CSE 515: Multimedia and Web Databases, Project Phase 1

Feature Extraction, Storage and Similar Image Retrieval

Abstract

This project introduces an approach to extract features from various visual models, such as Resnet 50 layers (Layer 3, Avgpool, & FC), Colour Moments, and Histogram of Oriented Gradients (HOG). These features are then stored to create an Image Retrieval System. This system allows users to find similar images within a large dataset based on different feature descriptors. It utilizes computer vision techniques and deep learning models for feature extraction and comparison. Users can input an image ID, and the system will provide a set of visually similar images in response. This project aims to provide an effective solution for content-based image retrieval in multimedia databases.

Keywords: Resnet 50, FC, Color Moments, HOG.

Group 21

Group members

Ariana Bui
Kasi Chikkala
Maryam Cheema
Peter Kim
Rajat Pawar
Yash Dhamecha

Introduction

In today's era of ubiquitous digital media, the exponential growth of visual content, particularly images, has presented a profound challenge in effectively organizing and retrieving specific visual information. This project tackles this challenge by delving into the realms of feature extraction, storage and content-based image retrieval. We explore the extraction of distinctive features from various visual models, including Resnet layers (Layer 3, Avgpool, & FC), Colour Moments, and Histogram of Oriented Gradients (HOG).

The primary objective of this project is to develop an Image Retrieval System that empowers users to navigate through extensive datasets and discover visually similar images based on distinct feature descriptors. By harnessing fundamental computer vision techniques and leveraging deep learning models, we aim to provide a practical solution for content-based image retrieval in multimedia databases.

In this report, we will delve into the technical aspects of our approach, detailing the process of feature extraction, feature storage, and the implementation of an efficient image retrieval mechanism. We will also discuss the tools and libraries employed, such as Python, PyTorch, and torchvision. Moreover, we will showcase the results of our system, demonstrating its capability to facilitate seamless image search based on visual characteristics.

Terminology

Resnet 50 - It is a 50-layer convolutional neural network (48 convolutional layers, one MaxPool layer, and one average pool layer).

FC - Fully connected layer of Resnet 50 model.

Color Moments - They are measures that characterize color distribution in an image. It describes how similar two images are based on color.

Histogram of oriented Gradients - It is a feature descriptor used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image.

Image Retrieval: The process of finding images similar to a given query image based on multiple types of features or descriptors.

Feature Descriptors: Quantitative representations of image characteristics used for comparison and retrieval.

Deep Learning: A subset of machine learning involving neural networks with multiple layers, used for complex feature extraction and pattern recognition.

Computer Vision: The field of study that focuses on enabling computers to interpret and understand visual information from the world.

Goal Description (Problem Specification)

The core objective of this project is to design and implement an Image Retrieval System capable of retrieving visually similar images from a vast dataset. The specific challenges addressed include:

1. Feature Extraction: Develop methods to extract meaningful and distinctive features from images, encompassing Resnet layers (Layer 3, Avgpool, & FC), Colour Moments, and Histogram of Oriented Gradients (HOG). These features will serve as the basis for comparing and retrieving images.

2. Feature Storage: Create a robust system for storing extracted features efficiently, ensuring rapid access to data during retrieval operations.

3. Similarity Computation: Implement various similarity measures tailored to each feature descriptor, enabling accurate comparison between the query image and images within the dataset.

4. User-Friendly Interface: Construct an intuitive user interface that allows users to input an image ID and receive a set of visually similar images, thereby facilitating seamless exploration of the dataset.

Assumptions

1. The dataset used for image retrieval is Caltech-101, consisting of a wide variety of objects.
2. Users have access to the system's interface, which allows them to enter an image ID.
3. Feature descriptors, such as HOG, Color Moments etc. are computed and stored for all images in the dataset.
4. The system can access a MongoDB database to store and retrieve feature descriptors efficiently.
5. The deep learning model (ResNet-50) used for feature extraction is pre-trained and loaded using PyTorch.
6. Users are familiar with the interface provided by the python program and will use the system for tasks like displaying feature vectors for an image ID and finding visually similar K images.

