Eakins J. P, Boardman J. M and Graham M. E (1998). "Similarity retrieval of trade mark images". *IEEE Multimedia*, 5(2), 53-63.

[16] N.Vasconcelos and A. Lippman (2000). "Feature Representations for Image Retrieval: Beyond the Color Histogram". *Proceedings of the International Conference on Multimedia and Expo*, New York, 2000.

[17] John P. Eakins and Margaret E. Graham (1999). "Content-based Image Retrieval". *A Report to the JISC Technology Applications Program*, University of Northumbria at Newcastle.

[18] Eakins J P, Graham M E and Boardman J M (1997). "Evaluation of a trademark retrieval system", *in 19th BCS IRSG Research Colloquium on Information Retrieval*, Robert Gordon University, Aberdeen.

[19] Shengjiu Wang (2001). "A Robust CBIR Approach Using Local Color Histograms*",TR 01-13*, Canada, October 2001.

[20] Lu, G. J. Phillips and S. Rahman (1998). "Techinques to Improve Color Based Image Retrieval Performance*". Proceedings of International Symposium on Audio, Video, Image Processing and Intelligent Applications*, Baden Baden, Germany. pp 57-61. August 17-21, 1998.

[21] Nuno Vasconcelos (2007). "From Pixels to Semantic Spaces: Advances in Content-Based Image Retrieval", *IEEE Computer*, 2007.

[22] Karande S.J, Maral V (2013). "Semantic content based image retrieval technique using cloud computing", *Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on*, 26-28 Dec. 2013, Enathi.

[23] Belloulata K, Belallouche L, Belalia A, Kpalma K (2014). "Region based image retrieval using Shape-Adaptive DCT", *Signal and Information Processing (ChinaSIP), 2014 IEEE China Summit*

& International Conference on, 9-13 July 2014, Xi'an.

[24] Choudhary R, Raina N, Chaudhary N, Chauhan R, Goudar R.H (2014). "An integrated approach to Content Based Image Retrieval", *Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on*, 24-27 Sept. 2014, New Delhi.

[25] Yang Di, Liao, Jianxin, Wang, Jingyu Qi, Qi Sun, Haifeng (2014). "Multiple features for image retrieval in distributed datacenter", *Network Operations and Management Symposium (APNOMS), 2014 16th Asia-Pacific*, 17-19 Sept. 2014, Hsinchu, Taiwan.

[26] Di Yang, Jianxin Liao, Qi Qi, Jingyu Wang, Tonghong Li (2014). "An image retrieval framework for distributed datacenters", *Local Computer Networks (LCN), 2014 IEEE 39th Conference on*, 8-11 Sept. 2014, Edmonton, AB.