Experiment No. : 10

Name : Parth Gharat

Roll number : 23

Date of Performance : 5/10/2023

Date of Submission : 21/10/2023

Aim: To Create Program to perform a retrieving Image and Searching.

Objective:

The fundamental need of any image retrieval model is to search and arrange the images that are in a visual semantic relationship with the query given by the user.

Most of the search engines on the Internet retrieve the images based on text-based approaches that require captions as input.

Theory:

There are several crucial processes involved in developing an image retrieval and search program. Gather and arrange your image dataset first, making sure it is well-structured for access. Then, decide the features such as colour histograms, texture features, or deep learning-based features you want to use to represent images. Create a database to store these features along with pertinent metadata, such as file paths or image descriptions, after extracting them from your dataset. Create a querying system that lets users input a search image and then preprocess it to extract its features.

Use the proper similarity metrics to determine the degree of similarity between the query image's features and those in your database. Find the top k images that are most similar to the search image by ranking the images according to their similarity. Consider developing a user interface, which could be a web app, desktop program, or command-line tool, if you want your system to be user-friendly. Users should see the findings, along with the top k comparable images and associated metadata.

Also use indexing or approximate search methods to optimize your system's performance, especially when working with enormous datasets. Use assessment measures like precision, recall, and F1-score to thoroughly test and assess your system using a variety of query images to confirm its effectiveness. Consider deployment on a web server, as an application, or through an API if you intend to share your solution. Last but not least, practice continuous improvement by modifying your picture retrieval system in response to user feedback and changing requirements.

Code :

```
import cv2

import numpy as np

import os

from IPython.display import Image, display

 def calculate_histogram(image):

    hist = cv2.calcHist([image], [0, 1, 2], None, [8, 8, 8], [0, 256, 0, 256, 0, 256])

hist = cv2.normalize(hist, hist).flatten()     return hist

 def chi_squared_distance(hist1, hist2):

    return 0.5 * np.sum(((hist1 - hist2) ** 2) / (hist1 + hist2 + 1e-10))

 def build_image_database(image_folder):

        image_database = {}      for image_filename in os.listdir(image_folder):

        image_path = os.path.join(image_folder, image_filename)

        image = cv2.imread(image_path)

             if image is not None:

            hist = calculate_histogram(image)

 image_database[image_filename] = hist     return image_database

 def search_similar_images(query_image_path, image_database, top_k=5):
```

```
query_image = cv2.imread(query_image_path)

query_hist = calculate_histogram(query_image)

similarities = {}

for image_filename, hist in image_database.items():

similarity = chi_squared_distance(query_hist, hist)

similarities[image_filename] = similarity

similar_images = sorted(similarities, key=similarities.get)[:top_k]

return similar_images

 image_folder = '/content/sample_data/Dataset_cars' image_database =

build_image_database(image_folder)

 query_image_path = '/content/sample_data/Dataset/cars.png'

similar_images = search_similar_images(query_image_path, image_database, top_k=5)

 for idx, image_filename in enumerate(similar_images):

 similar_image_path = os.path.join(image_folder, image_filename)

display(Image(filename=similar_image_path))
```

Output:

Input Image :

Output :

Conclusion:

Creating a program to perform image retrieval and searching is a complex and valuable task. Such a program involves the use of computer vision, image processing, and potentially machine learning techniques to enable users to efficiently find and retrieve images based on various criteria. The program provided allows for efficient image retrieval and searching based on color histograms. It utilizes OpenCV for image processing and analysis, enabling the creation of a database of images and their corresponding colour histograms. The development of such a program requires careful consideration of user interface design, database management, and algorithm selection. Ultimately, a successful image retrieval and searching program can greatly enhance the user's ability to manage and access their image collection, making it a valuable tool for individuals and organizations alike.