Project Phase 1

CSE515 - Multimedia and Web Databases

Group 17
Abhinav Gorantla
Rohan Samuel Gangavarapu
Krishna Siddhardh Potluri
Ram Abhishek Ramadoss Sivadoss
Hrigvid Madan Sangle
Sensen Wang

Table of Contents

Abstract

In the first phase of this project, we will be experimenting with various distance measures and vector models. A major part of this involves generating image features and extracting images using those features. We will be analysing which distance/similarity metrics are suitable in which specific cases. We will be analysing which distance metric is useful for what kind of features and other things like the impact of scaling down or scaling up certain feature values.

Introduction

As a part of this introductory project to CSE515 Multimedia and Web Databases, the task given to the class was to extract image features from Caltech101 dataset, store those features in a database management system and perform image retrieval using those features.

Goal Description

Three milestones in the phase 1 of this project are:

1. Being able to print out the five image features (Color moment, histogram of gradients and layer3, avgpool and fc layer outputs of the ResNet50 model available in the torchvision package).
2. Storing features of all the images in the Caltech101 dataset.
3. Retrieval of top k images similar to a given image given the ID of that image.

Assumptions

We made the following assumptions when building this phase of the project:

1. All the images in the provided dataset have three channels (R, G, B). Any image which does not have all three channels have been ignored.
2. We are being provided with a pretrained Resnet50 model with default weights.

Proposed Solution

Description

One of the important decisions I had to take when designing a solution for this phase was to select a database. I chose mongoDB for the simplicity and ease of use it has. But later I realised “ease of use” comes with some drawbacks. This is explained in the next section, design decisions. Once I selected the database, i.e. MongoDB, I installed the database client locally.

To calculate color moments, I used basic mathematical operations to calculate the mean, standard deviation and skewness and used for loops to iterate through the image array. I followed a similar approach for HoG. When calculating Histogram of Gradients, I first padded the image with a black border, so that the corner pixels of the image do not get lost when calculating the gradients for them. For layer3, avgpool and fc (fully connected) outputs of the Resnet50 model, I used hooks to extract them.I utilized hooks to extract the outputs of layers 3, avgpool, and fc (fully connected) from the Resnet50 model in torchvision.

For the distance measures, I chose to go with Euclidean distance for all the features except fc as it was giving satisfactiry results for all the query images provided for Phase 1. For the fully connected layer I experimented with cosine similarity, earth movers' distance and also euclidean distance. From trial and error, I found out that cosine similarity worked best for FC.

Implementation

We implemented this project in modules. Every feature had its own function to extract that specific feature from the image, this reduced the number of lines of code in each file and helped with better resolution of errors when we had any. We also used the opencv-python, pymongo, numpy and scipy packages in addition to the ones which were provided in the project specification document. I used opencv-python to resize images and pymongo was used to connect to the mongo database and perform insertion and search operations. I used scipy package for its distance metric formulae.

Design Decisions

1. Why MongoDB?

   The main reason I chose to use mongoDB is because of the ease of use it has. But as I progressed through task 2 and 3, I realised an SQL database would have better suited for this application. This is mainly because given the large size of the features collection in the mongoDB, mongoDB Compass GUI was frequently crashing. I also felt that MongoDB is not able to handle cache properly.
2. Why local database?

   I tried using the cloud based version of MongoDB (Mongo Atlas), but the space provided by the free tier was not sufficient for features of all the 8677 images in the Caltech101 dataset. This was one of the major reasons I started using local mongoDB. I also felt it will be faster for queries to get processed when there is no internet involved, hence it will enable faster testing.
3. Why did I not use numpy functions to calculate mean, etc?

   I wanted my code to be simple and easy to edit whenever I wanted to make changes.In the beginning of the project, I also had some difficult trying to think

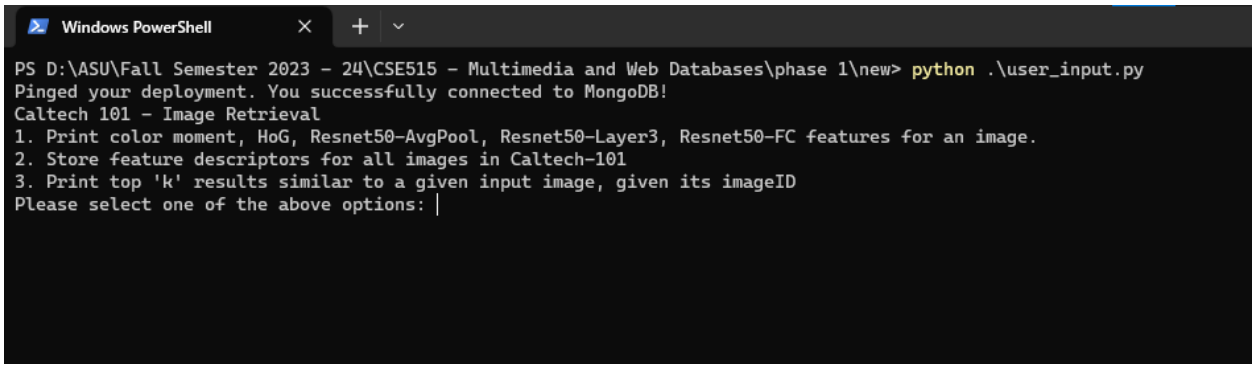
and make sense of the multidimensional image array. These were the reasons I did not use any inbuilt functions when trying to calculate color moments and Histogram of Gradients.

Key Challenges Faced

1. One of the major mistakes I made when starting off with this project was not loading the dataset from torchvision and using cv2 to load images. Later I changed my approach to load the dataset from torchvision, this also simplified the code significantly.
2. Another difficulty I faced during this project was trying to make sense of what image formats are good for which scenarios. In general I found it is better to load the images as a Tensor or a numpy array.
3. I could not store the image or the features as a tensor or a numpy array in the mongo database. I still have not figured out a way to do so. Finding a way to store tensors and numpy arrays in mongoDB will help reduce the code to perform certain tasks in the project.
4. Understanding the computation of HoG: Initially, I had difficult in understanding how to compute HoG. But with the TAs help, I could figure out how HoG computation works for an image and successfully implement it.