Description of the Proposed Solution/Implementation

The Image Retrieval System employs a combination of feature descriptors and deep learning to achieve its goal. The following components make up the system:

1. Feature Extraction: The system extracts feature descriptors from images in the dataset using techniques such as HOG, Color Moments, and features from a

pre-trained ResNet-50 model. These descriptors are used to represent the visual characteristics of each image.

2. Database Storage: Extracted feature descriptors are stored in a MongoDB database for efficient retrieval and comparison.

3. Similarity Measurement: When a user uploads an input image, the system calculates similarity scores between the input image's feature descriptor and those of images in the database. Various similarity measures, such as cosine similarity and Euclidean distance, are used.

4. Ranking and Retrieval: The system ranks the database images based on their similarity scores and retrieves the top-k similar images to the user's query.

5. User Interface: The system provides a user-friendly web interface where users can upload query images and view the retrieved similar images.

Feature implementation

1. HOG (Histogram of Oriented Gradients) Feature:

HOG features are extracted by first converting the input image to grayscale. The image is then resized to a fixed size of 300 x 100 pixels. Gradients in the image are computed using the Sobel operators to find the intensity variations in the x and y directions. These gradients are used to calculate gradient magnitude and direction for each pixel. The image is divided into a 10x10 grid, and for each cell in the grid, a histogram of gradient orientations is created. This results in a 10x10x9-dimensional feature vector representing the HOG features of the image.

2. Color Moments Feature:

Color Moments features capture color information from the image. The image is first resized to 300 x 100 pixels. It is then divided into a 10x10 grid. For each grid cell, color moments, including mean, standard deviation, and skewness, are computed for each of the three color channels (Red, Green, and Blue) in the RGB color space. This process results in a 10x10x3x3 = 900-dimensional feature descriptor that combines these color moments.

3. ResNet Layer 3 Feature:
For ResNet Layer 3 features, the input image is resized to 224x224 pixels. A hook is attached to the output of "layer3" of the pre-trained ResNet architecture. The feature vector is then obtained by averaging each 14x14 slice of the 1024x14x14-dimensional tensor produced by "layer3." The final feature is a 1024-dimensional vector representing the image.

4. ResNet Avgpool Feature:

ResNet Avgpool features are extracted by resizing the image to 224x224 pixels and attaching a hook to the output of the "avgpool" layer of the ResNet model. The resulting 2048-dimensional vector is reduced to 1024 dimensions by averaging consecutive entries.

5. ResNet Fully Connected (FC) Feature:
The ResNet FC feature is computed by resizing the image to 224x224 pixels and attaching a hook to the output of the "fc" (fully connected) layer of the ResNet model. This results in a 1000-dimensional tensor representing the image features.

These feature extraction methods provide diverse representations of the input images, allowing for various types of image analysis and retrieval based on different visual characteristics.

Interface Specifications

The system's user interface allows users to perform the following actions:

1. Task 1 interface specs:
   Input:
   Image ID: Unique identifier for the image to be processed.

   Feature Model Selection: User specifies one of the following feature models: Color Moments (CM10x10), Histograms of Oriented Gradients (HOG), ResNet-AvgPool-1024, ResNet-Layer3-1024, or ResNet-FC-1000.

   Output:
   Visual Representation: The program displays the input image for visual reference.

   Feature Descriptors: Human-readable feature descriptors extracted based on the selected model. The format of the output depends on the chosen feature model.

2. Task 2 interface specs:
   Input:
   None, as this task involves processing all images in the dataset.

   Output:
   Feature Descriptors Persistence: The program extracts and stores feature descriptors for all images in the dataset to MongoDB.

3. Task 3 interface specs:
   Input:
   Image ID: Unique identifier for the query image.

   K: Number of most similar images to retrieve.

   Output:

Visual Representation: The program displays the query image for visual reference.

K Most Similar Images: Visual representation of the K most visually similar images to the query image based on the selected feature model. Also, the output is saved to the Outputs directory.

Distance/Similarity Scores: For each of the K matches, the program lists the corresponding distance or similarity score for all feature models.

System Requirements/Installation and Execution Instructions

System Requirements:

To run the Image Retrieval System, the following requirements must be met:

1. Python 3.7 or higher installed.
2. Required Python packages installed (PyTorch, torchvision, pymongo, scipy, etc.).
3. Access to the Caltech-101 dataset (downloadable from source).

