Yunyu Guo

CS5330 Project2

Feb 2 2025

This project is to implement a content-based image retrieval system. It includes 6 feature types:

1. Center patch (7x7 RGB values)
2. Chromaticity histogram
3. Split RGB histogram
4. Color-texture histogram
5. ResNet18 features (deep learning)
6. Combined features (ResNet + Color-texture + Central RGB)

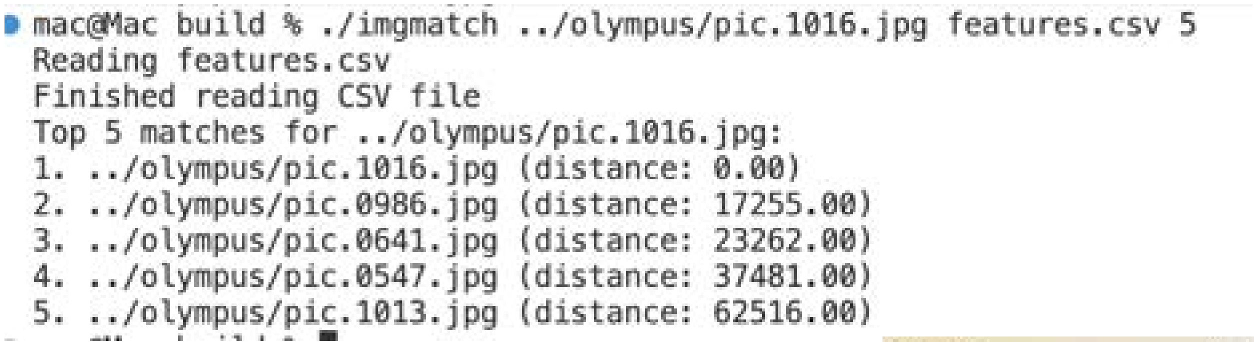
1. Baseline Matching

Use the 7x7 square in the middle of the image as a feature vector. Use sum-of-squared-difference as the distance metric.

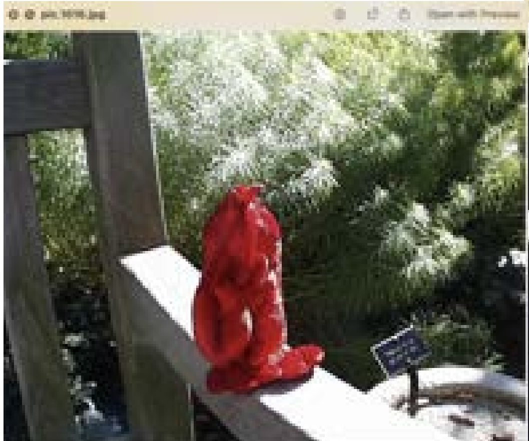
The first program is given a directory of images and feature set and it writes the feature vector for each image to a file.

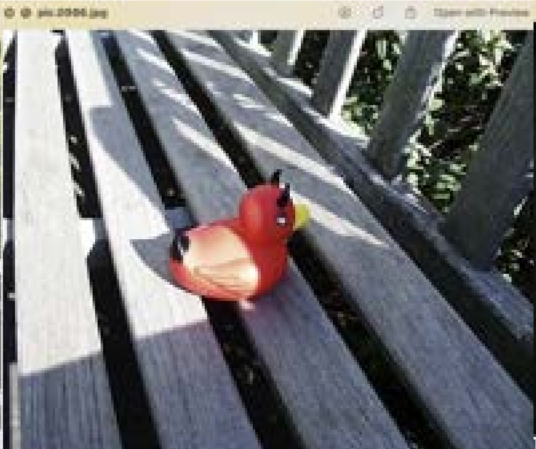
The second program is given a target image, the feature set, and the feature vector file. It then computes the features for the target image, reads the feature vector file, and identifies the top N matches.

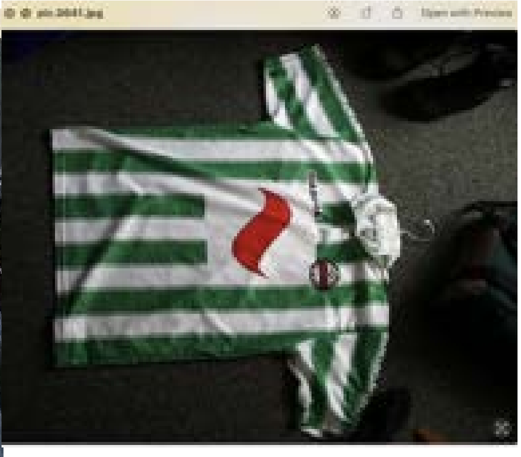
Required result 1: show the top three matches for the target image pic.1016.jpg.

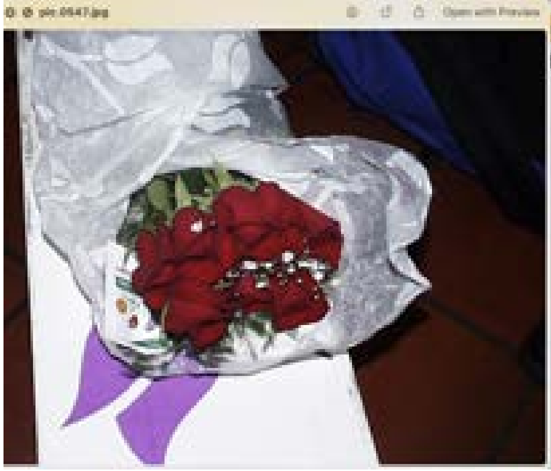


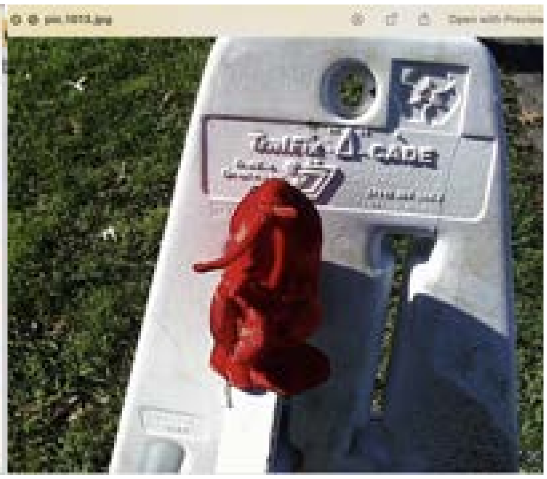
```
mac@Mac build % ./imgmatch ../olympus/pic.1016.jpg features.csv 5
Reading features.csv
Finished reading CSV file
Top 5 matches for ../olympus/pic.1016.jpg:
1. ../olympus/pic.1016.jpg (distance: 0.00)
2. ../olympus/pic.0986.jpg (distance: 17255.00)
3. ../olympus/pic.0641.jpg (distance: 23262.00)
4. ../olympus/pic.0547.jpg (distance: 37481.00)
5. ../olympus/pic.1013.jpg (distance: 62516.00)
```



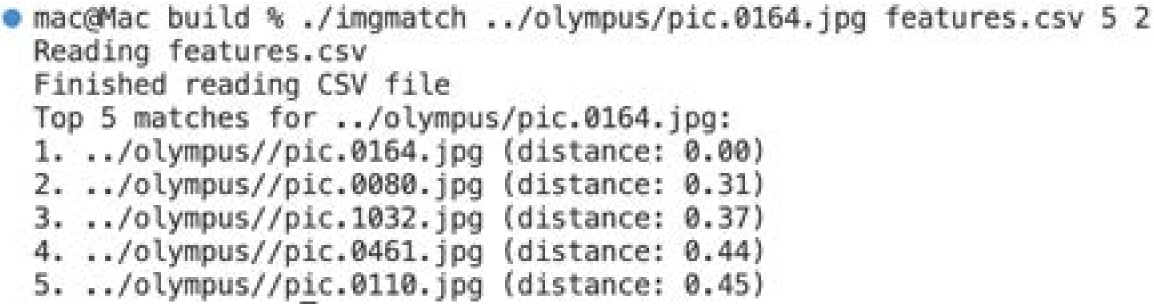




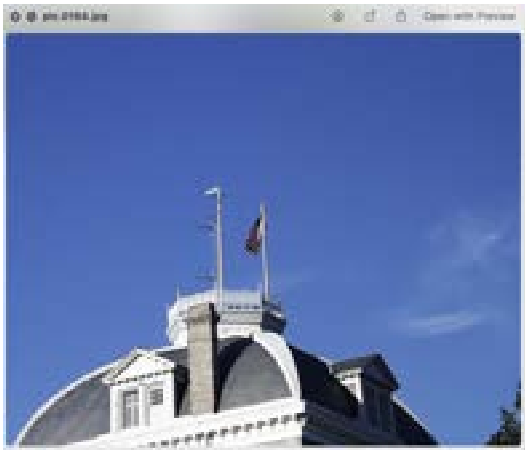


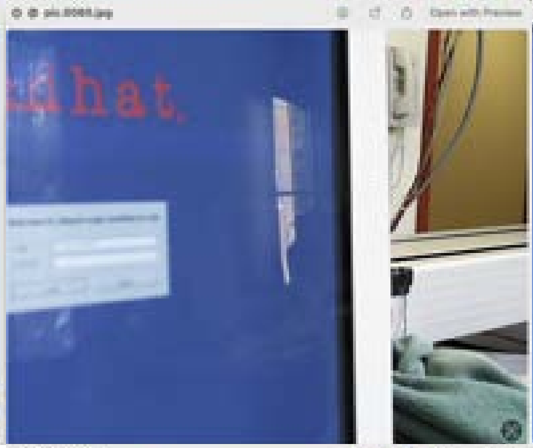


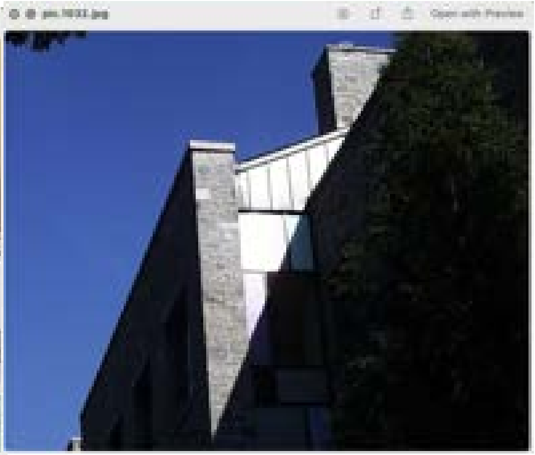
2. **Histogram Matching:**
   Required results 2: show the top three matches for the target image pic.0164.jpg.