System Interface and Outputs

1. User Interface



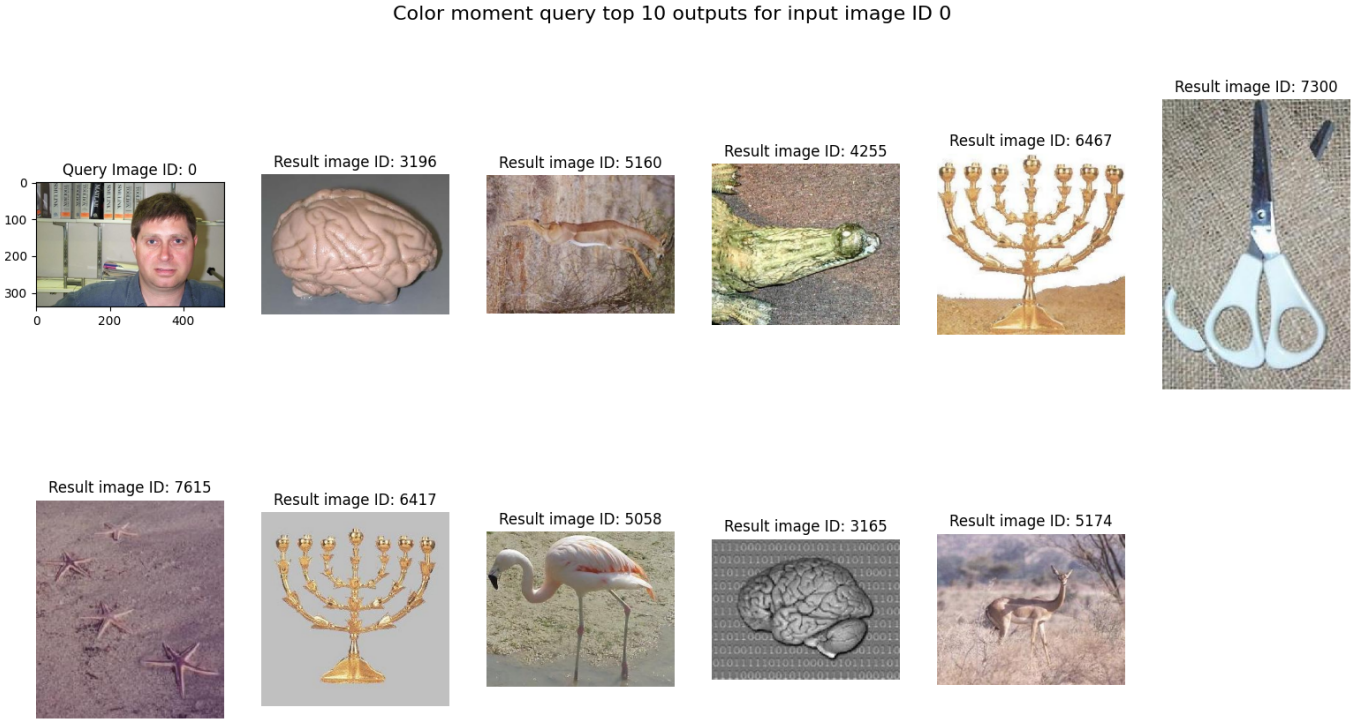
2. Image query example:

```
PS D:\ASU\Fall Semester 2023 - 24\CSE515 - Multimedia and Web Databases\phase 1\new> python .\user_input.py
Pinged your deployment. You successfully connected to MongoDB!
Caltech 101 - Image Retrieval
1. Print color moment, HoG, Resnet50-AvgPool, Resnet50-Layer3, Resnet50-FC features for an image.
2. Store feature descriptors for all images in Caltech-101
3. Print top 'k' results similar to a given input image, given its imageID
Please select one of the above options: 3
Enter image ID: 880
How many similar images have to be returned10
```

