Hybrid Semantic Model for Content based Image Retrieval

Mr. Lalit Gehlod¹ Aman Chouhan²
¹²Department of Computer Engineering

1,2Institute of Engineering & Technology, Devi Ahilya University, Indore, India

Abstract— The challenging topic in image processing is extracting accurate images from the data base according to the user defined query and objects available in image database. Therefore in this paper we described different technique for content based image retrieval and their functioning advantages. In addition of that it includes the investigation about the different recent approaches that are developed recently. We go through many technologies and made some comparisons between them. Finally we proposed a methodology that is accurate for image retrieval as well as efficient as possible.

Key words: Hybrid Semantic Model, Content based Image Retrieval

I. INTRODUCTION

Internet serves data to all the users of internet. Due to the communication aspects of the internet the use of search is extended for finding not only text data from internet that is also used for multimedia data search. Therefore the modifications over the traditional search technique are performed for finding the multimedia data from internet. There are a number of techniques are available for image data processing Among them the content based image analysis and search techniques are more popular. In addition of that the content based image retrieval techniques are more accurate as compared to the other techniques.

Thus in this presented work the content based image retrieval techniques are investigated. In addition of that for improving the image search methodology a new concept is also provided in this study. The improvement on the existing system is based on improving the search methodology according to the user query relevance. In order to justify the proposed solution against the traditional method the proposed technique is implemented and compared with the traditional approach of image retrieval which is based on the given concept in [1].

This chapter provides the understanding about the proposed study domain and the key objectives of the study. In addition of that the technological background of the proposed work is also discussed in this chapter and finally the document organization is discussed in detail.

Basically the image retrieval is a kind of data analysis and retrieval technique. Among them Content based image retrieval technique is an efficient and accurate technique for finding image according to the user query relevance. Therefore the content of images are extracted or distinguished on the basis of the image objects. In order to recover these features for image data retrieval mainly three features namely color, texture and shape features are utilized. These features are recovers the image data pattern and then the similarity matching techniques or classification techniques are employed to obtain the more nearer images form the image directories.

In addition of that for improving the quality of search results in content based image retrieval. There are various other techniques are available that promises to optimize the search results according to the user need such as relevance feedback, by which the user provide the direction of search. Or sometimes the image re-ranking techniques are utilized for re-rank the search results outcomes. In this proposed technique the pseudo relevance feedback technique is utilized for optimizing the search results. The key advantage of the pseudo relevance, which is not consuming additional memory and time for re-rank the search results according to their relevancy.

In further discussion the proposed model design concept and their sub component design aspects are discussed in detail. In addition of that for simulation of the proposed methodology simulation architecture is also provided with their description. Finally the given chapter is summarized.

II. PROPOSED WORK

The entire design of the proposed content based image retrieval technique is demonstrated in two major modules. The detailed understanding about these modules is discussed as:

A. Training Module:

In this phase the user train the proposed data model and during training the image, input object tag are incorporated first then after their feature vectors are estimated. The feature vectors are essential properties of the images which is stored in database for future comparisons and results listing.

B. Testing:

In this phase the user provides query for making search to the data base. In this phase if the input query is in terms of text then the tags are compared with the image contents and results are build and if the query the query is found in terms of image then the feature vectors of input image is calculated and then after the input image features are compared with the target database.

![Fig. 1: proposed system](image)
1) Input Image: That is training phase of the system where system accepts user image as input to learn from example.

2) Add text in images: A small note on image also stored in the data base as annotation of text.

3) Normalize feature vectors: In this phase from multiple values a common value is prepared and stored in data base in place of complete image. In testing session we use these stored values from the database and consumes to extract the optimum results from the database.

4) Input query image: User can make query for search results as image from the query image feature vectors are estimated as in training phase.

5) Feature vectors: Here grid color movement, Canny edge detection, local binary patterns are estimated as feature set.

6) Normalized feature vector: Individual features are contains their own definition and storage of each features can take more space and time for comparison thus a common value is calculated to store them.

7) Text input: User can also place text for search image contains.

8) Search similar tag: The text in image is searched on local database to get image from database.

9) Combine results: Here the result obtained from text query and image query is listed in same place.

10) Filter results: To obtain the optimum results from the data omitted as result.

