Report Document - Phase 1

Title: k-nearest image retrieval

Group members:

1.Chandra Kishore Reddy Gurram
2.Rahul Rajaram
3.Bevin Biju Thomas
4.Lalit Arvind Balaji
5.Khadyothan Choudari Dasari
6.Venkata Sai Rohit Bathi

Abstract:

This project presents a comprehensive image similarity analysis system that leverages multiple feature extraction techniques to compare and retrieve similar images from a database. The system consists of various components, including color moments, histogram of gradients, average pool, and feature vectors extracted from ResNet50's Layer 3 and fully connected (FC) layers. Users can interact with the system through a user-friendly interface provided by the `starter.py` script.

The project involves three main functionalities: Testing, Database Population, Feature Vector Visualization.Read the README.md file in the "code" for the detailed explanation of the project structure. This report details the project's architecture, data collection, preprocessing, and code structure. It also discusses the results of extensive testing and provides insights into potential future improvements. The system's ability to analyze and retrieve visually similar images holds promise for applications such as image search engines, content recommendation systems, and more.

Introduction

Primary Goal and Objectives:

The primary goal of this project is to develop a comprehensive image similarity analysis system that can effectively compare and retrieve visually similar images from a database. The project's specific objectives include:

1. Feature Extraction
2. Database Integration
3. User-Friendly Interface:
4. Testing and Evaluation:
5. Documentation and Reporting

By achieving these goals and objectives, the project aims to provide a versatile tool for image similarity analysis, which can have applications in image search, recommendation systems, content retrieval, and other areas where Visual content comparison is essential.

Motivation behind the project.

The motivation behind this project stems from the growing importance of visual content analysis and image similarity retrieval in various fields,

including:
1.Content-Based Image Retrieval
2. Recommendation Systems
3. Visual Search Engines
4. Medical Imaging
5. Art and Cultural Heritage
6. Data Organization and Retrieval

Given these real-world applications and the increasing volume of image data, the motivation behind this project is to create a flexible and effective image similarity analysis system that can be adapted for various use cases. The project aims to provide a valuable tool for researchers, developers, and organizations seeking to harness the power of visual content analysis.

Description of the proposed solution/implementation

Project Architecture:

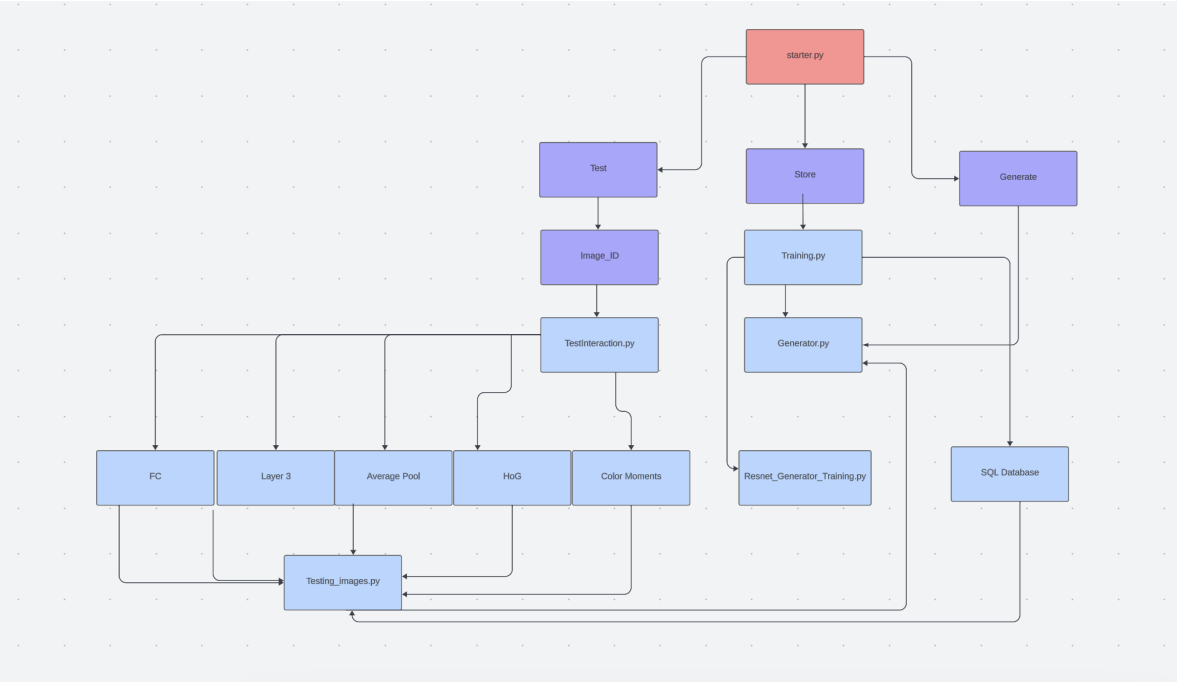
The overall architecture of the project encompasses several key components that work together to enable image similarity analysis and retrieval. Here's an explanation of the architecture and its main components:

1.User Interface (starter.py): The user interacts with the project through the `starter.py` script, which serves as the entry point to the system. This script provides a user-friendly interface and presents the user with options to perform different actions, including image similarity testing, database population, and feature vector visualization.
2. Feature Extraction: The heart of the project lies in feature extraction, which is crucial for describing the visual content of images. The project employs various feature extraction techniques, such as:
- Color Moments: Extracts color distribution information from images.
- Histogram of Gradients (HOG): Captures gradient-based texture and shape features.
- Average Pool: Computes average values of feature maps from an image.
- ResNet50 Layer 3: Extracts feature vectors from Layer 3 and fully connected (FC) layers of the ResNet50 deep neural network.
3.Database (MySQL, local): To facilitate efficient image retrieval, a MySQL database is used to store feature vectors generated by the feature extraction techniques. The database is structured to accommodate feature vectors for a vast dataset of images.
4.Generator (Generator.py): The `Generator.py` script is responsible for populating the database with feature vectors. It extracts features from a dataset of images using the various feature extraction techniques and stores these feature vectors in the database for future retrieval.
5.Testing (TestInteraction.py and Testing_images.py): This component allows users to perform image similarity tests. When a user selects this option, the

`TestInteraction.py` script is called, presenting a list of available vector spaces (feature extraction techniques). After choosing a vector space, the script calls `Testing_images.py`, which calculates the similarity between the input image and the images in the database based on the selected vector space and distance measures. The results are stored in the "outputs" folder and presented to the user.
6.Feature Vector Visualization: Users have the option to visualize the feature vectors of a specific image within their chosen vector space. This component allows users to gain insights into how different feature extraction techniques represent images visually.

These components work together to create a comprehensive image similarity analysis system. Users can interact with the system to perform tests, populate the database, and explore feature vectors. The system leverages a variety of feature extraction techniques and database management to enable efficient content-based image retrieval and analysis.

Flowchart:



System Requirements:

Before running the provided Python script (code.py), ensure that you have the necessary packages and prerequisites installed. Below are the steps to set up your environment:

To set up your environment for running the provided Python script (code.py), follow these instructions:

1. Install Python 3:
   - Ensure that Python 3 is installed on your system.
2. Check Pip Installation:
   - Verify if `pip` is available on your system.
3. Install Required Packages:
   - Open a command prompt or terminal and use `pip` to install the necessary Python libraries.
   - Install `torchvision`
   - Install `mysql.connector`:
   - Install `matplotlib`:
   - Install `scikit-learn` (for `MinMaxScaler`)
4. Database Configuration:
   - Ensure that you have configured the MySQL database connection within your Python script (code.py) with the appropriate credentials and connection details.