3. Query Outputs

a. Image ID - 0

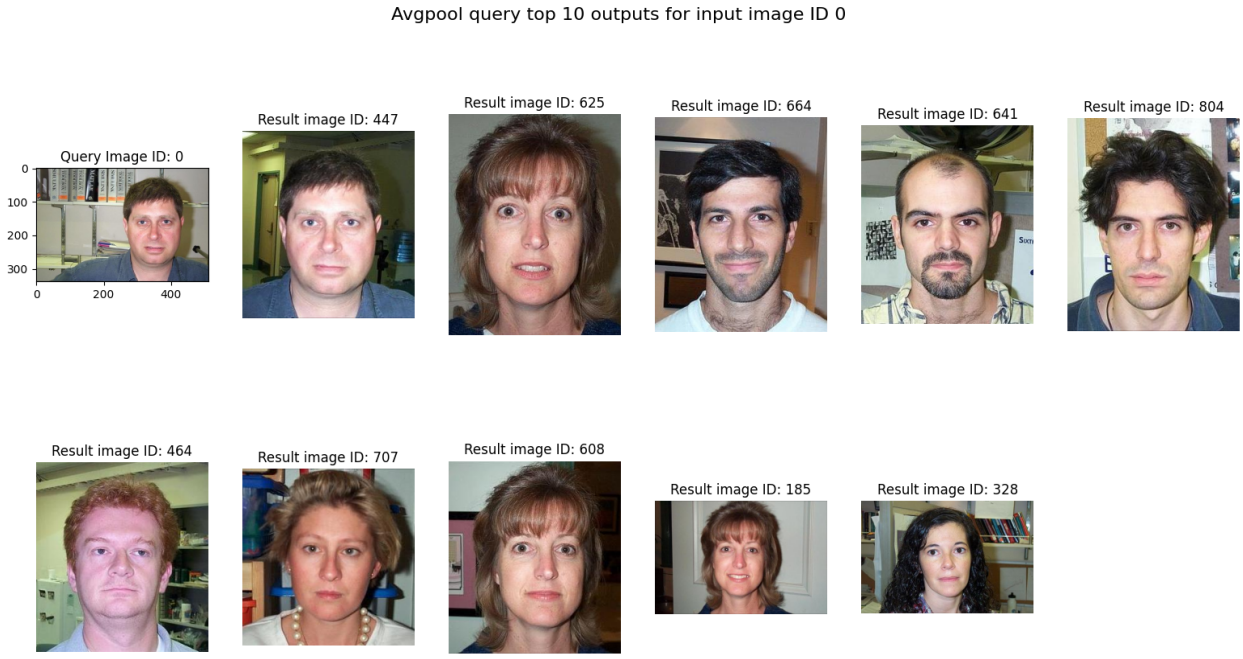
i. Color Moment



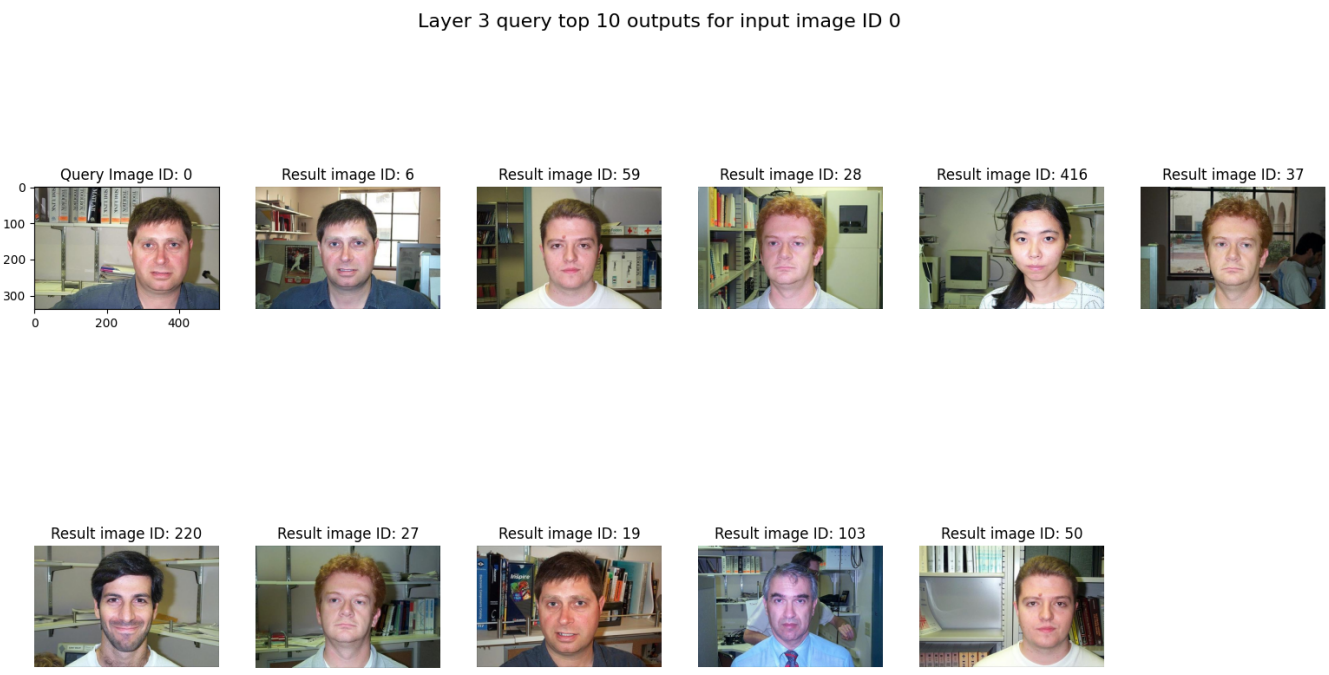
ii. HoG

HoG query top 10 outputs for input image ID 0

iii. AvgPool

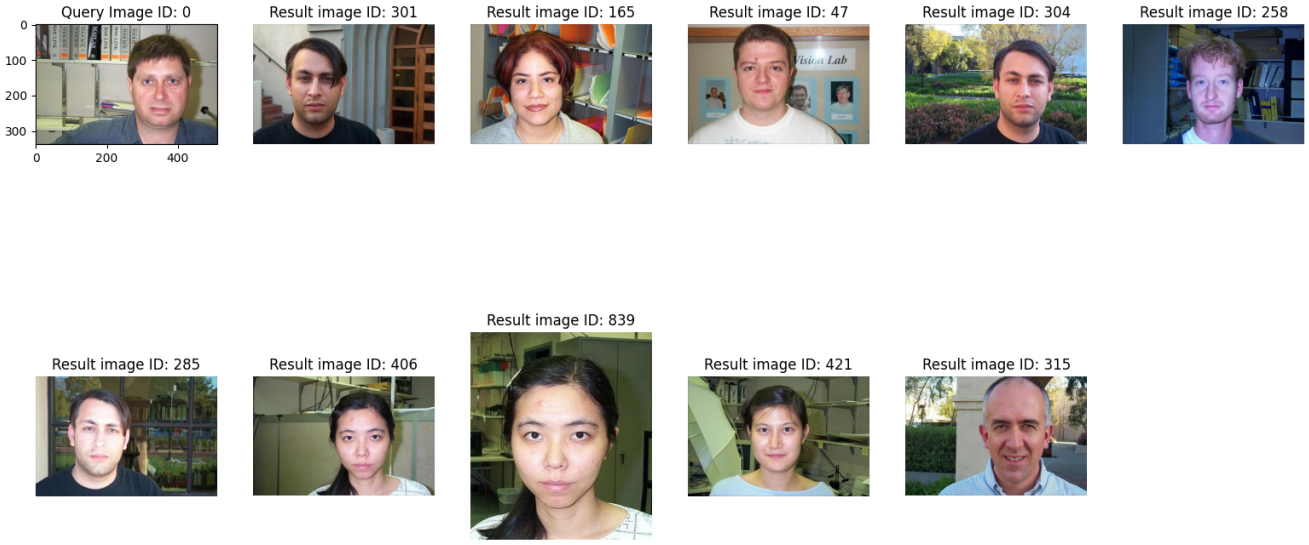


iv. Layer 3



v. FC

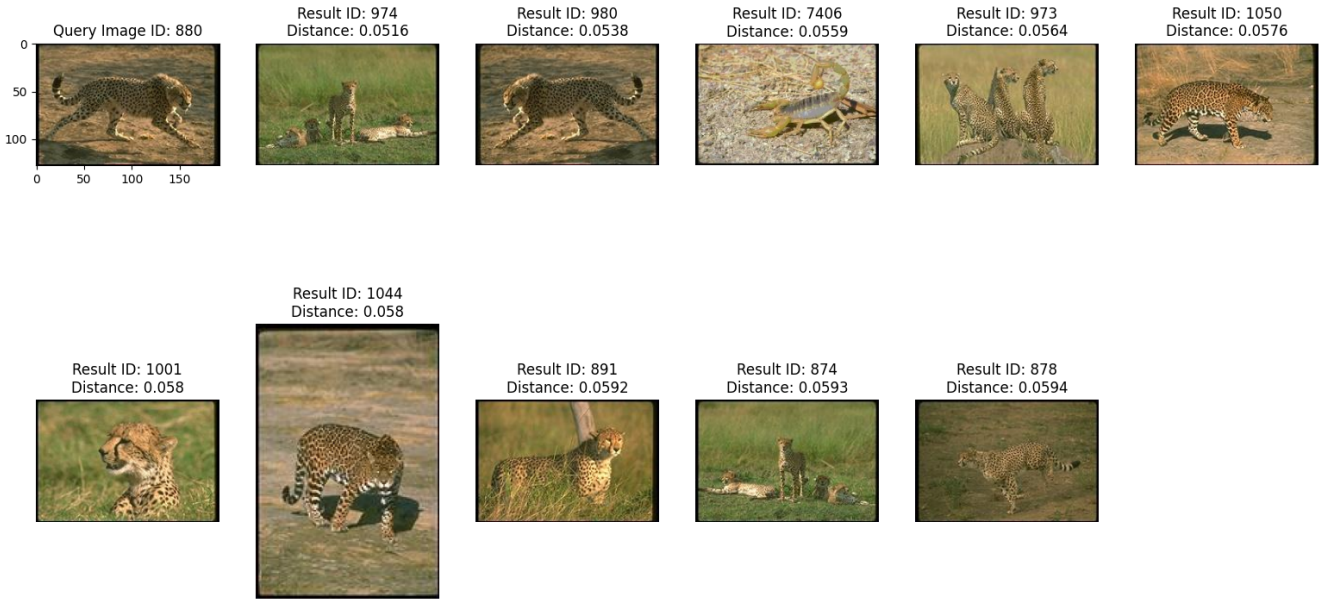
FC query top 10 outputs for input image ID 0