The given figure 1 describes the training process of the proposed data model where first a provision is made to accept the input image data. The image data is further analyzed using the region growing algorithm, which extend the image features for improving the image feature quality. In next step the color features are extracted therefore that need to analyze the image using the grid color movement analysis. The grid color movement analysis first convert the entire image into small blocks of images and then the formulas for color distribution is applied. The computation of the grid color movement analysis can be described as:

C. Grid Color Moment

Color feature is one of the most widely used features in low level feature. Compared with shape feature and texture feature, color feature shows better stability and is more insensitive to the rotation and zoom of image. Color not only adds beauty to objects but also more information, which is used as powerful tool in content-based image retrieval. In color indexing, given a query image, the goal is to retrieve all the images whose color and texture compositions are similar to those of query image. In color image retrieval there are various methods, but here we will discuss some prominent methods.

The feature vector we will use is called “Grid-based Color Moment”. Here is how to compute this feature vector for a given image:[19]

- Convert the image from RGB for HSV color space
- Uniformly divide the image into 3x3 blocks
- For each of these nine blocks
- Computer its mean color (H/S/V)

\[ x_i = \frac{1}{N} \sum_{i=1}^{N} x_i \]

Where N is the number of pixels within each block, \( x_i \) is the pixel intensity in H/S/V channels.

- Compute its variance (H/S/V)

\[ \sigma^2 = \frac{1}{N} \sum_{i=1}^{N} (x_i - x')^2 \]

- Compute its skewness (H/S/V)

\[ \gamma = \frac{1}{n} \sum_{i=1}^{n} (x_i - x')^3 \]

- Each block will have 3+3+3=9 features, and thus the entire image will have 9x9 features. Before we use SVM to train the classifier, we first need to normalize the 81 features to be within the same range, in order to achieve good numerical behavior. To do the normalization, for each of the 81 features:

- Compute the mean and standard deviation from the training dataset

\[ \mu = \frac{1}{M} \sum_{i=1}^{M} f_i \]

\[ \sigma = \sqrt{\frac{1}{M} \sum_{i=0}^{M} (f_i - \mu)^2} \]

- Where M is the number of images in the training dataset, and \( f_i \) is the feature of the i-th training sample.

- Perform the “whitening” transform for all the data (including both the training data and the testing data), and get the normalized feature value:

\[ f_i' = \frac{f_i - \mu}{\sigma} \]

The computed color features of images are preserved in a feature database. And in next step the edge detection technique is applied on image for recovering the edges the image therefore the canny edge detection technique is applied on image, the canny edge detection technique can be described as:

D. Canny Edge Detection

The purpose of edge detection in general is to significantly reduce the amount of data in an image, while preserving the structural properties to be used for further image processing. Several algorithms exists, and this worksheet focuses on a particular one developed by John F. Canny(JFC) in 1986.[20,21,22]

The algorithm runs in 5 separate steps:

- Smoothing: Blurring of the image to remove noise.
- Finding gradients: The edges should be marked where the gradients of the image has large magnitudes.
- Non-maximum suppression: Only local maxima should be marked as edges.
- Double thresholding: Potential edges are determined by thresholding.
Edge tracking by hysteresis: Final edges are determined by suppressing all edges that are not connected to a very certain (strong) edge.

E. Smoothing

It is inevitable that all images taken from a camera will contain some amount of noise. To prevent that noise is mistaken for edges, noise must be reduced. Therefore the image is first smoothed by applying a Gaussian filter. The kernel of a Gaussian filter with a standard deviation of $\sigma = 1.4$. The effect of smoothing the test image with this filter is shown in Fig. 1.

F. Finding Gradients

The gradient magnitudes (also known as the edge strengths) can then be determined as an Euclidean distance measure by applying the law of Pythagoras:

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

It is sometimes simplified by applying Manhattan distance measure to reduce the computational complexity.

$$|G| = |G_x| + |G_y|$$

Gx and Gy are the gradients in the x- and y-directions respectively.

The Euclidean distance measure has been applied to the test image. The computed edge strengths are compared to the smoothed image in Figure (6).