By following these steps, you'll set up your environment with Python 3 and the necessary packages to run your code successfully. Make sure to adjust the database connection settings and other parameters within your code.py script as needed for your specific project.

Installation and execution instructions

The entire project is organized into three main parts: code, output, and a detailed report. The code section consists of eight Python files and a README.md file. Below is an explanation of the starter.py file and its functionality.

starter.py - Initialization and Execution File --> This is the primary file to execute the entire project. The project is implemented in Python 3, and you can run it by executing the following command in the terminal:

```
python3 starter.py
```

Upon execution, the user is presented with options on how to interact with the project:

1. Test: Choose this option if you have an image ID and want to display the top 10 most similar images to your input image using five different vector spaces (Color Moments, Histogram Of Gradients, Average Pool, Layer 3, and FC layers of ResNet50). Selecting this

option will call the TestInteraction.py file, which displays all available vector spaces. You can then choose one from the list. This will internally invoke the Testing_images.py script, which generates the results and stores them in the "outputs" folder. The result will be a JPEG image containing the 10 most similar images to your input image.

2. Store: Use this option to populate the database (in my case, MySQL, local) with all the feature vectors. This option calls the Generator.py script to generate the feature vectors for all five vector spaces and stores them in the database. This database is used in the Test phase to retrieve the k-nearest images to the input image using various distance measures.

3. Generate: Choose this option if you want to view the feature vectors of any input image. This option allows you to select an image and a specific vector space. It then displays the feature vectors in a more human-readable format on the screen.

Data Collection and Preprocessing:

In this project, the Caltech 101 database served as the primary data source for image analysis and similarity testing.

Feature Extraction:

1. Color Moments: Objective: Color moments capture statistical information about the distribution of RGB colors in an image, including its mean, standard deviation, and skewness. The resulting vector is of shape (900,)

2. Histogram of Gradients (HOG): Objective: HOG is a texture descriptor that quantifies the distribution of gradient orientations in an image. It highlights local shape and texture information.The resulting vector is of shape (900,)

3. ResNet50 Layers: Objective: ResNet50 is a deep neural network architecture. Extracting feature vectors from specific layers (e.g AvgPool,Layer 3 and fully connected (FC) layers) captures hierarchical image representations.The resulting vector is of shape (1024,). (1024,) and (1000,) respectively

Database Setup:

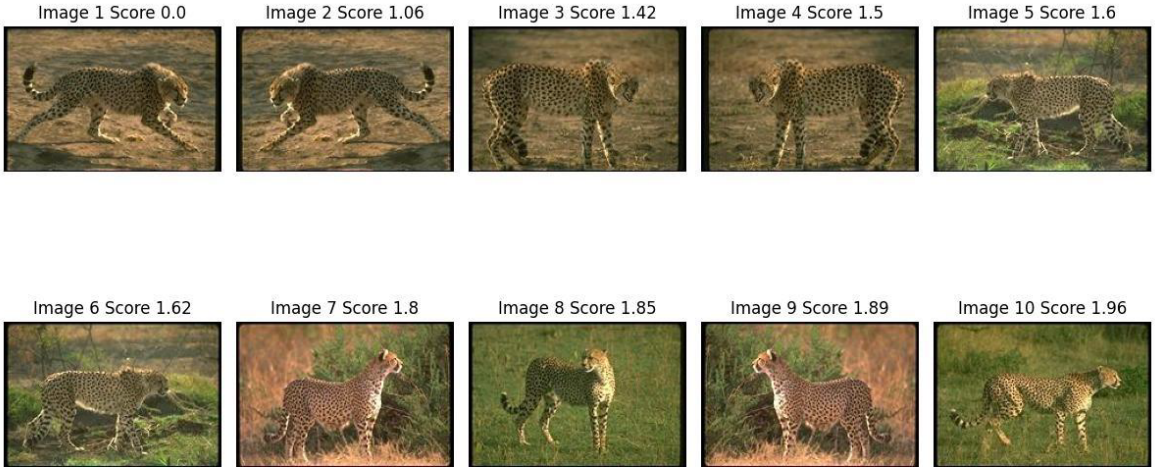
In this project, MySQL was chosen as the relational database management system (DBMS) for storing and managing the feature vectors. MySQL is a popular open-source DBMS known for its reliability, performance, and ease of use. It is well-suited for handling structured data, making it an excellent choice for storing feature vectors and metadata associated with images.