System installation:

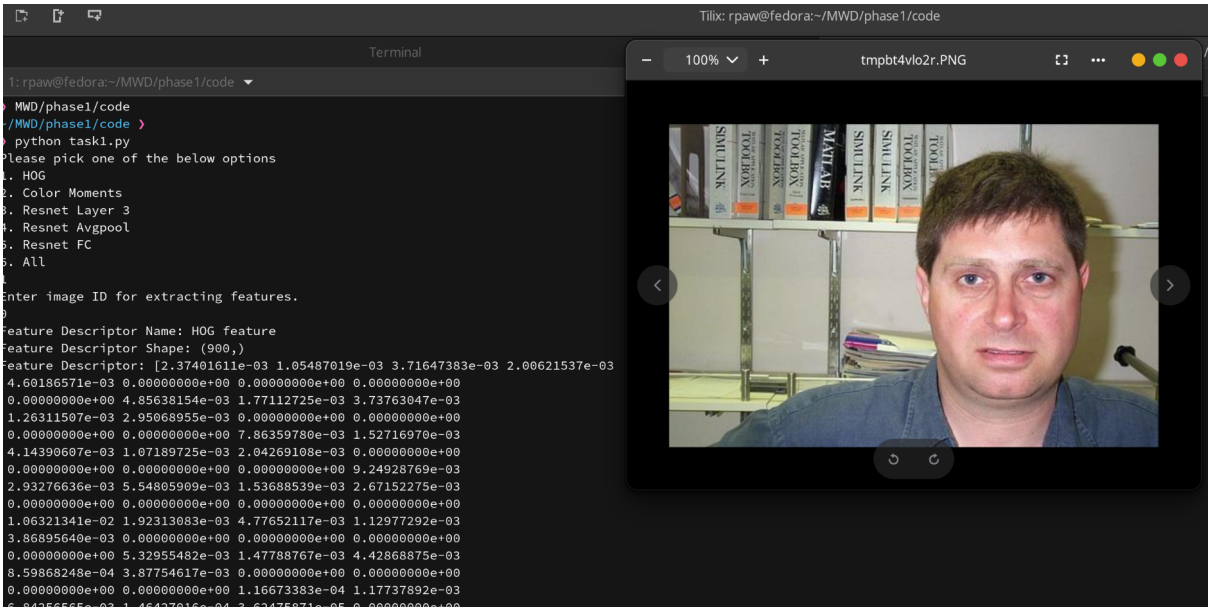
1. Unzip the project repository.
2. Install the required Python packages using `pip install -r requirements.txt`.
3. Ensure `MongoDB` is running on the local machine and update the MongoDB connection string in the code.
4. Create Index on image ID for faster access

```
db.feature_descriptors.createIndex({"image_id"},{unique: true})
```

Execution Instruction:

Task 1:

1. Run `task1.py` to extract all feature descriptors including Histogram of Oriented Gradients (HOG), Colour Moments, and features from a pre-trained ResNet-50 model for a given image.
2. Enter image ID to extract its corresponding feature descriptors