b. Image ID - 880

i. Color Moment

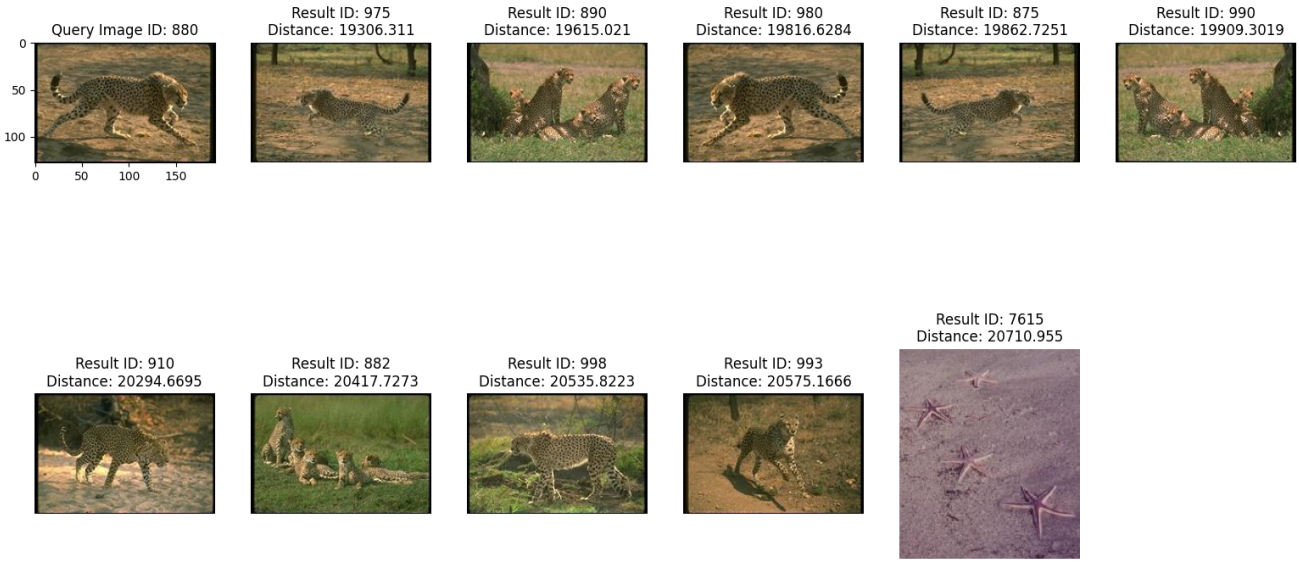
Color moment query top 10 outputs for input image ID 880