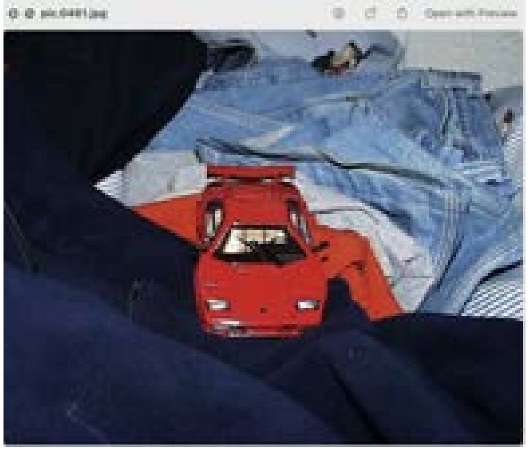


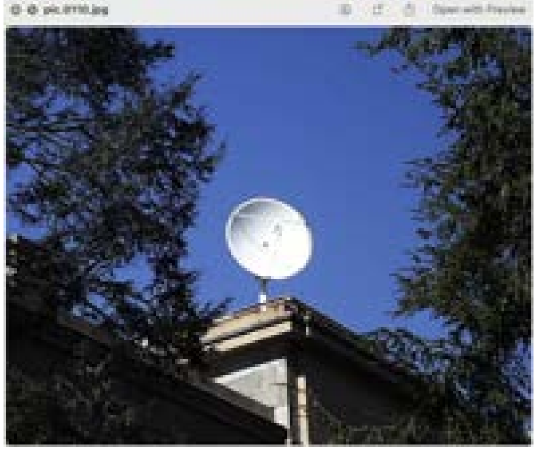
```
mac@Mac build % ./imgmatch ../olympus/pic.0164.jpg features.csv 5 2
Reading features.csv
Finished reading CSV file
Top 5 matches for ../olympus/pic.0164.jpg:
1. ../olympus//pic.0164.jpg (distance: 0.00)
2. ../olympus//pic.0080.jpg (distance: 0.31)
3. ../olympus//pic.1032.jpg (distance: 0.37)
4. ../olympus//pic.0461.jpg (distance: 0.44)
5. ../olympus//pic.0110.jpg (distance: 0.45)
```







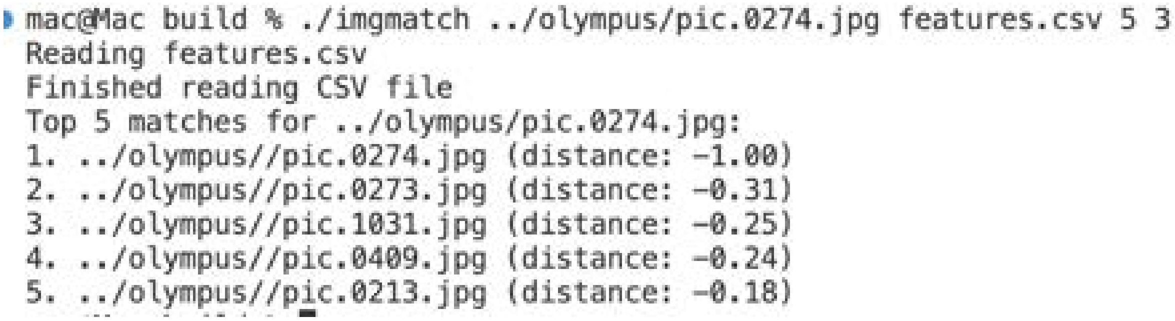




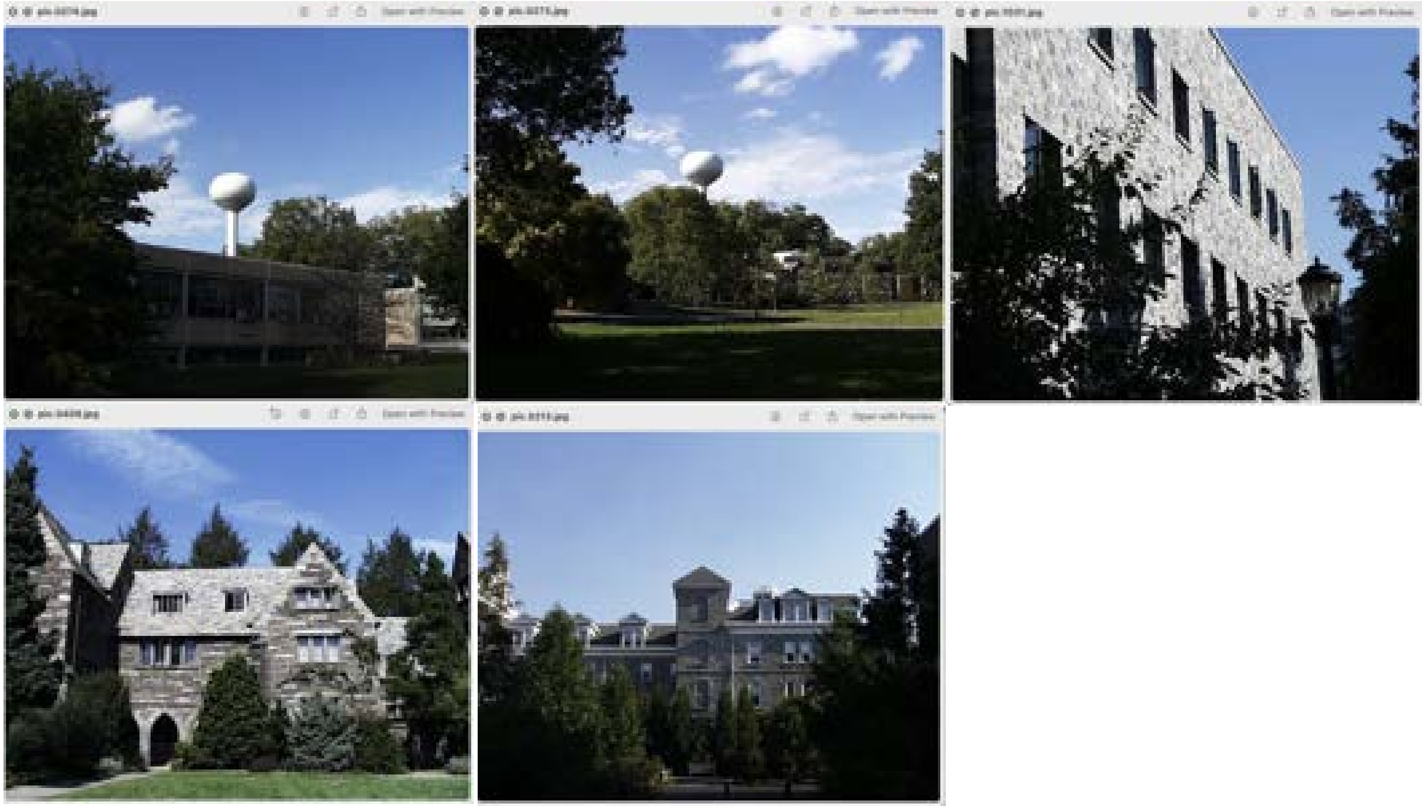
The results above for query pic.0164 were obtained with a whole image rg chromaticity histogram using 16 bins for each of r and g and histogram intersection as the distance metric.

3. **Multi-histogram Matching:**
   Required results 3: show the top three matches for the target image pic.0274.jpg.

   These images were obtained with two RGB histograms, representing the top and bottom halves of the image, using 8 bins for each of RGB, histogram intersection as the distance metric, and the histograms were equally weighted.