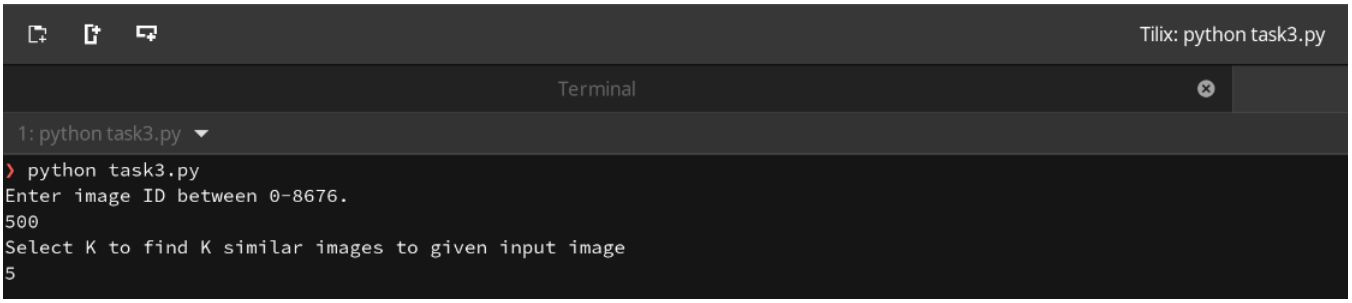


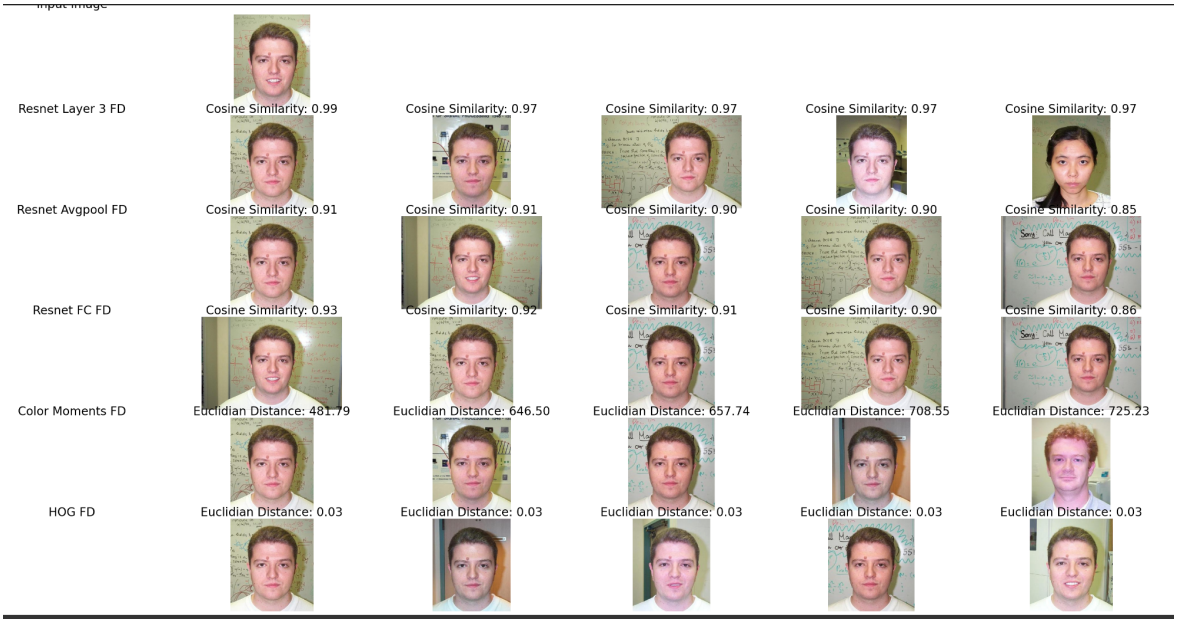
Task 2:

1. Run `task2.py` to save extracted feature descriptors for the whole dataset to a MongoDB database for efficient retrieval and analysis.
2. No input is required

Task 3

1. Run `task3.py` to perform similarity analysis on the dataset by calculating the similarity between images using different feature descriptors and visualize the results.
2. Enter image ID and value of K to retrieve K similar images to the given input image.





Related Work

This project builds upon existing research and implementations in the fields of image retrieval, feature extraction, and deep learning. Notable related work includes:

1. N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," Computer Vision and Pattern Recognition (CVPR), 2005.[https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf] - Discusses the use of HOG feature in detecting humans.
2. Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). "Object detection with discriminatively trained part-based models." IEEE transactions on pattern analysis and machine intelligence, 32(9), 1627-1645.[https://ieeexplore.ieee.org/document/5255236] - Discusses the use of HOG features in object detection, which is a fundamental application.
3. "Image Retrieval Using Colour Moments" by Na Li and Jianwei Niu[https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8938070/]. - This paper investigates the use of color moments for image retrieval and proposes an image retrieval algorithm based on color moments.
4. Retrieving similar images in an image database using a relational matrix by J. Tang; S.T. Acton; D.P. Mukherjee[https://ieeexplore.ieee.org/document/1187309] - This work presents a method for image similarity measurement based on relational matrices, contributing to the field of content-based image retrieval.