Database Schema:

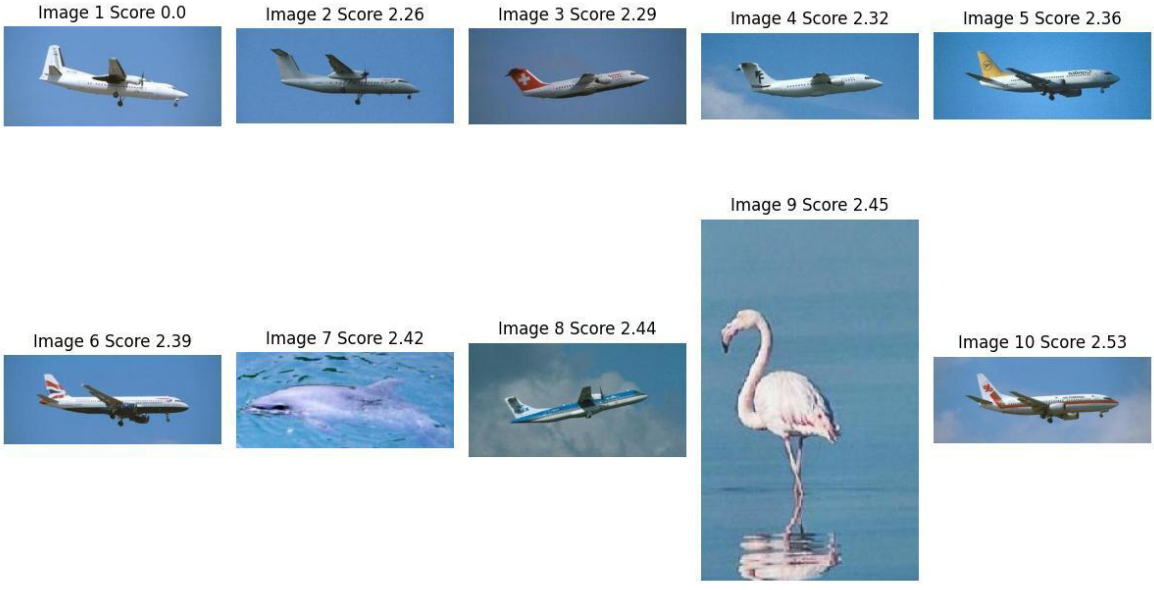
image_Id int,
label varchar(255),
Feature_Descriptor longtext,
HoG_Values longtext,
AVG_Pool longtext,
layer3 longtext,
FC longtext