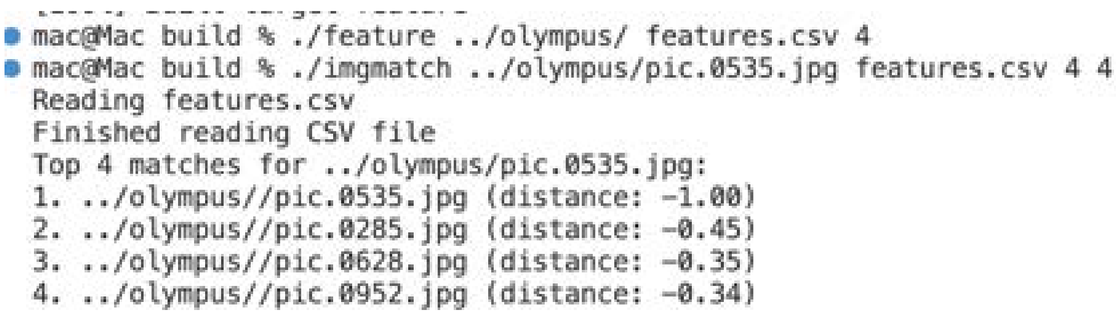


```
mac@Mac build % ./imgmatch ../olympus/pic.0274.jpg features.csv 5 3
Reading features.csv
Finished reading CSV file
Top 5 matches for ../olympus/pic.0274.jpg:
1. ../olympus//pic.0274.jpg (distance: -1.00)
2. ../olympus//pic.0273.jpg (distance: -0.31)
3. ../olympus//pic.1031.jpg (distance: -0.25)
4. ../olympus//pic.0409.jpg (distance: -0.24)
5. ../olympus//pic.0213.jpg (distance: -0.18)
```

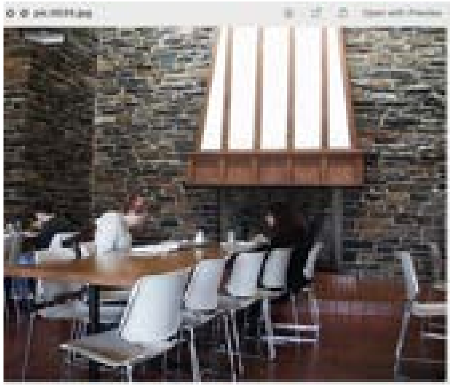


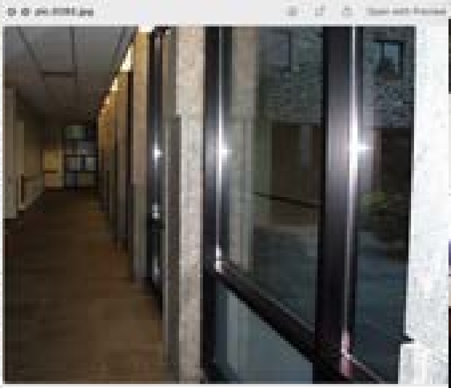
4. **Texture and Color**
   Use a whole image color histogram and a whole image texture histogram as the feature vector. Choose histogram of gradient magnitudes as a texture metric. Use equal weighted histogram intersections for color and gradient magnitude to calculate combined distance ((color_dist + grad_dist) / 2.0f).

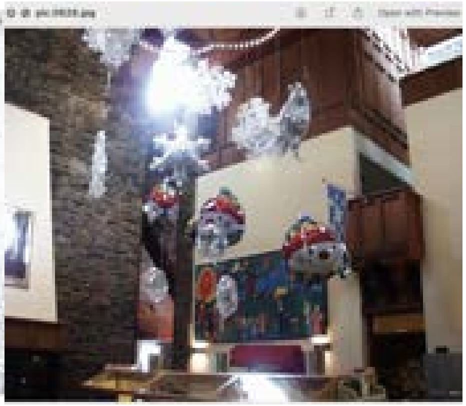
   Required results 4: show the top three matches for the target image pic.0535.jpg and show how they differ when compared to tasks 2 and 3.

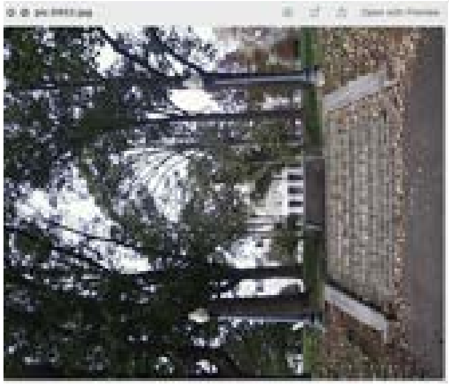


```
mac@Mac build % ./feature ../olympus/ features.csv 4
mac@Mac build % ./imgmatch ../olympus/pic.0535.jpg features.csv 4 4
Reading features.csv
Finished reading CSV file
Top 4 matches for ../olympus/pic.0535.jpg:
1. ../olympus//pic.0535.jpg (distance: -1.00)
2. ../olympus//pic.0285.jpg (distance: -0.45)
3. ../olympus//pic.0628.jpg (distance: -0.35)
4. ../olympus//pic.0952.jpg (distance: -0.34)
```









Task 2 (Chromaticity):Good for simple color patterns. But it loses intensity information and no spatial or texture information.

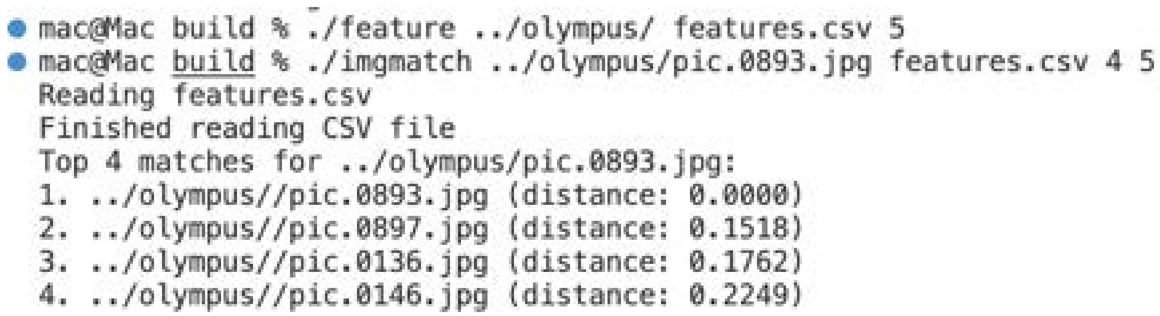
Task 3 (Split RGB):Captures spatial layout and full color information. It is better for objects with distinct top/bottom. It does not capture texture information and has fixed spatial split

Task 4 (Color + Texture):Combines color and edges thus it is better for textured or mixed materials objects. But it lacks spatial information.

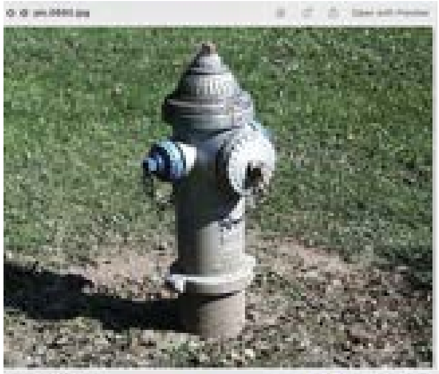
5. Deep Network Embeddings

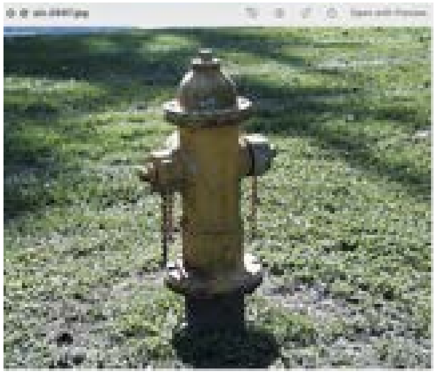
Use the feature vectors contained in the ResNet18 csv file. In ResNet18, the 512 features are the output of the final average pooling layer of a ResNet18 deep network pre-trained on ImageNet which is a 1M image database with 1k categories of diverse types. Use cosine distance as distance matrix.

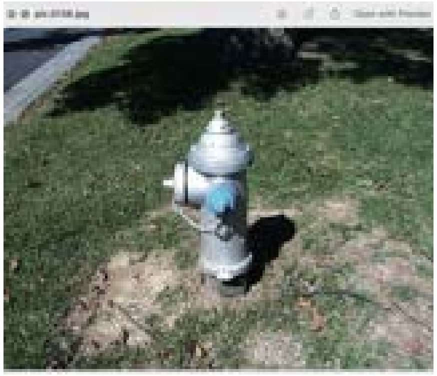
Required results 5: include the top 3 results for images pic.0893.jpg and pic.0164.jpg and compare the results with the prior methods.

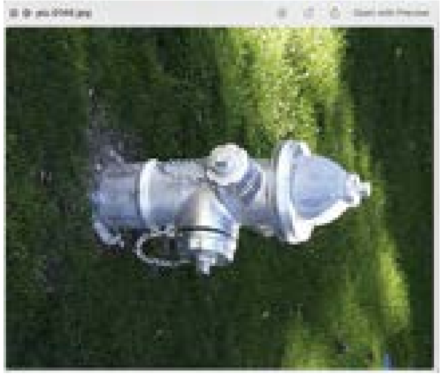


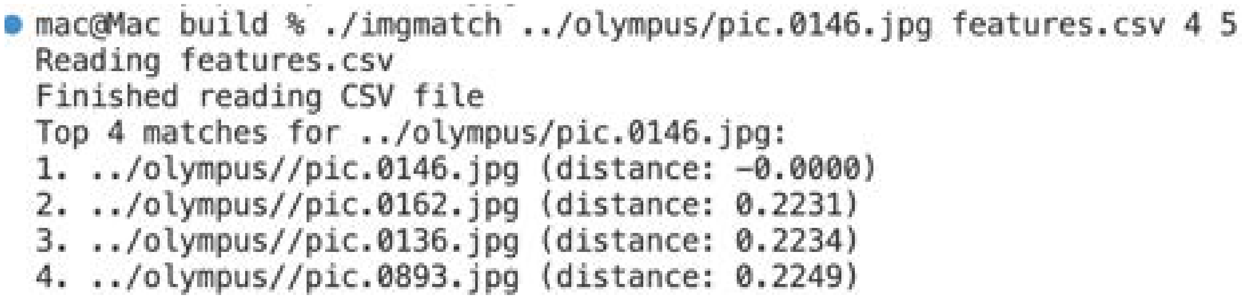
```
● mac@Mac build % ./feature ../olympus/ features.csv 5
● mac@Mac build % ./imgmatch ../olympus/pic.0893.jpg features.csv 4 5
  Reading features.csv
  Finished reading CSV file
  Top 4 matches for ../olympus/pic.0893.jpg:
  1. ../olympus//pic.0893.jpg (distance: 0.0000)
  2. ../olympus//pic.0897.jpg (distance: 0.1518)
  3. ../olympus//pic.0136.jpg (distance: 0.1762)
  4. ../olympus//pic.0146.jpg (distance: 0.2249)
```