Results

Input 1

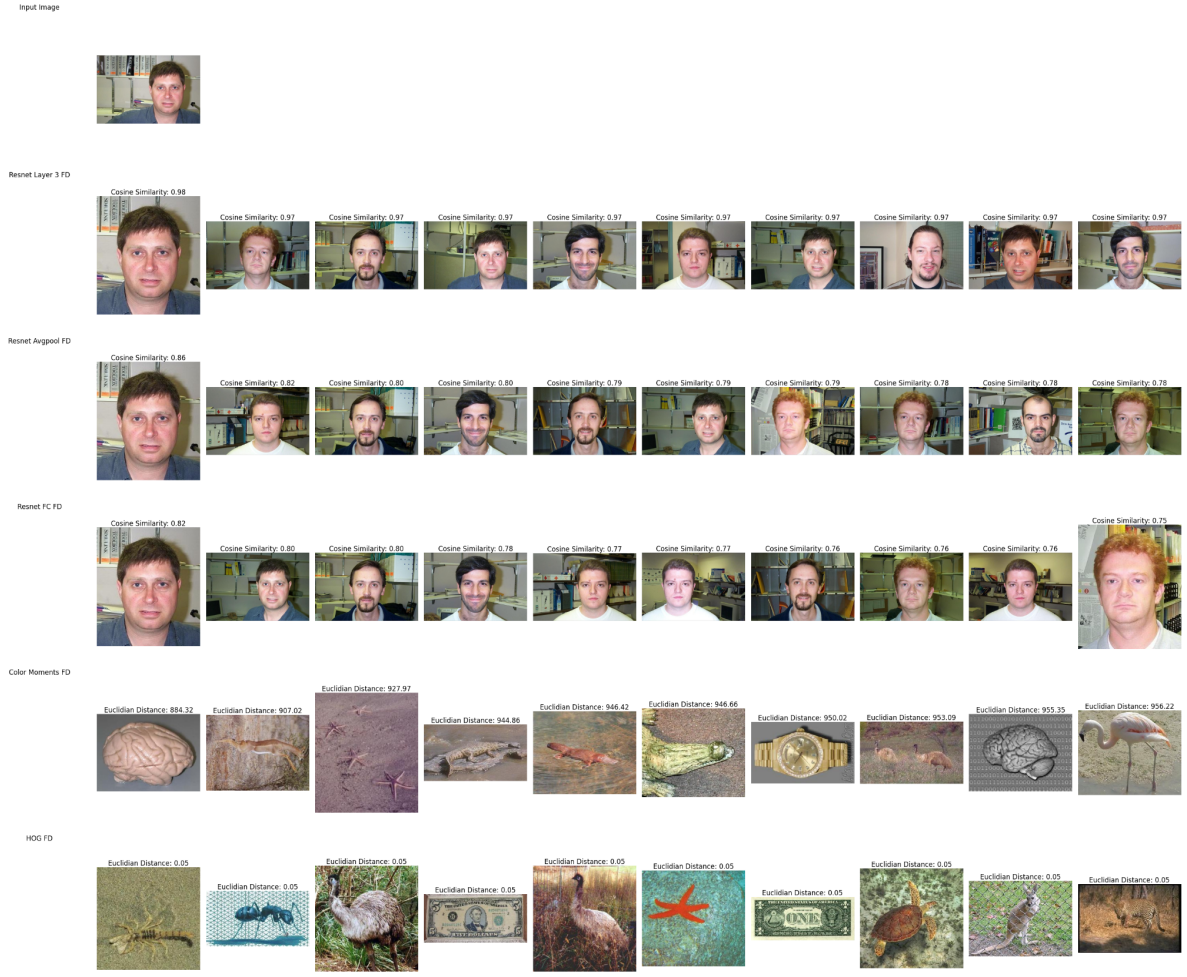


Image ID: 0
Explanation:
Resnet features: All output images are similar.
Color moments: Since the input image contains a wide range of colors where the face is just a small part, the output image doesn't exactly contain faces but objects with similar color distribution. For instance, the color composition of the brain is quite similar to that of skin.
HOG: The input image is a bit zoomed out and the face is just in one section of the image. HOG performs poorly as it is only able to recognize low level features like edges and shapes computed locally on each block of the grid. The output images must have similar edges for the local blocks.