G. Non-Maximum Suppression

The purpose of this step is to convert the “blurred” edges in the image of the gradient magnitudes to “sharp” edges. Basically this is done by preserving all local maxima in the gradient image, and deleting everything else. The algorithm is for each pixel in the gradient image:

- Round the gradient direction $\theta$ to nearest 45°, corresponding to the use of an 8-connected neighborhood.
- Compared the edge strength of the current pixel with the edge strength of the pixel in the positive and negative gradient direction. I.e. if the gradient direction is north (theta = 90°), compare with the pixels to the north and south.
- If the edge strength of the current pixel is largest; preserve the value of the edge strength. If not, suppress (i.e. remove) the value.

H. Double Thresholding

The edge-pixels remaining after the non-maximum suppression step are (still) marked with their strength pixel-by-pixel. Many of these will probably be true edges in the image, but some may be caused by noise or color variations for instance due to rough surfaces. The simplest way to discern between these would be to use a threshold, so that only edges stronger than a certain value would be preserved. The Canny edge detection algorithm uses double thresholding. Edge pixels stronger than the high threshold are marked as strong; edge pixels weaker than the low threshold are suppressed and edge pixels between the two thresholds are marked as weak.

I. Edge Tracking By Hysteresis

Strong edges are interpreted as “certain edges”, and can immediately be included in the final edges image. Weak edges are included if and only if they are connected to strong edge (with proper adjustment of the threshold levels). Thus strong edges will (almost) only be due to true edges in the original image. The weak edges can either be due to true edges or noise/color variations. The latter type will probably be distributed independently of edges on the entire image, and thus only a small amount will be located adjacent to strong edges. Weak edges due to true edges are much more likely to be connected directly to strong edges.

Edge tracking can be implemented by BLOB-analysis (Binary Large Object). The edge pixels are divided into connected BLOB’s using 8-connected neighborhood. BLOB’s containing at least one strong edge pixel is then preserved, while other BLOB’s are suppressed. The effect of edge tracking on the test image is shown in Figure.

Furthermore the texture analysis of input image is performed thus here the local binary pattern analysis technique is employed on image. The LBP feature analysis technique can be described as:
J. Local Binary Pattern

Given a pixel in the image, an LBP [21] code is computed by comparing it with its neighbors:

\[
LBP_{p,r} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p
\]

\[
s(x) = \begin{cases} 0 & x \geq 0 \\ 1 & x < 0 \\ \end{cases}
\]

where \(g_c\) is the gray value of the central pixel, \(g_p\) is the value of its neighbors, \(P\) is the total number of involved neighbors and \(R\) is the radius of the neighborhood. Suppose the coordinate of is \((0, 0)\), then the coordinates of \(g_p\) are

\[
\left(R\cos\left(\frac{2\pi p}{P}\right), R\sin\left(\frac{2\pi p}{P}\right)\right)
\]

The gray values of neighbors that are not in the image grids can be estimated by interpolation. Suppose the image is of size \(I\times J\). After the LBP pattern of each pixel is identified, a histogram is built to represent the texture image:

\[
H(k) = \sum_{i=1}^{I} \sum_{j=1}^{J} f(LBP(i,j), k), k \in [0, K]
\]

\[
f(x, y) = \begin{cases} 1 & x = y \\ 0 & otherwise \end{cases}
\]

where \(K\) is the maximal LBP pattern value. The \(U\) value of an LBP pattern is defined as the number of spatial transitions (bitwise 0/1 changes) in that pattern:

\[
U(LBP_{p,r}) = |s(g_{p-1} - g_c) - s(g_c - g_{p-1})|
\]

\[
+ \sum_{p=1}^{P-1} |s(g_p - g_c) - s(g_c - g_{p+1})|
\]

The uniform LBP patterns refer to the patterns which have limited transition or discontinuities (\(U \leq 2\)) in the circular binary presentation. In practice, the mapping from \(LBP_{p,r}\) to \(LBP_{p,r}^{\text{un}}\), which has \(P^2(P-1)+3\) distinct output values, is implemented with a lookup table of \(2^P\) elements. To achieve rotation invariance, a locally rotation invariant pattern could be defined as:

\[
LBP_{p,r}^{\text{rin}} = \sum_{p=0}^{P-1} s(g_p - g_c) i f(U(LBP_{p,r}) \leq 2)
\]

\[
P + 1 \text{ otherwise}
\]

The mapping from \(LBP_{p,r}\) to \(LBP_{p,r}^{\text{un}}\) which has \(P+2\) distinct output values.