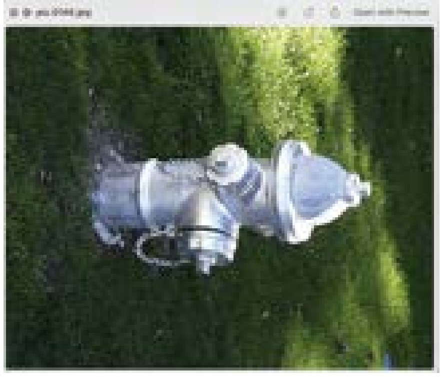


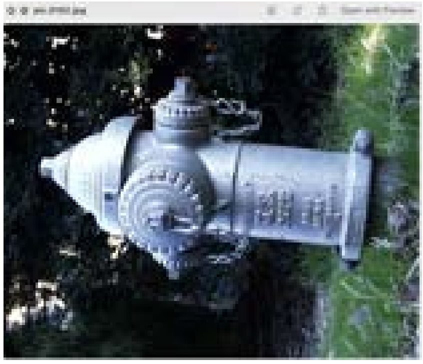


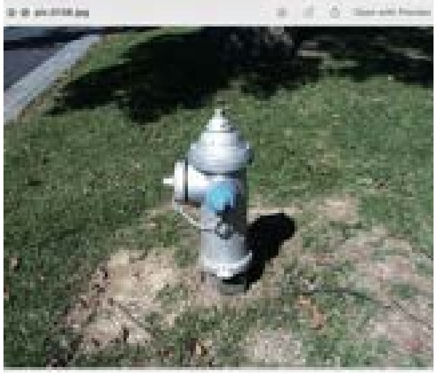


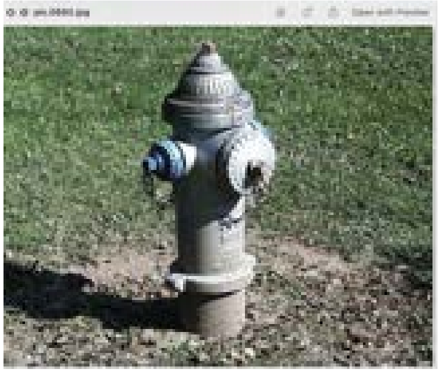


```
mac@Mac build % ./imgmatch ../olympus/pic.0146.jpg features.csv 4 5
Reading features.csv
Finished reading CSV file
Top 4 matches for ../olympus/pic.0146.jpg:
1. ../olympus//pic.0146.jpg (distance: -0.0000)
2. ../olympus//pic.0162.jpg (distance: 0.2231)
3. ../olympus//pic.0136.jpg (distance: 0.2234)
4. ../olympus//pic.0893.jpg (distance: 0.2249)
```









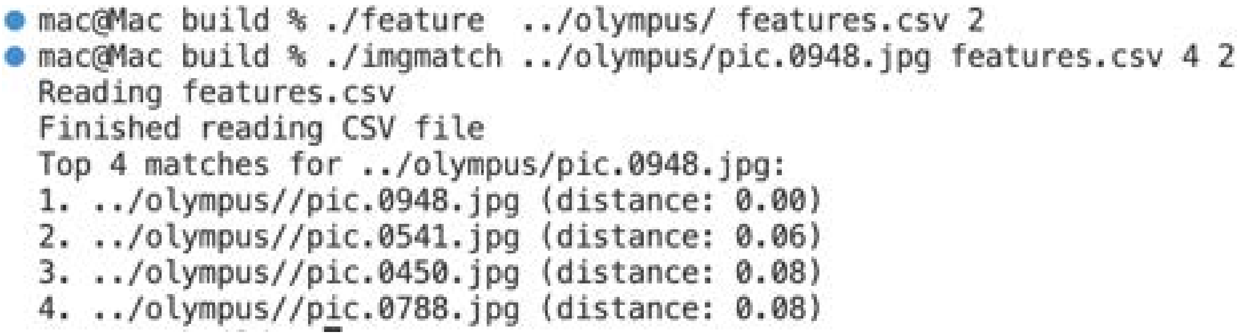
ResNet18 captures high-level semantic features which better understands object content, it also uses cosine distance instead of histogram intersection.

6. **Compare DNN Embeddings and Classic Features**
   Pick 2-3 images and compare the results of using DNN Embeddings versus classic features. Some interesting images to try are 1072, 948, and 734. Is the DNN embedding vector always better?

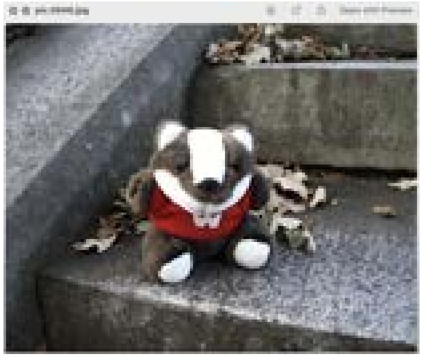
Required result 6: compare and contrast the DNN embedding and classic features results for 2-3 images of your choice.

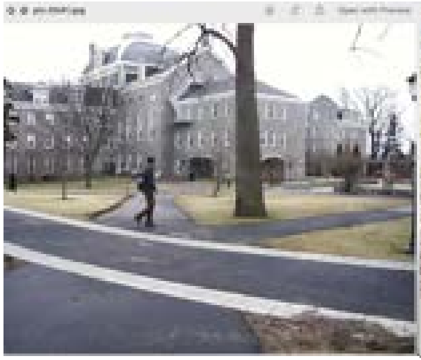
For image 0948

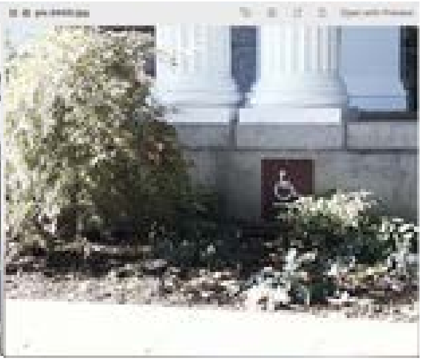
Chromaticity histogram

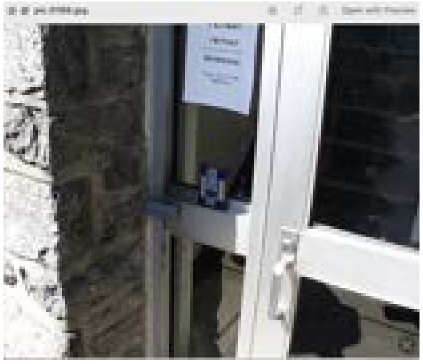


```
● mac@Mac build % ./feature  ../olympus/ features.csv 2
● mac@Mac build % ./imgmatch ../olympus/pic.0948.jpg features.csv 4 2
  Reading features.csv
  Finished reading CSV file
  Top 4 matches for ../olympus/pic.0948.jpg:
  1. ../olympus//pic.0948.jpg (distance: 0.00)
  2. ../olympus//pic.0541.jpg (distance: 0.06)
  3. ../olympus//pic.0450.jpg (distance: 0.08)
  4. ../olympus//pic.0788.jpg (distance: 0.08)
```