ii. Histogram of Gradients

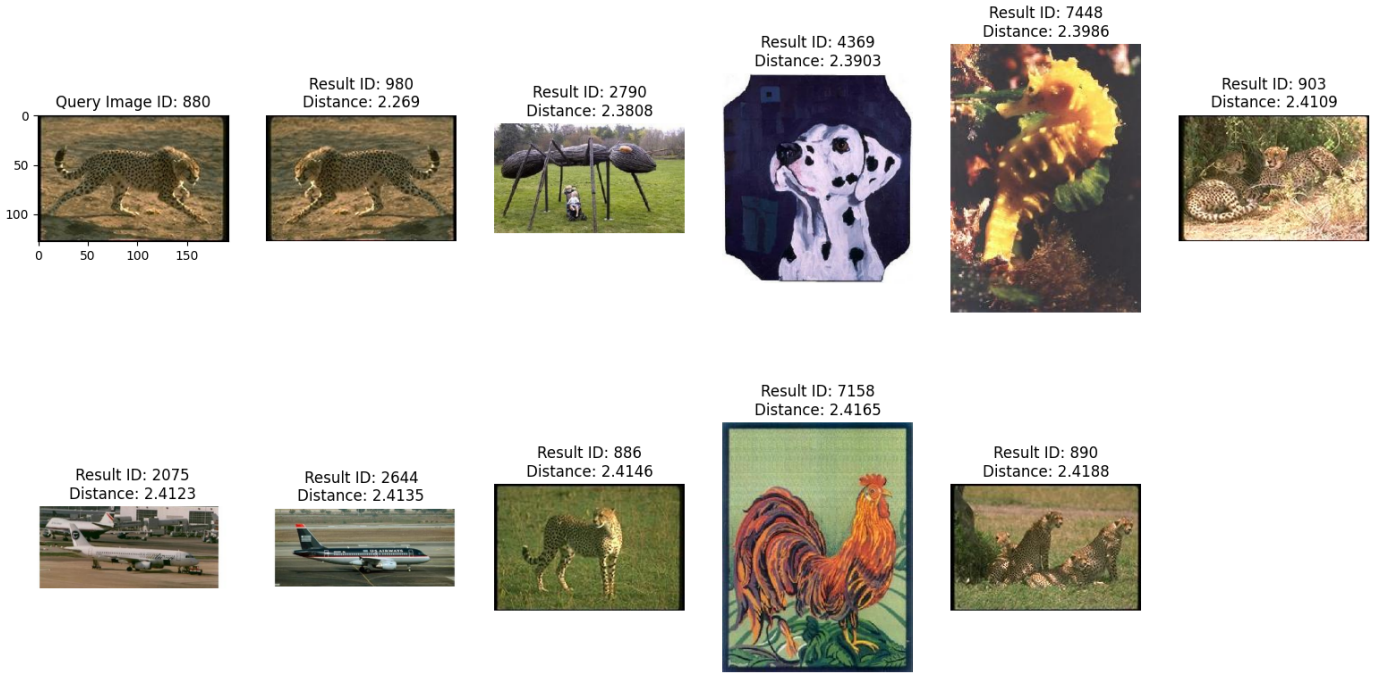
\

HoG query top 10 outputs for input image ID 880



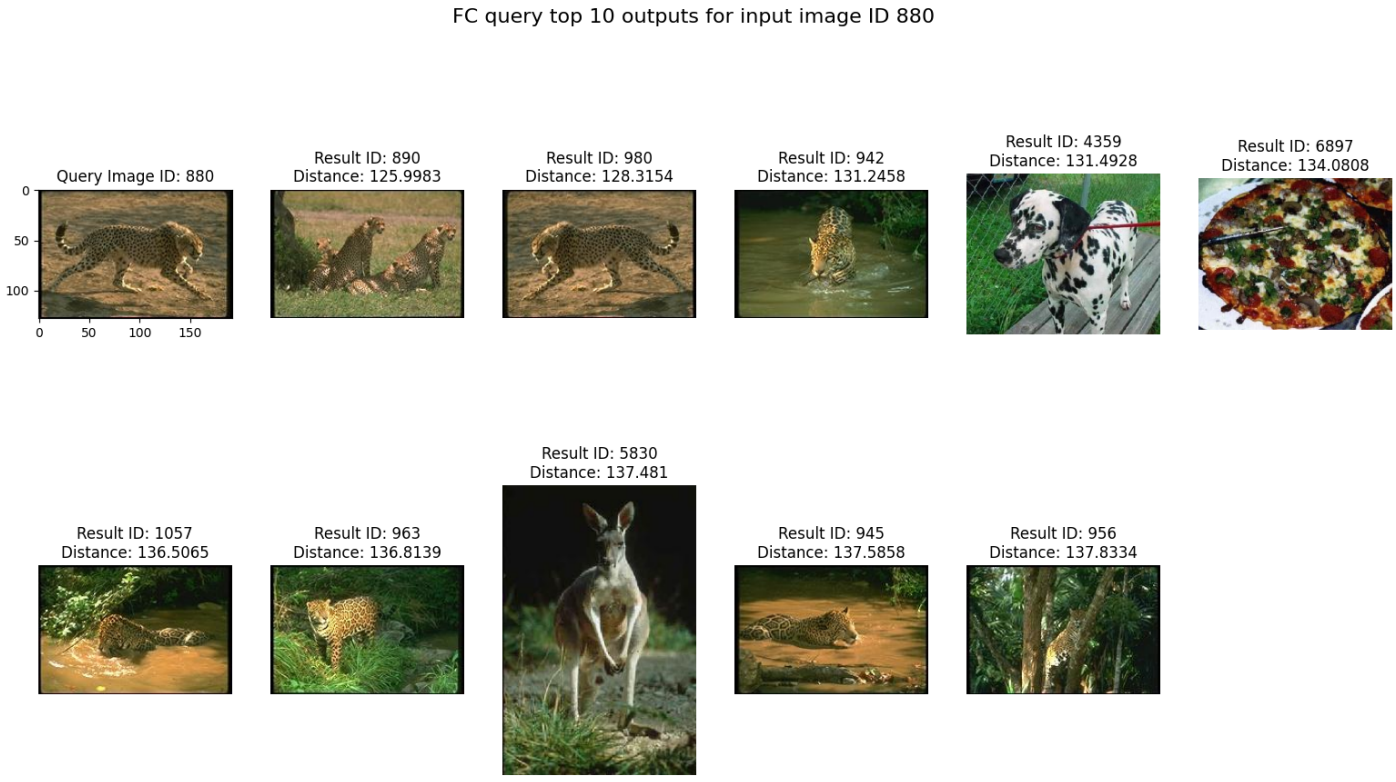
iii. AvgPool

Avgpool query top 10 outputs for input image ID 880



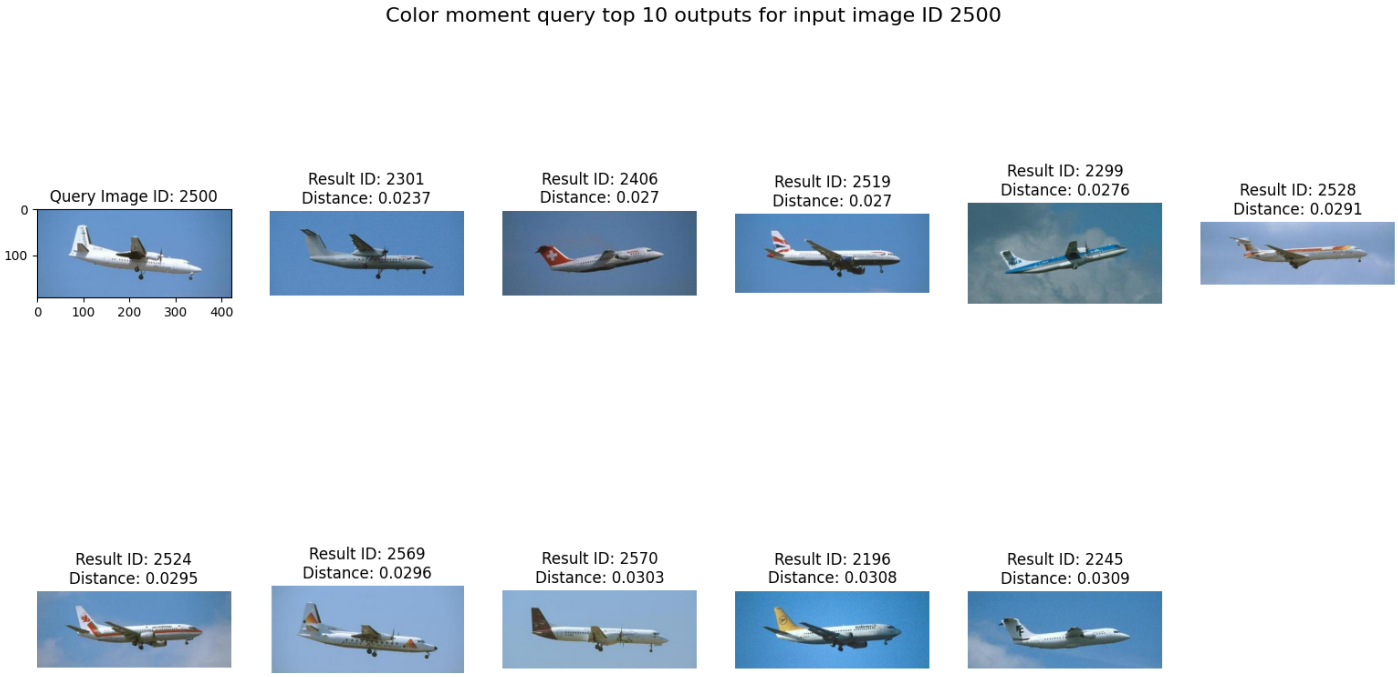
iv. Layer3

v. FC

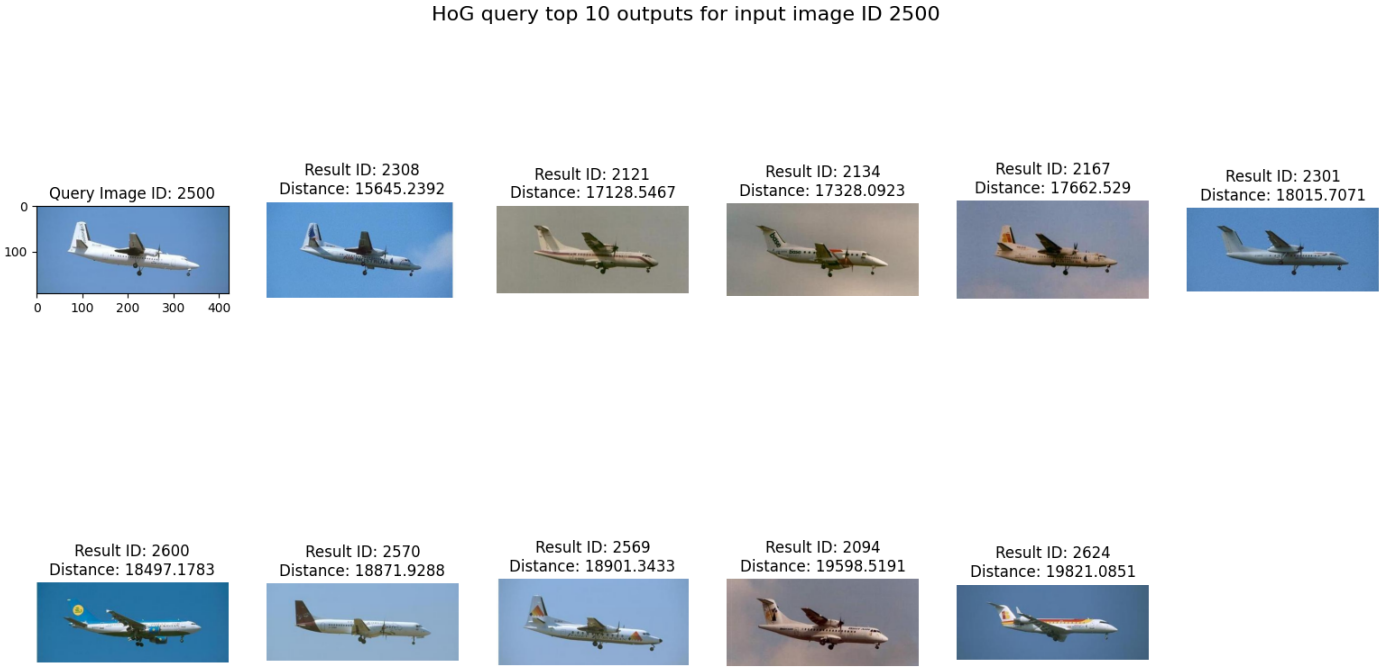


c. Image ID - 2500

i. Color Moment



ii. Histogram of Gradients



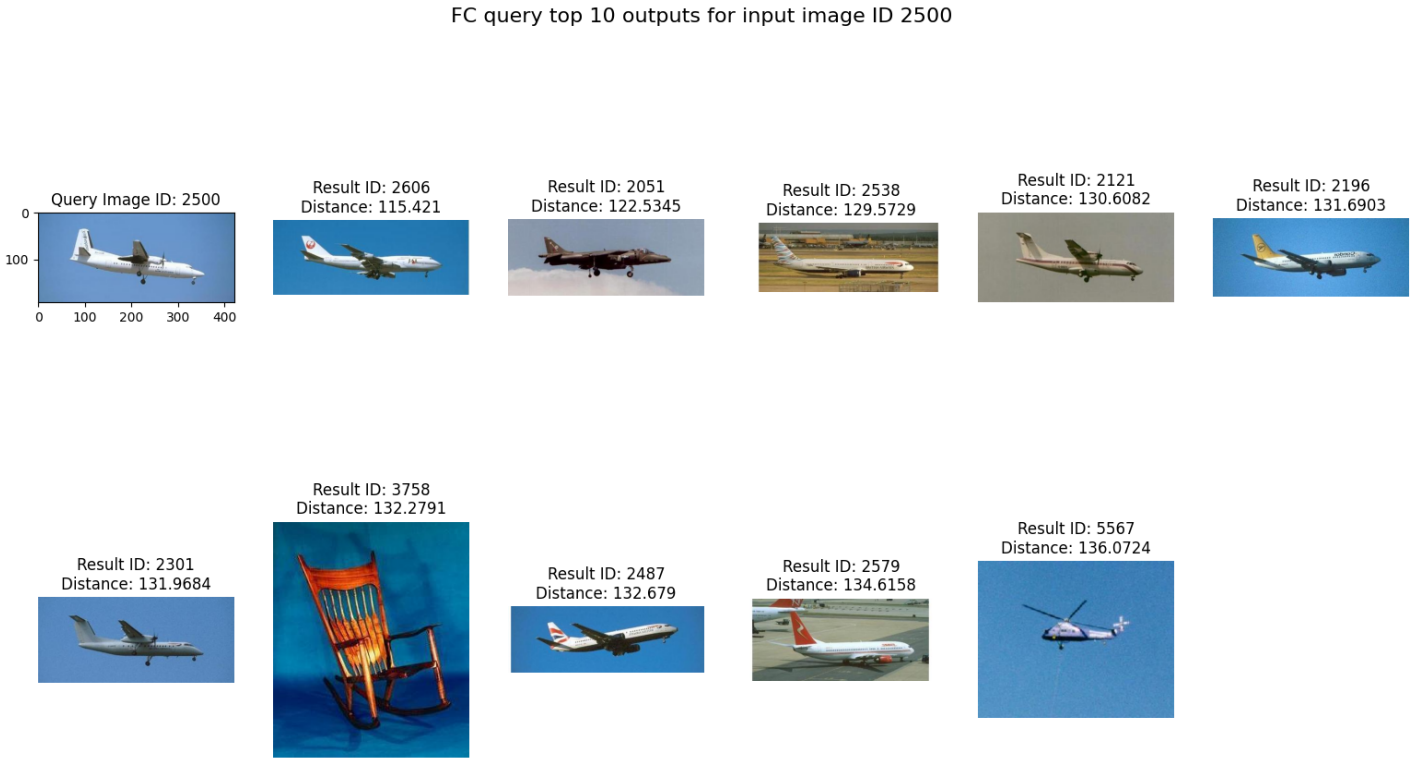
iii. AvgPool

Avgpool query top 10 outputs for input image ID 2500

iv. Layer3

Layer 3 query top 10 outputs for input image ID 2500

v. FC



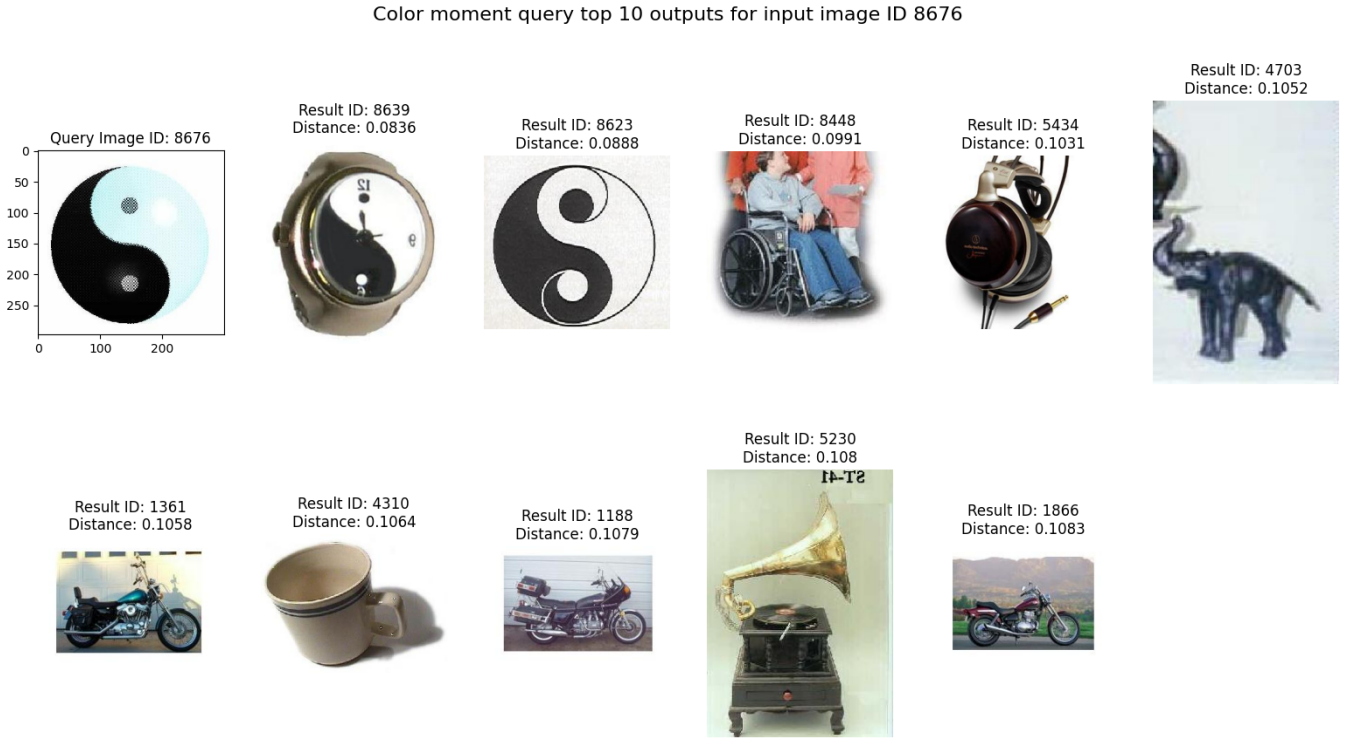
d. Image ID - 5122