Input 2

Image ID: 880
Resnet features: All output images are similar.
Color moments: All output images are similar to the input image and have similar color composition.
HOG: All output images are similar based on shape. Animals like hippo and deer are also included in the output based on edge and shape detection as HOG works on grayscale images and detects edges and shapes only.

Input 3

Image ID: 2500
Explanation:
Resnet features: All output images are similar.
Color moments: All images are similar to the input image except for the bird. Since, color moments calculate the similarity of image based on color composition we can say both images are similar as they contain white and blue color primarily.
HOG: All output images are similar, as the plane object is in the major part of the image.

Input 4

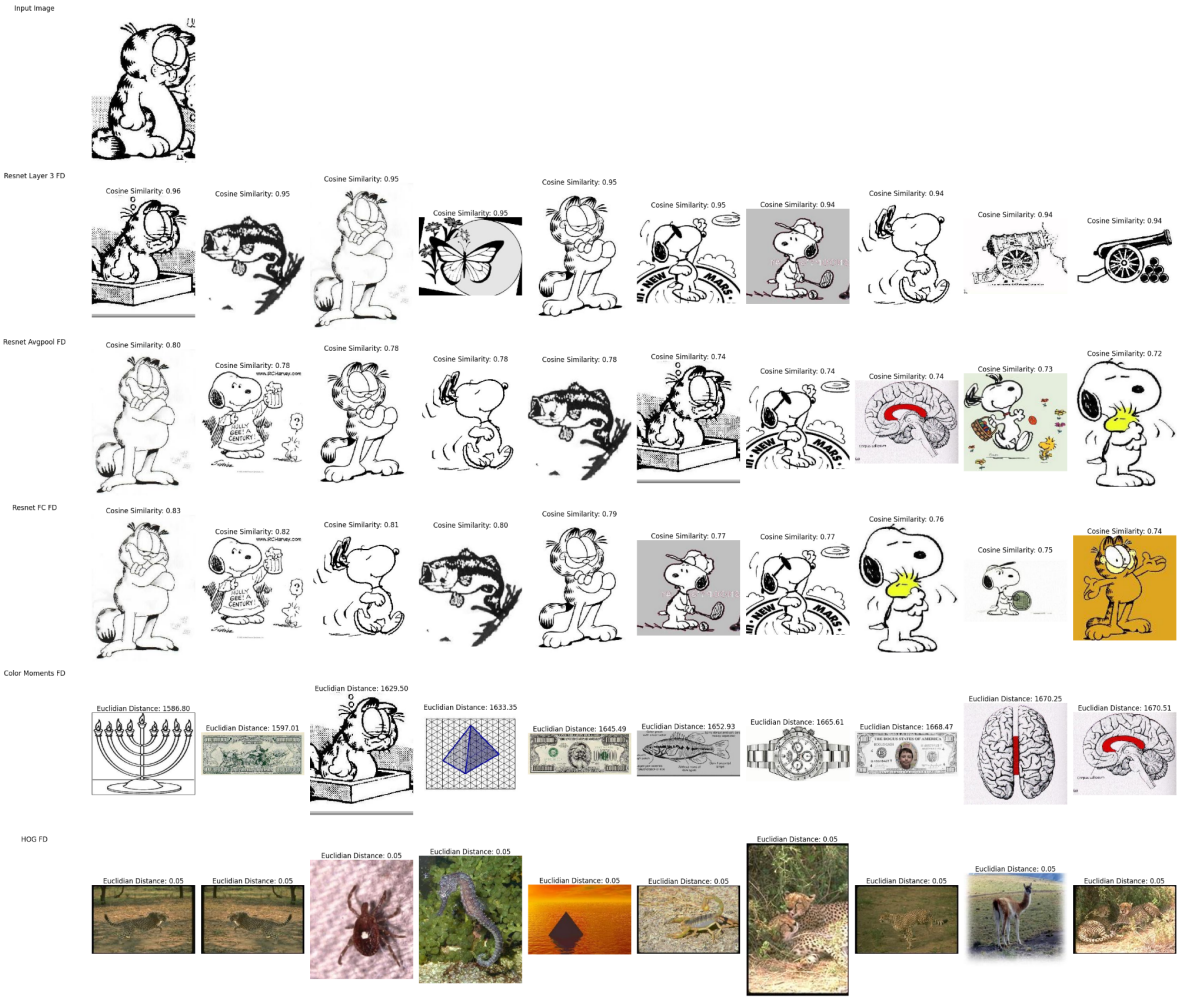


Image ID: 5122
Explanation:
Resnet features: output images are similar.
Color moments: Since the input image is grayscale, all the images in the output that are grayscale can be treated as similar on the basis of color composition.
HOG: Since, HOG relies on the distribution of oriented gradients, It performs poorly on images that only have black outlines as such images may not contain enough texture or detail for HOG to capture significant variations in gradients. The images in output must have similar edges for local blocks in the grid, hence these images are outputted.

Input 5

Image ID: 8676
Resnet features: All output images are similar.
Color moments: All the images in output have the same color composition of black and white including chairs.
HOG: Since, HOG features are helpful in edge detection and shape detection, all the images in the output have similar shape and hence are similar by the definition of shape.

Conclusions

In this project, I worked on extracting features from images using ResNet 50 layers (specifically, Layer 3, Avgpool, and FC), Color Moments, and Histogram of Oriented Vectors (HOG). My goal was to create meaningful descriptions for each image in the Caltech101 dataset and then use these features to find similar images.

Through this project, I gained insights into the performance of these methods. It became clear that HOG and Color Moments have limitations in handling images of varying complexity. HOG struggled with cluttered or highly zoomed images because it calculates edges and shapes locally for each block in the image grid. Additionally, HOG didn't perform well with images featuring black outlines or lacking detailed texture.