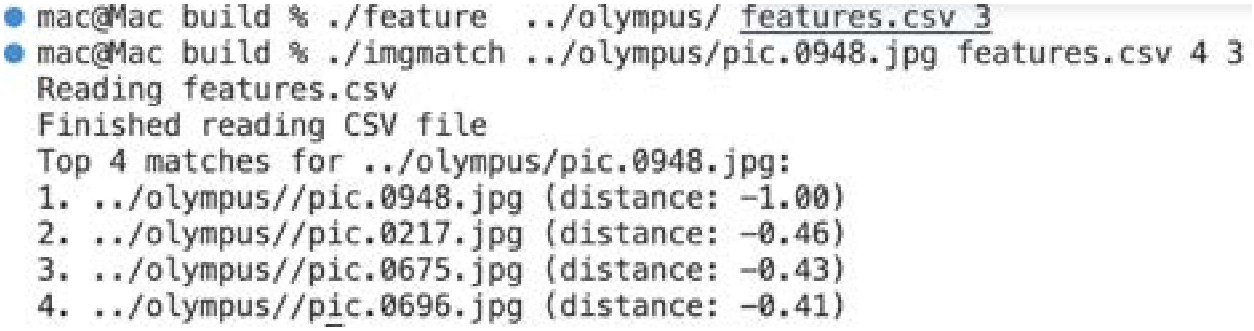




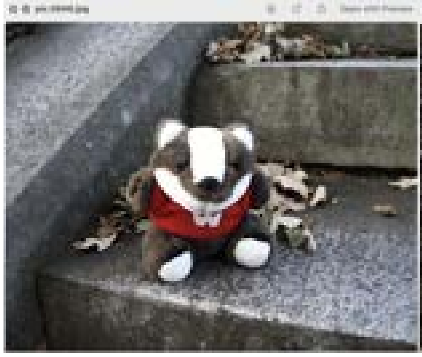


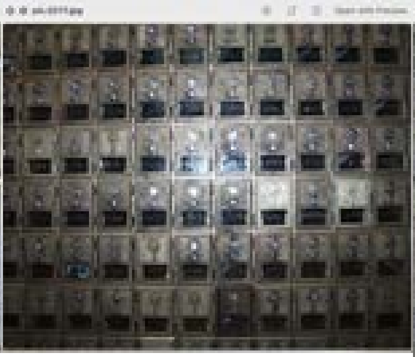


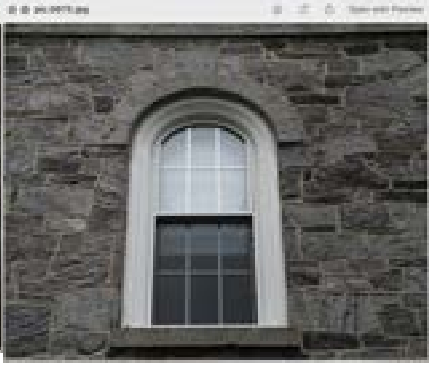
Split RGB histogram

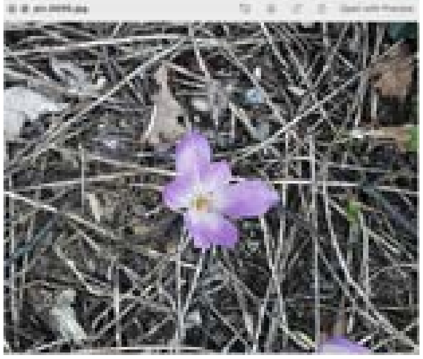


```
● mac@Mac build % ./feature  ../olympus/ features.csv 3
● mac@Mac build % ./imgmatch ../olympus/pic.0948.jpg features.csv 4 3
  Reading features.csv
  Finished reading CSV file
  Top 4 matches for ../olympus/pic.0948.jpg:
  1. ../olympus//pic.0948.jpg (distance: -1.00)
  2. ../olympus//pic.0217.jpg (distance: -0.46)
  3. ../olympus//pic.0675.jpg (distance: -0.43)
  4. ../olympus//pic.0696.jpg (distance: -0.41)
```

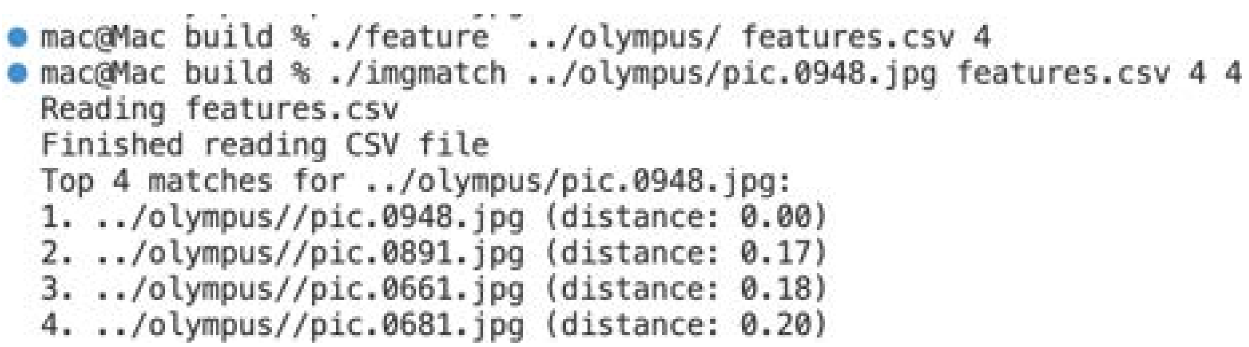




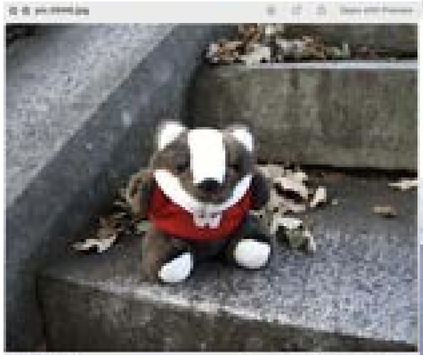


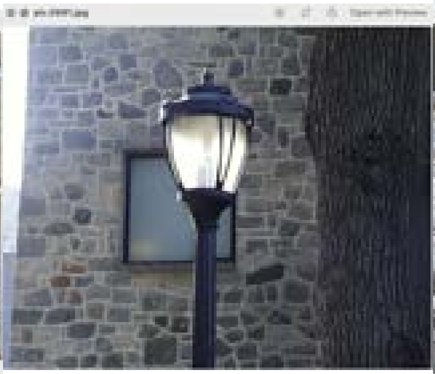


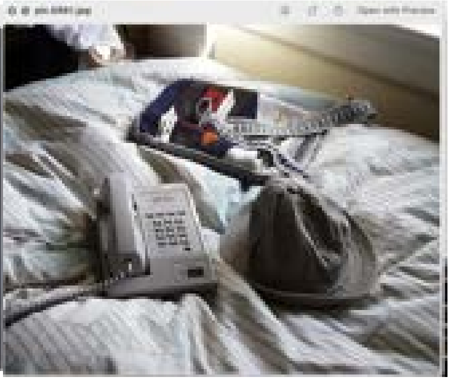
Color-texture histogram

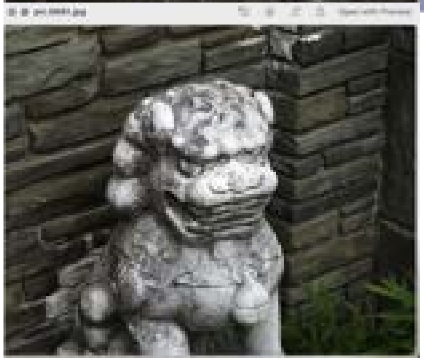


```
● mac@Mac build % ./feature  ../olympus/ features.csv 4
● mac@Mac build % ./imgmatch ../olympus/pic.0948.jpg features.csv 4 4
  Reading features.csv
  Finished reading CSV file
  Top 4 matches for ../olympus/pic.0948.jpg:
  1. ../olympus//pic.0948.jpg (distance: 0.00)
  2. ../olympus//pic.0891.jpg (distance: 0.17)
  3. ../olympus//pic.0661.jpg (distance: 0.18)
  4. ../olympus//pic.0681.jpg (distance: 0.20)
```