This image had only one channel (it was a grayscale image). I ignored grayscale images when generating features for this project. Resnet50 model was not accepting images with only one color channel.

e. Image ID - 8676

i. Color Moment

The output images for the yin yang input image 8676 all come have a part of the image colored darker than the other half. This is quite evident in images 8448, 5434, 5230, 4703 and 4310. The color moments show the spatial distribution of color values in an image and this is clearly similar for all the images returned in the query result.



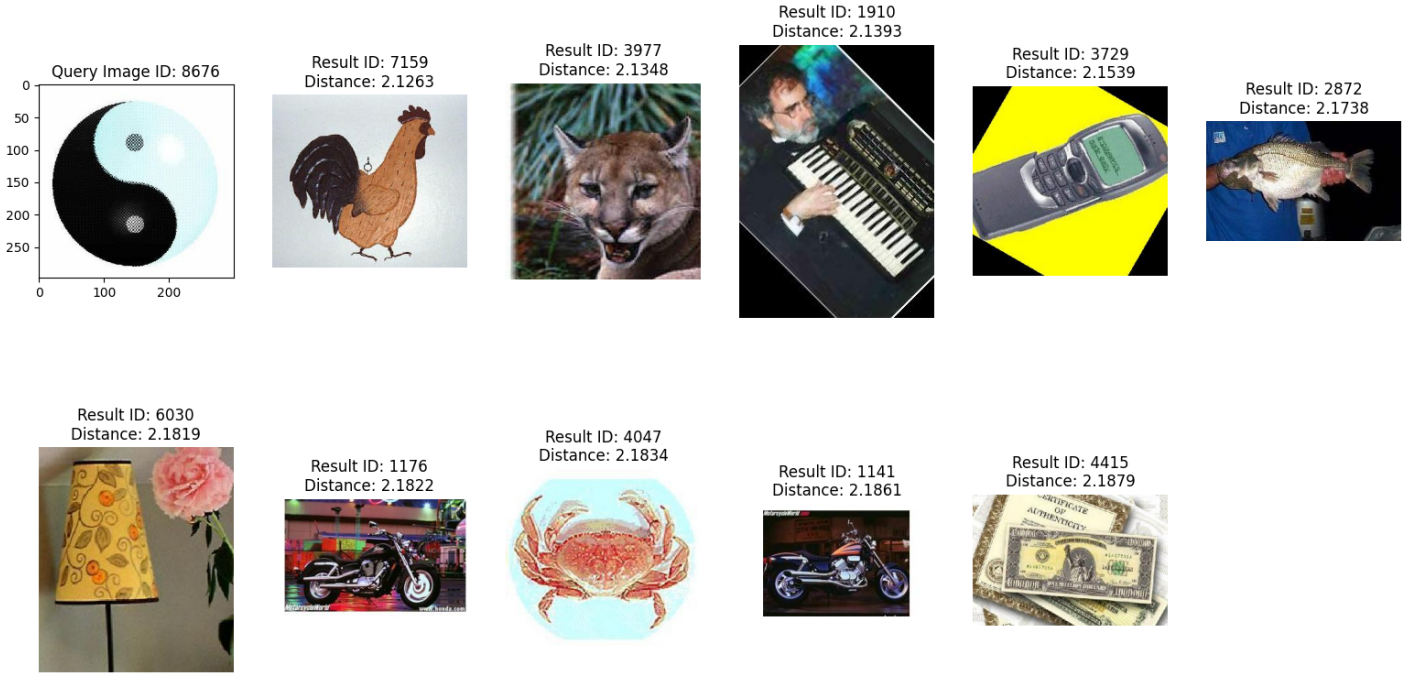
ii. Histogram of Gradients