Finally the tag on the image is also included and all three features and their tags are saved on the database. This preserved features and tags of the image objects are used for search the data according to the user supplied query. The next section described the test phase of the proposed model.

K. Testing

The testing model of the proposed content based image retrieval model is given using figure 3.5. This model is basically used for extracting the images from data base according to the input query. Therefore a provision is made to provide input for search there are two kinds of input can be applied first by using the image query and second for the text query. User can search the images by the text or image or from both image and text. After user query input the system analyze the input query type and according to the user query if the input is only text then the system search on the tag data base here for comparing the image existing tag and user input text the cosine similarity is utilized. And if the input query is an image than first the query image’s low level features are computed and then compared with the feature database. In these conditions for finding the more relevant images from the database the KNN (k nearest neighbor) algorithm is applied. Finally according to the omitted distance from the available images the search results are ranked and reflected as list of images retrieved.

For classifying image the K-nearest neighbour algorithm can be described as:

The K-nearest-neighbour (KNN) algorithm measures the distance between a query scenario and a set of scenarios in the data set. We can compute the distance between two scenarios using some distance function \(d(x, y)\), where \(x, y\) are scenarios composed of features, such that:

\[
X = \{x_1, x_2, x_3, \ldots \}
\]

\[
Y = \{y_1, y_2, y_3, \ldots \}
\]

Two distance functions are discussed here:

Absolute distance measuring:

\[
d_A(x, y) = \sum_{i=1}^{N} |x_i - y_i|
\]

Euclidean distance measuring:

\[
d_A(x, y) = \sum_{i=1}^{N} \sqrt{x_i^2 - y_i^2}
\]

Because the distance between two scenarios is dependant on the breaks, it is suggested that resulting distances be scaled such that the arithmetic mean across the dataset is 0 and the standard deviation is 1. This can be accomplished by replacing the scalars with according to the following function:

\[
x' = \frac{x - \bar{x}}{\sigma(x)}
\]

Where the un-scaled value is the arithmetic mean of feature across the data set, is its standard deviation, and is the resulting scaled value.

The arithmetic mean is defined as:

\[
\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i
\]

We can then compute the standard deviation as follows:

\[
\sigma(x) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2}
\]

KNN can be run in these steps:

1) Store the output values of the \(M\) nearest neighbors to query scenario \(Q\) in vector \(r = \{r_1, \ldots, r_m\}\) by repeating the following loop \(M\) times:

   - Go to the next scenario \(S_i\) in the data set, where \(I\) is the current iteration within the domain \(\{1, \ldots, P\}\)
   - If \(Q\) is not set or \(q < d(q, S_i)\): \(q \leftarrow d(q, S_i), t \leftarrow O_i\)
   - Loop until we reach the end of the data set.
   - Store \(q\) into vector \(c\) and \(t\) into vector \(r\).

2) Calculate the arithmetic mean output across \(r\) as follows:
$$\bar{r} = \frac{1}{M} \sum_{i=1}^{M} r_i$$

3) Return $r$ as the output value for the query scenario $q$.

L. Simulation Architecture

In order to simulate and organize the system the following model is utilized as given in figure 3.6. According to the given system architecture of the proposed model the training and testing modules are incorporated in this model. First the training data samples are produced as input the system and then the image features are computed these image feature vectors are shape, color and texture features. After normalizing them the user input the object tags into the images and the system store the normalized feature vectors with their associated tag into the database storage. On the other hand during testing or search the user query input is provided to the system and according to the user query the system processes the user request.

![Fig. 5: System Architecture](image)

Thus after computing the query data input the system perform search on the basis of defined classifier for images. The search system returns more than one outcomes. These search outcomes are ranked according to the user query input and the results are demonstrated. At the same time the performance of the system is evaluated and the preserved for future visualization.

III. Result Analysis

The implemented model of the proposed image retrieval system is discussed in this chapter. Therefore the performance of both the techniques traditional as given in [1] and the proposed model is compared in this chapter over different performance parameters.