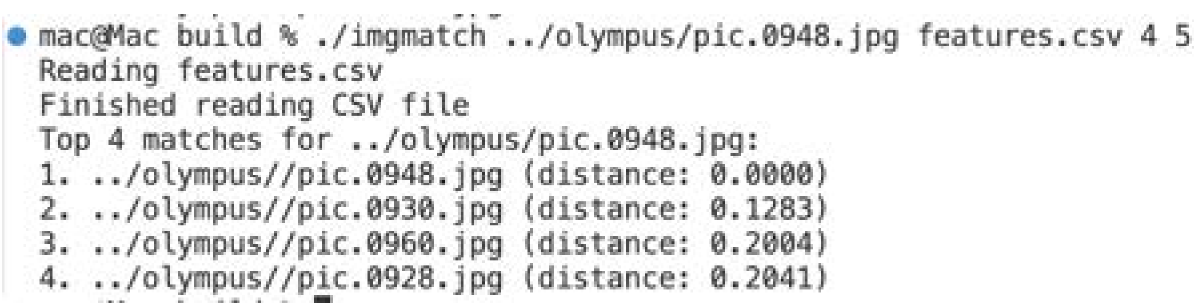




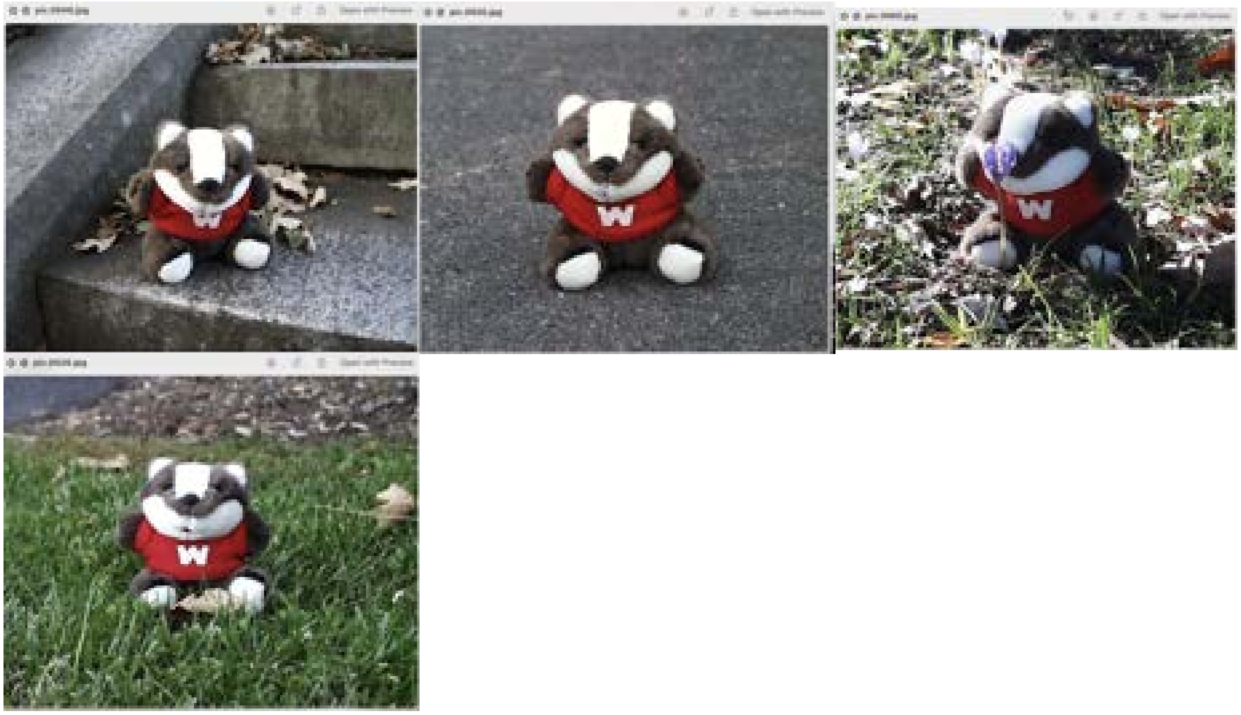




ResNet18



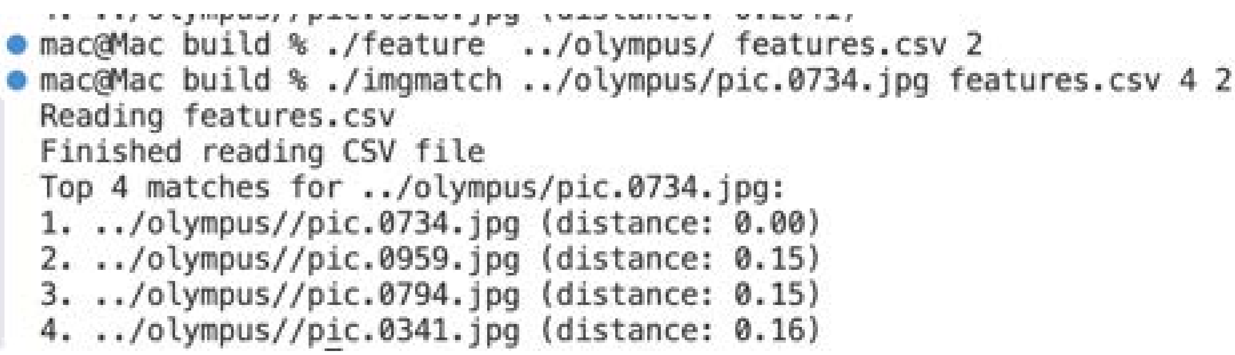
```
mac@Mac build % ./imgmatch ../olympus/pic.0948.jpg features.csv 4 5
Reading features.csv
Finished reading CSV file
Top 4 matches for ../olympus/pic.0948.jpg:
1. ../olympus//pic.0948.jpg (distance: 0.0000)
2. ../olympus//pic.0930.jpg (distance: 0.1283)
3. ../olympus//pic.0960.jpg (distance: 0.2004)
4. ../olympus//pic.0928.jpg (distance: 0.2041)
```



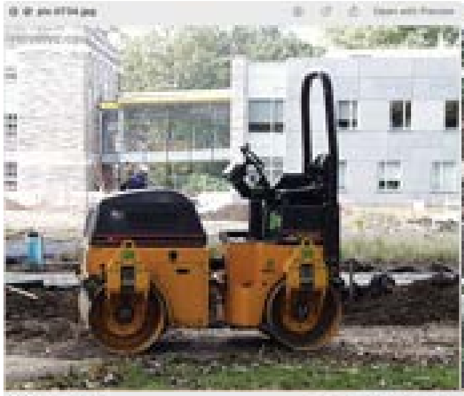
Looking good!

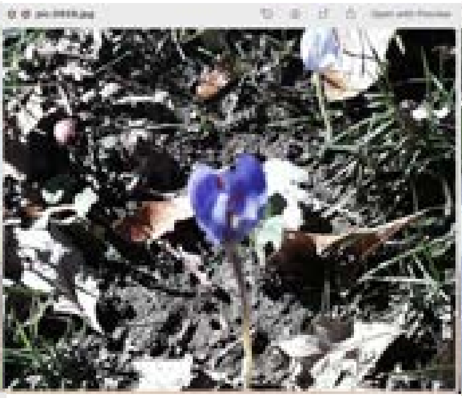
For image 0734

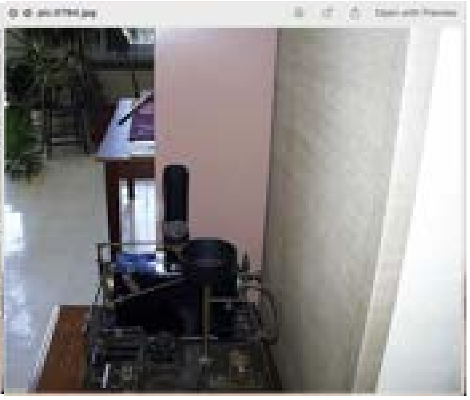
Chromaticity histogram

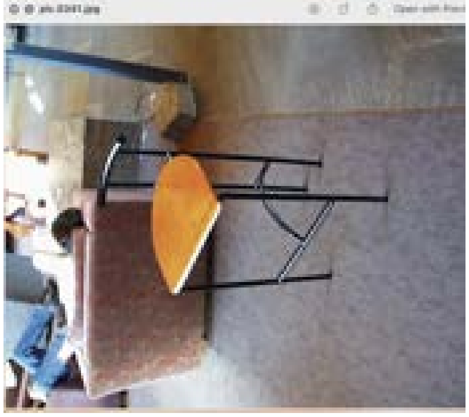


```
● mac@Mac build % ./feature  ../olympus/ features.csv 2
● mac@Mac build % ./imgmatch ../olympus/pic.0734.jpg features.csv 4 2
  Reading features.csv
  Finished reading CSV file
  Top 4 matches for ../olympus/pic.0734.jpg:
  1. ../olympus//pic.0734.jpg (distance: 0.00)
  2. ../olympus//pic.0959.jpg (distance: 0.15)
  3. ../olympus//pic.0794.jpg (distance: 0.15)
  4. ../olympus//pic.0341.jpg (distance: 0.16)
```