Histogram of Gradients basically allows us to find images which have similar texture. It is also used in situations where we want images which have similar shapes as the input image. The texture (in this case, “S”/”C” shape curves) similarity is clearly seen in image 6614, 6607 and 6457. In image 8244, we can see that the watch has some black circles on its dial, it could be because of this reason that the watch image was returned as the result to the yin and yang input query image.

HoG query top 10 outputs for input image ID 8676

iii. AvgPool

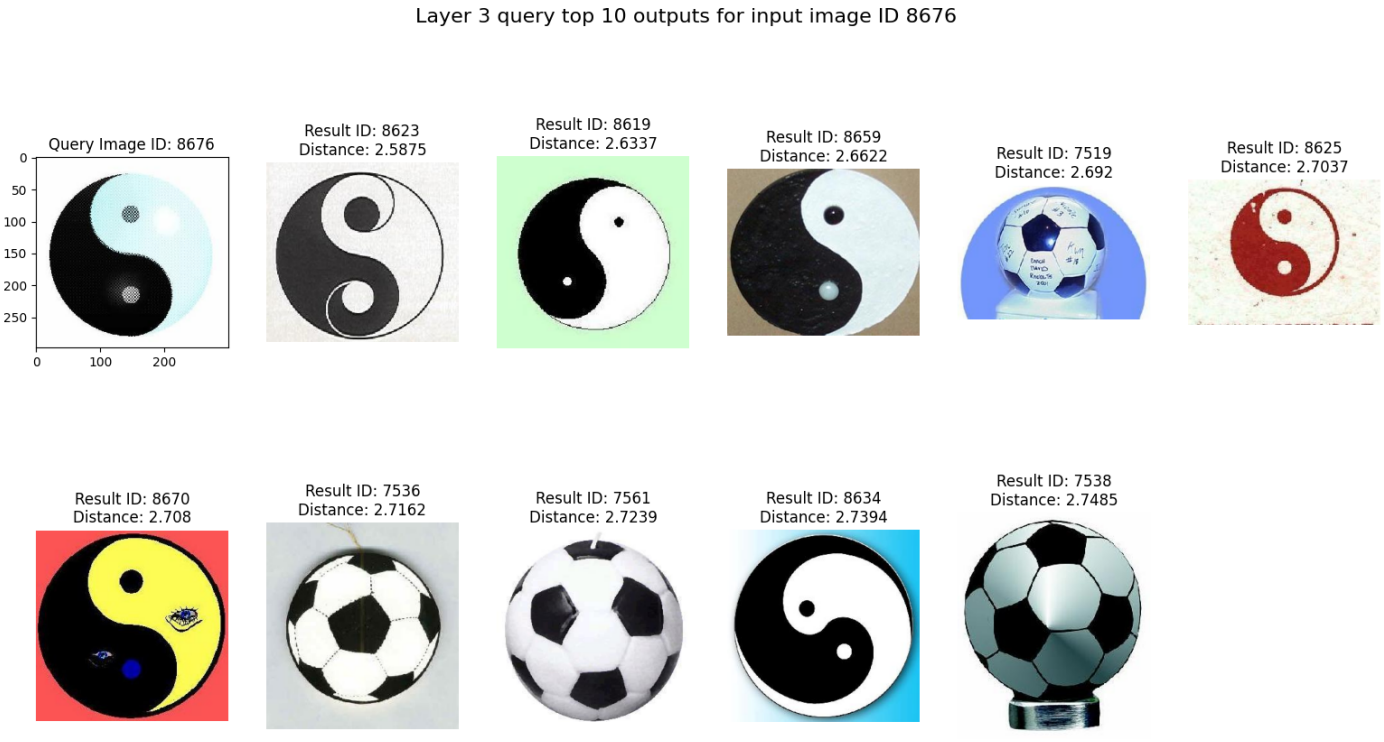
Avgpool query top 10 outputs for input image ID 8676