A. Memory Consumption

The amount of space in main memory required to execute the algorithm is known as the memory consumption of the algorithm. The performances in terms of memory for both the algorithms are demonstrated using the figure 3.1.

![Fig. 6: Memory Consumption](image)

According to the outcomes as given in Fig. 6 the performance of the algorithm is efficient. But sometimes the memory consumption of the proposed algorithm is higher than 35000KB. In order to demonstrate the performance of the system the X axis contains the different experiments performance with the system and the Y axis shows the memory consumption of the system in terms of KB. The performance of the proposed method is provided using the blue line.

B. Training Time

The amount of time consumed for train the algorithm is termed here as the training time. The training time of the system is given using figure 7.

![Fig. 7: Training Time](image)

In this diagram the blue line shows the performance of the proposed system. For representation of the performance the X axis shows the different experiments performed on the system and the Y axis shows the consumed time in second. According to the obtained results the performance of the proposed algorithm is consumes less time therefore the performance of the proposed technique is much optimum as compared to traditionally available technique.

C. Search Time

The search time of the system is termed as the amount of time required to search the images from data base using the input query. The time consumption of the system for the proposed techniques is reported using the figure 8. In this diagram the X axis shows the number of experiment performed and the Y axis shows the amount of time consumed in terms of seconds. According to the obtained performance the proposed technique consumes less time.
D. Accuracy
In the predictive data models the amount of correctly identified patterns are known as the accuracy of the predictive system. That can be calculated using the following formula.

\[
\text{accuracy} = \frac{\text{total correctly classified samples}}{\text{total input samples}} \times 100
\]

The estimated accuracy of the proposed data model is given using figure 9, in this diagram the X axis shows the number of data samples on which the model performs training and the Y axis simulated the total percentage of samples which are correctly identified. According to the obtained results when the model is trained with less amount of data than the performance of prediction is poorer and as the training samples are increases the performance of the system is enhanced.

E. Error rate
The amount of miss-classified samples is known as the error rate of system. That can be estimated using the following formula.

\[
\text{error rate} = \frac{\text{total misclassified samples}}{\text{total samples as input}} \times 100
\]

Or

\[
\text{error rate} = \text{accuracy value} - 100
\]

IV. CONCLUSION
This chapter draws the conclusion of entire research work performed therefore the experimental and observational facts are given as conclusion. Internet is a source of information and data, where different kinds and formats of data are available for use. Internet users can search the specific kind of data from web using the web search engines. The extraction from the data from a given source of data is known as the information retrieval. In this presented work the image retrieval systems are key area of research. During investigation various kinds of image search systems are found some of them are working on the basis of text associated with the images and some of them are works on the basis of image contents. The content based image retrieval systems are much accurate than the text based search systems. But due to annotation concepts that technique can be more appropriate then content based search.

In this presented work a hybrid model for image retrieval proposed. That technique usage the concept of image annotation and content based image retrieval. Therefore the technique allows a user to search image by their text query as well as the image query. For design aspects the system first utilizes the feature extraction techniques thus for shape feature the canny edge detection technique, for color features the grid color movement analysis and for texture analysis the local binary pattern is investigated.

The implementation of the proposed content based image retrieval system is given by the JAVA technology. After implementation of the concept the performance of the technique is also investigated in terms of time and space complexity and the search relevancy is measured in terms of accuracy and error rate. The performance summary of the proposed model is given using table 1.

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Parameters</th>
<th>Proposed method</th>
<th>Traditional system</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Memory usage</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>2</td>
<td>Training time</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>3</td>
<td>Search time</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>4</td>
<td>Accuracy</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>5</td>
<td>Error rate</td>
<td>High</td>
<td>Low</td>
</tr>
</tbody>
</table>

Table 1: performance summary

According to the obtained results the performance of the proposed system is optimum as compared to the traditional data model. That produces high accurate image listing during search and also provides the outcomes in less amount of time.

V. FUTURE WORK
In this presented work the key focus is to improve the traditional work for content based image retrieval. Therefore
a hybrid approach of image search is presented and implemented using the JAVA technology. The presented technique of image retrieval is much efficient and provides more accurate results as compared to the other available technique. In near future the performance of the system in terms of memory consumption is required to be optimized.

REFERENCE