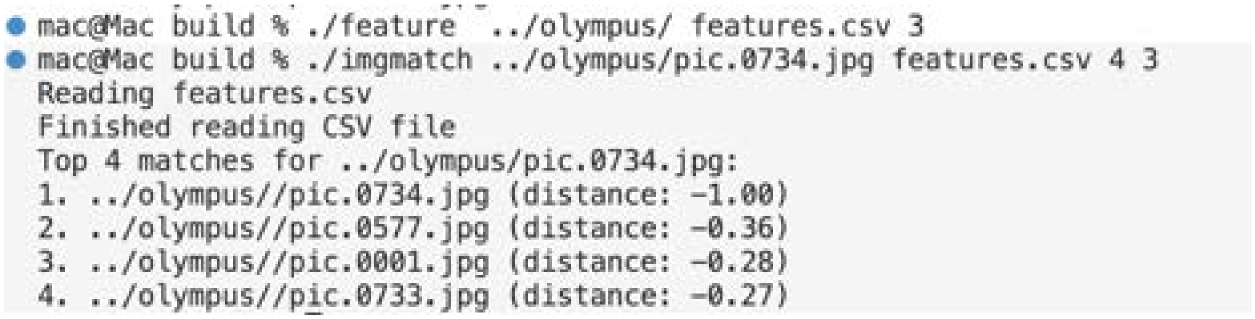




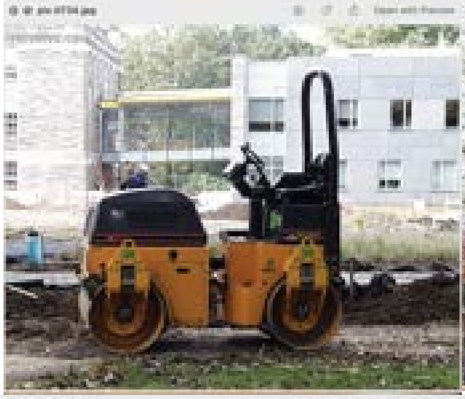


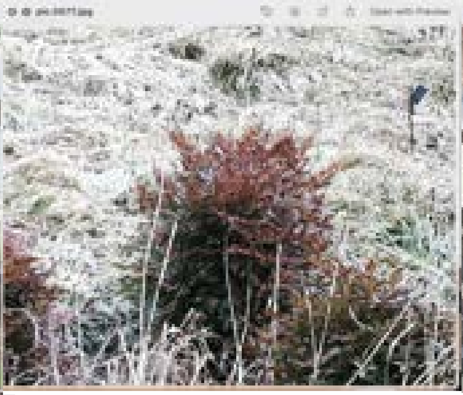


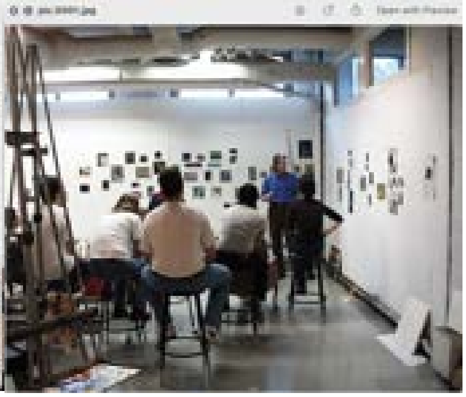
Split RGB histogram

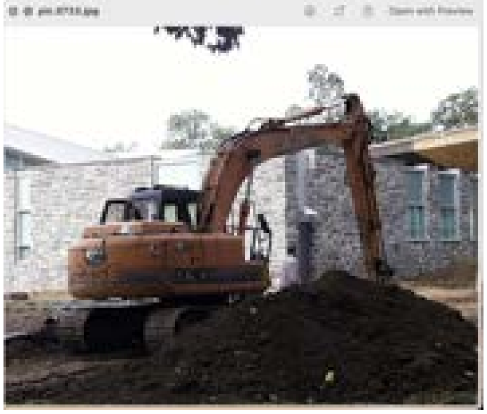


```
mac@Mac build % ./feature  ../olympus/ features.csv 3
mac@Mac build % ./imgmatch ../olympus/pic.0734.jpg features.csv 4 3
Reading features.csv
Finished reading CSV file
Top 4 matches for ../olympus/pic.0734.jpg:
1. ../olympus//pic.0734.jpg (distance: -1.00)
2. ../olympus//pic.0577.jpg (distance: -0.36)
3. ../olympus//pic.0001.jpg (distance: -0.28)
4. ../olympus//pic.0733.jpg (distance: -0.27)
```

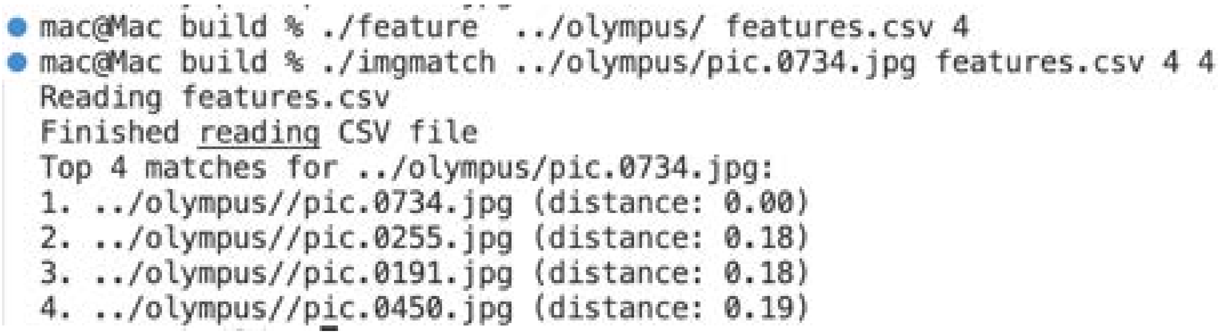




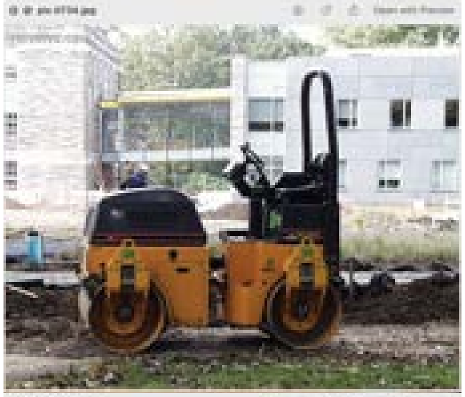


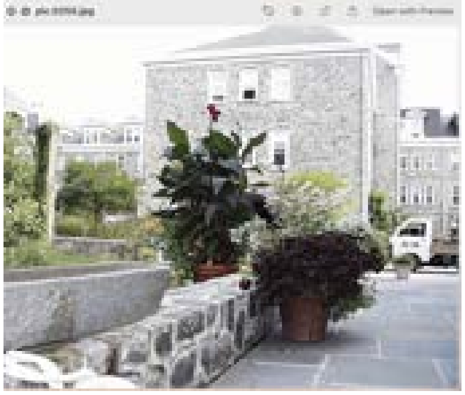


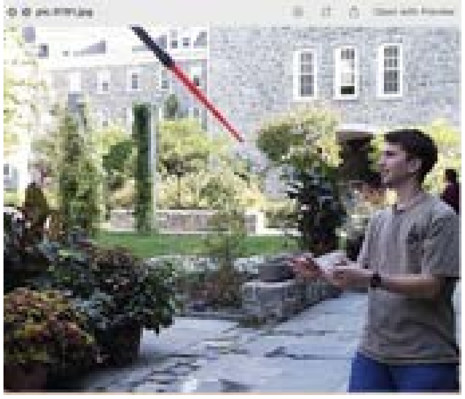
Color-texture histogram

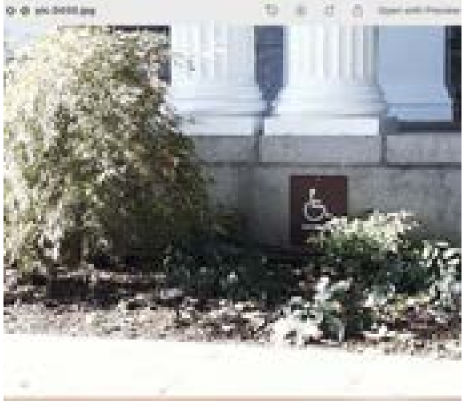


```
● mac@Mac build % ./feature  ../olympus/ features.csv 4
● mac@Mac build % ./imgmatch ../olympus/pic.0734.jpg features.csv 4 4
  Reading features.csv
  Finished reading CSV file
  Top 4 matches for ../olympus/pic.0734.jpg:
  1. ../olympus//pic.0734.jpg (distance: 0.00)
  2. ../olympus//pic.0255.jpg (distance: 0.18)
  3. ../olympus//pic.0191.jpg (distance: 0.18)
  4. ../olympus//pic.0450.jpg (distance: 0.19)
```

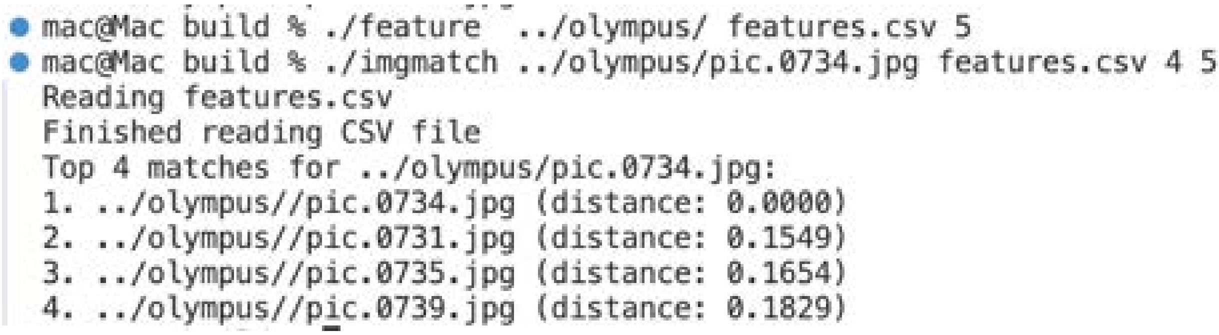




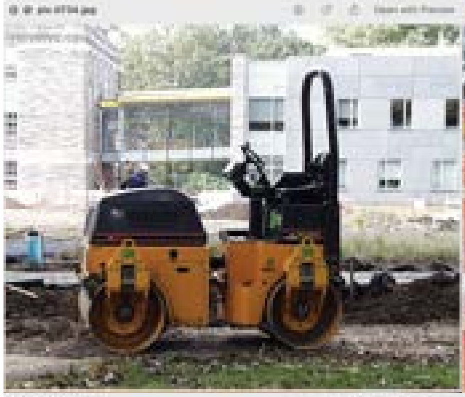


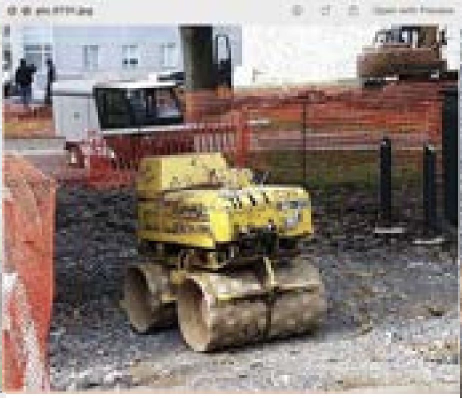


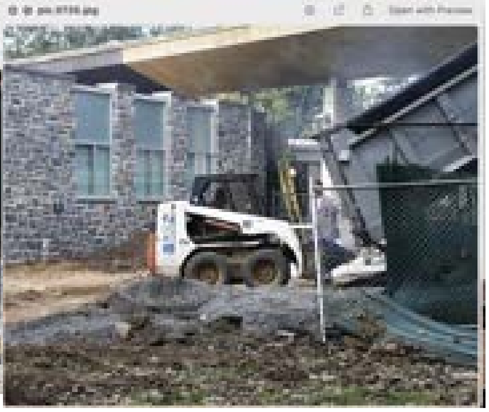
ResNet18

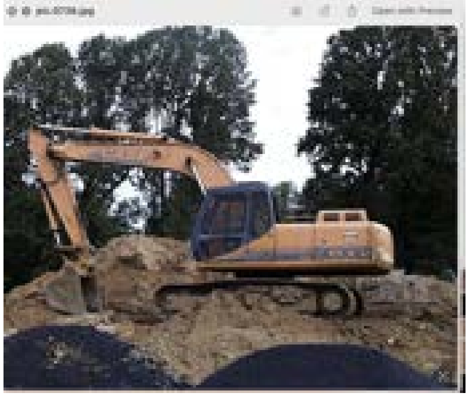


```
● mac@Mac build % ./feature  ../olympus/ features.csv 5
● mac@Mac build % ./imgmatch ../olympus/pic.0734.jpg features.csv 4 5
  Reading features.csv
  Finished reading CSV file
  Top 4 matches for ../olympus/pic.0734.jpg:
  1. ../olympus//pic.0734.jpg (distance: 0.0000)
  2. ../olympus//pic.0731.jpg (distance: 0.1549)
  3. ../olympus//pic.0735.jpg (distance: 0.1654)
  4. ../olympus//pic.0739.jpg (distance: 0.1829)
```









In the two examples of image 0948 and 0734, ResNet18 works better for understanding object types.