In summary, this project highlighted the strengths and weaknesses of different feature extraction techniques. While ResNet 50 layers demonstrated versatility in capturing complex image features, HOG and Color Moments showed context-dependent performance. Understanding these nuances is essential for building effective image retrieval systems, particularly when dealing with diverse image datasets.

Choosing similarity/distance measure for feature vectors:

1. Cosine Similarity for ResNet Features:
   ResNet features, particularly those extracted from deeper layers like Layer 3, tend to represent high-level, abstract image features. These features are often characterized by their orientation or direction within the feature space. Cosine similarity is well-suited for comparing such vectors because it measures the cosine of the angle between them. It captures the similarity in direction, making it effective for high-dimensional vectors where the magnitude of the vectors might vary. Since ResNet features are typically represented as high-dimensional vectors, cosine similarity helps in focusing on the orientation of feature vectors rather than their magnitudes.

2. Euclidean Distance for Color Moments and HOG:
   Color Moments and HOG features represent lower-level image characteristics like color statistics and local gradient patterns. Hence, euclidean distance is a good measure for comparing these vectors because it calculates the straight-line distance between two points in the feature space. It emphasizes the magnitude and absolute differences between feature values. For Color Moments, which represent statistical moments of color distributions, Euclidean distance can effectively measure differences in color content. HOG, which captures local gradient orientations, can also benefit from Euclidean distance because it accounts for variations in gradient values

Bibliography

1. Wikipedia contributors. "Colour moments." Wikipedia, The Free Encyclopedia(https://en.wikipedia.org/wiki/Color_moments).
2. Wikipedia contributors. "Histogram of oriented gradients." Wikipedia, The Free Encyclopedia(https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients).
3. Kaggle user pankajgiri. "ResNet Feature Extraction - PyTorch." Kaggle(https://www.kaggle.com/code/pankajgiri/resnet-feature-extraction-pytorch).
4. DataGen Tech. "Resnet 50." DataGen Tech Guides(https://datagen.tech/guides/computer-vision/resnet-50/).

Appendix

Ariana Bui - Worked on the project individually
Kasi Chikkala - Worked on the project individually
Maryam Cheema - Worked on the project individually
Peter Kim - Worked on the project individually
Rajat Pawar - Worked on the project individually

Yash Dhamecha - Worked on the project individually