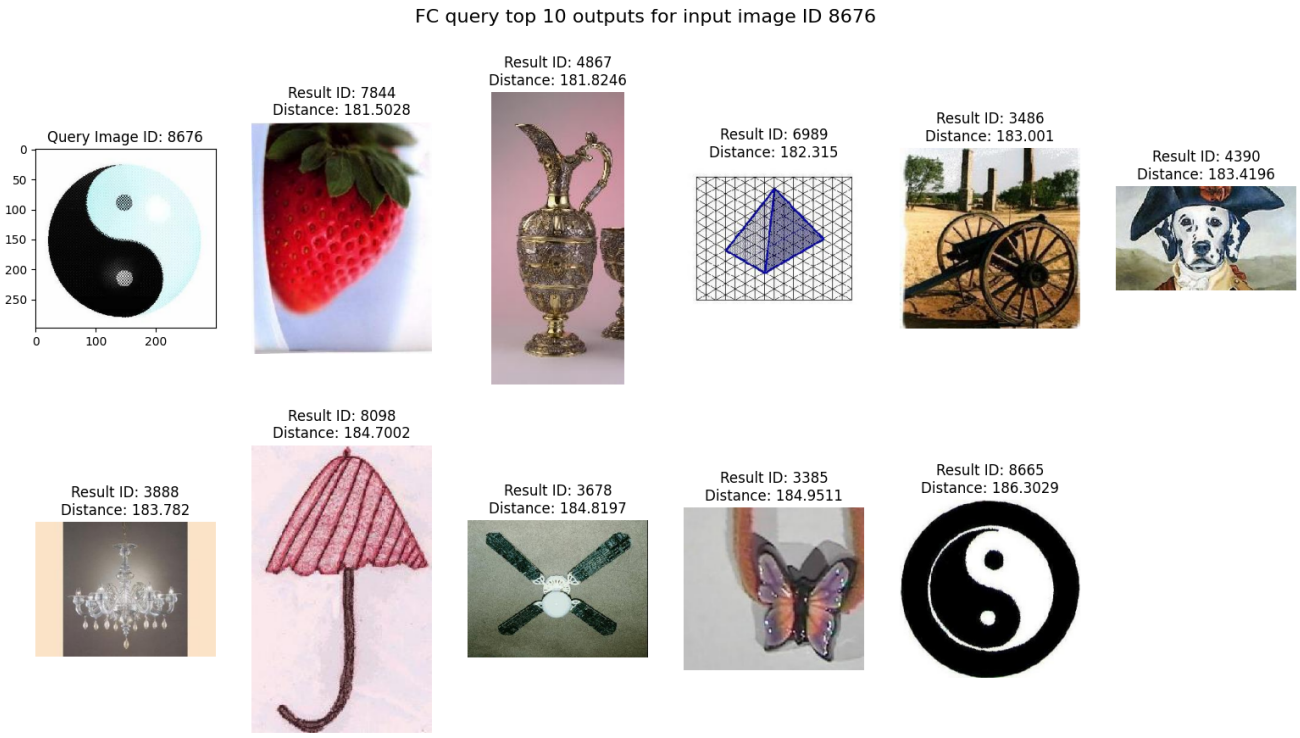
iv. Layer3

6/10 of the layer3 feature outputs for the the yin and yang symbol are yin and yang symbols. But not all of them are in black and white colors. From

this we can maybe make a conclusion that, layer3 features are **color invariant**. 4/10 of th eoutput images are soccer balls, these have a similar composition of colors as the yin and yang symbol, but the spatial distribution of the color is different. Maybe from these two observations we can say that layer3 looks at a single channel image and does not take into account the spatial properties of the image when generating the features.



v. FC



System requirements

Software and hardware requirements to run this project are:

1. A 64 bit operating system (code has been tested on Windows 10 and Linux Mint 21.2)
2. Python 3.11 or later
3. Mongo compass
4. Mongo Shell (mongosh)

To run this project locally, all the python packages mentioned in the requirements.txt file have to be installed.

Execution instructions

The steps to run this project locally are:

1. Open Mongo Compass and connect to your local mongo instance.
2. Go the project directory and run ‘pip install -r requirements.txt’ to install all the required python packages for this project.
3. Run the python file “**extract_image_features.py**” to generate features for all the images in the Caltech101 dataset.

4. Once all the features get geenerated and inserted into the database, we can start querying the database. Run the python file “index.py” to query image feature data from the database.

Future Work

In the future, I plan to find a way to convert grayscale images to RGB images and incorporate even single channeled images into this project. I also plan to correct a few of the mistakes I made when designing a solution for this phase of the project such as selecting a noSQL database instead of a SQL database. My teammates who used SQL database could query data much faster. I would also like to experiment with combining the feature vectors and computing results instead of computing results for each feature vector separately.

Conclusion

In this project I got a chance various distance and similarity metrics like wasserstein distance, cosine similarity and euclidean distance. We could get good accuracy with the five features we used in this project.

Appendix

Roles of group members

The distance measures were discussed among the 4 memes of our group (Abhinav Gorantla, Rohan Samuel Gangavarapu, Krishna Siddhardh Potluri, Ram Abhishek Ramadoss Sivadoss) and decided upon collectively. For the rest of the code implementation Abhinav worked on his own without much discussion with Rohan Samuel Gangavarapu, Krishna Siddhardh Potluri and Ram Abhishek Ramadoss Sivadoss. But, Rohan Samuel Gangavarapu, Krishna Siddhardh Potluri and Ram Abhishek Ramadoss Sivadoss worked together for the code implementation.