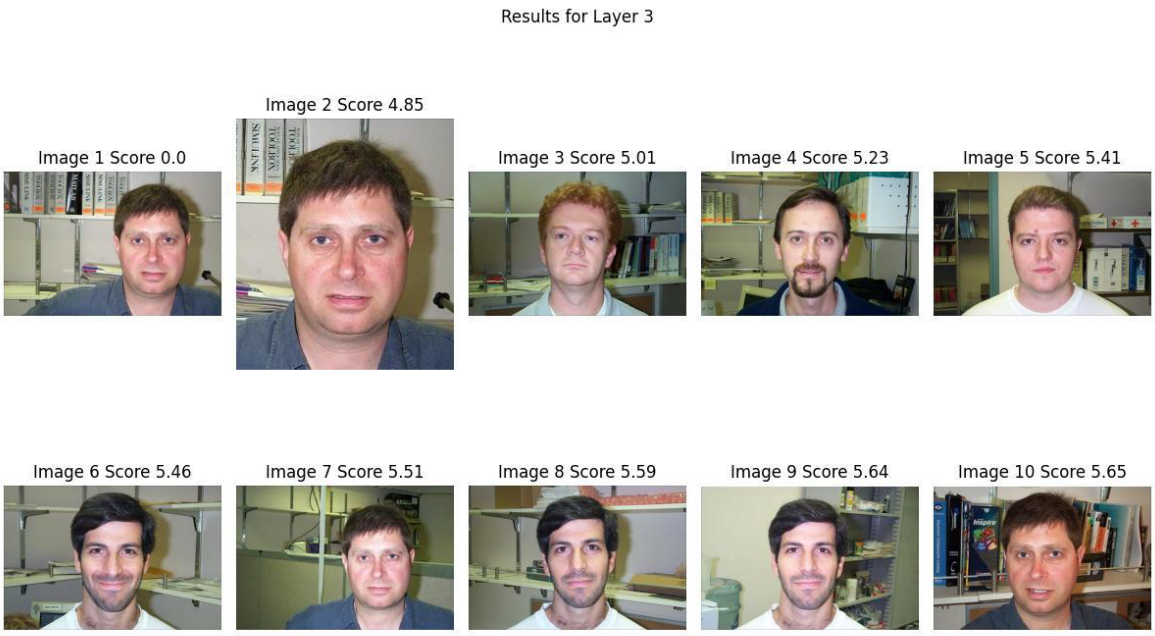
Results:

Results for Average Pool Layer



Results for Color Moments





Conclusion:

In this phase of the Multimedia & Web Databases project, we have successfully developed a robust system for extracting image features and conducting similarity analysis. By employing a variety of visual models, which include color moments, histogram of gradients (HOG), and features derived from a pre-trained ResNet50 neural network, we have enabled the efficient retrieval and visualization of image feature descriptors.

Throughout the execution of our project, we encountered a notable observation concerning the color moments feature extraction for Image ID 0. The outcomes for this particular image did not align with our initial expectations. This discrepancy serves as an illustrative example of the limitations associated with using the Euclidean distance metric in the context of color moments, indicating that it may not be the most suitable measure for this specific feature model.

It is important to emphasize that, despite this observation, the remaining images within our dataset produced satisfactory results when subjected to color moments feature extraction.

Potential Future Work:

Distance Metric Enhancement: In future work, addressing the challenge observed with color moments' Euclidean distance metric is imperative. Exploring and selecting more appropriate distance or similarity measures tailored to color-based features should be a priority. This could involve experimenting with alternative metrics like Earth Movers' distance or statistical measures designed to capture nuances in color distribution. Such exploration can potentially lead to significantly improved results for color-based feature analysis.

Normalization and Dimensionality Reduction: An essential step is to eliminate the assumption of not normalizing feature descriptors. Normalization can standardize feature values across different images, enhancing the consistency and interpretability of distance measures. Additionally, the high dimensionality of feature descriptors (e.g., 900, 1000,

1024) presents challenges in terms of computational complexity and storage. Future work should delve into dimensionality reduction techniques, such as Principal Component Analysis (PCA) or t-Distributed Stochastic Neighbour Embedding (t-SNE). These methods can effectively reduce dimensionality while preserving essential information, streamlining analysis and visualization processes.

By addressing these areas in future work, our image feature extraction and similarity analysis system can undergo significant improvements, making it more adaptable and effective in a broader spectrum of multimedia and web database applications.

Appendix:

In Phase 1 of the Multimedia & Web Databases project, it's crucial to highlight that all group members completed their project tasks independently and submitted their individual deliverables. Nevertheless, there was active collaboration and discussion within the group when addressing challenges, such as selecting appropriate distance metrics, and resolving other project-related complexities. This collaborative exchange of ideas and assistance among group members contributed to a collective understanding and enhanced the overall project outcomes