The others are color-based matches which work when color is the main distinguishing feature.

7. **Custom Design:**

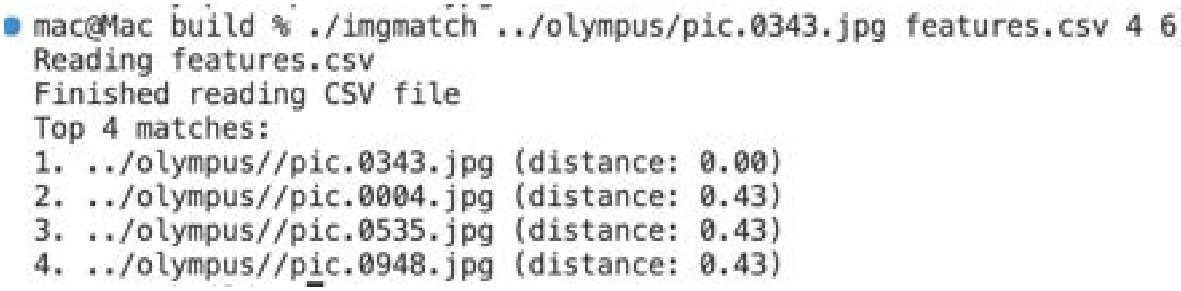
   Required results 7: for two target images of your choice, show the top five results. It's also helpful to show some of the least similar results.

   It includes the following features: ResNet18 features for semantic understanding, color-texture histogram on foreground, split RGB histogram on central region, weighted distance combination: (implement in header file)

   ResNet18: 40%

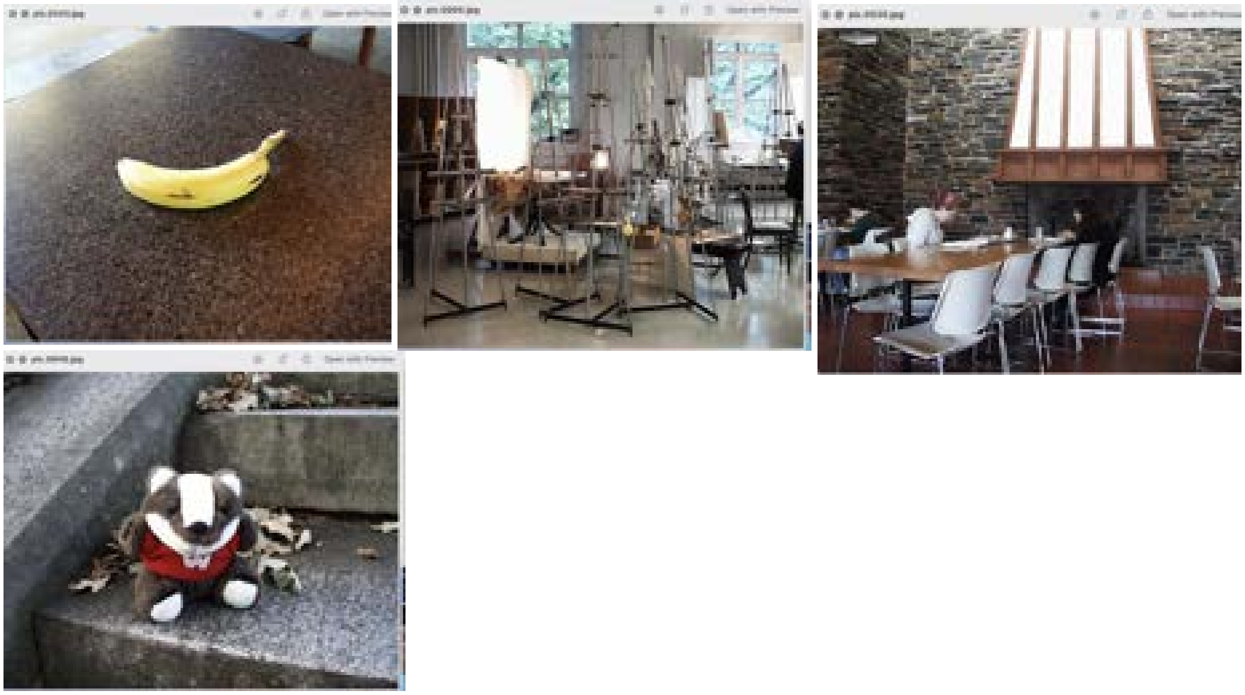
   Foreground: 40%

   Central: 20%



```
mac@Mac build % ./imgmatch ../olympus/pic.0343.jpg features.csv 4 6
Reading features.csv
Finished reading CSV file
Top 4 matches:
1. ../olympus//pic.0343.jpg (distance: 0.00)
2. ../olympus//pic.0004.jpg (distance: 0.43)
3. ../olympus//pic.0535.jpg (distance: 0.43)
4. ../olympus//pic.0948.jpg (distance: 0.43)
```

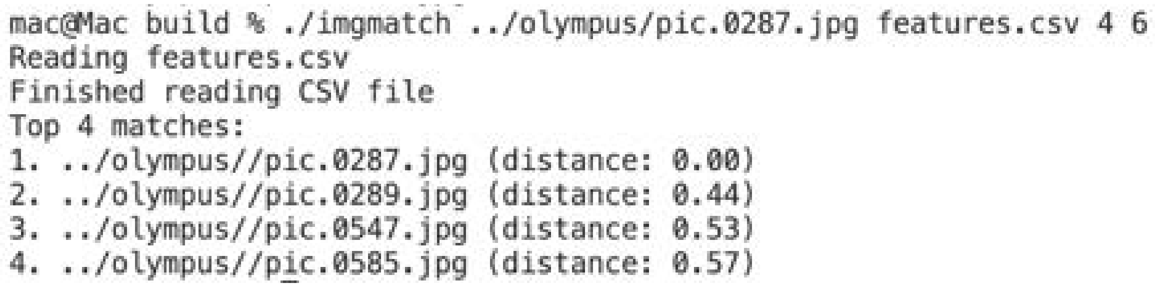
   Results for target picture 0343 (banana)



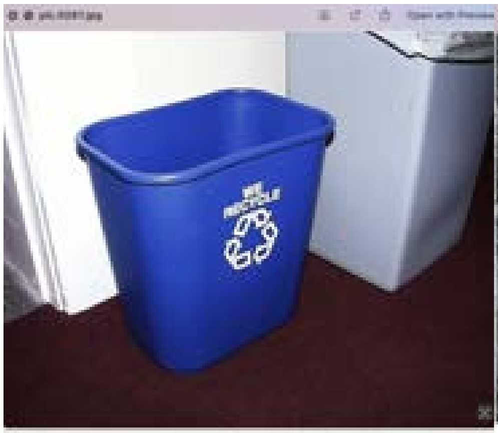
Extension: There are also lots of pictures of blue trash can bins. How many of these can your system recall given a target image that contains one?

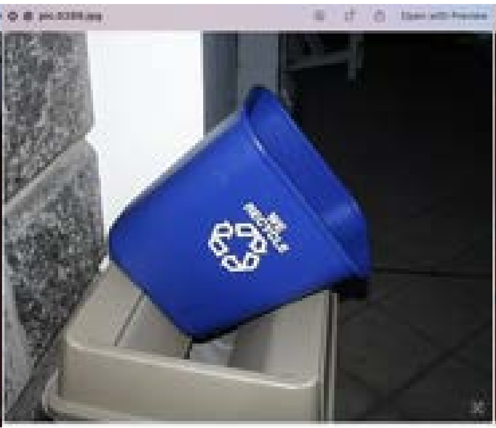
My system can identify 1 blue bin given a target image.

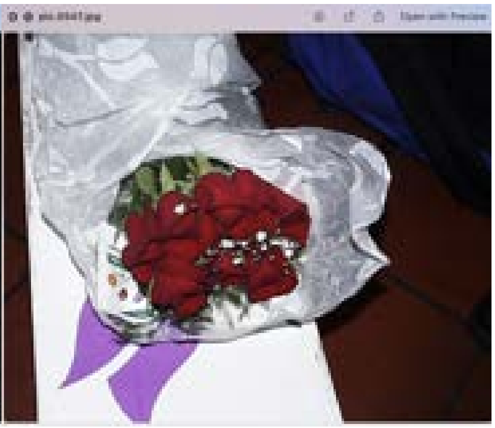
Results for 0287(blue bin)

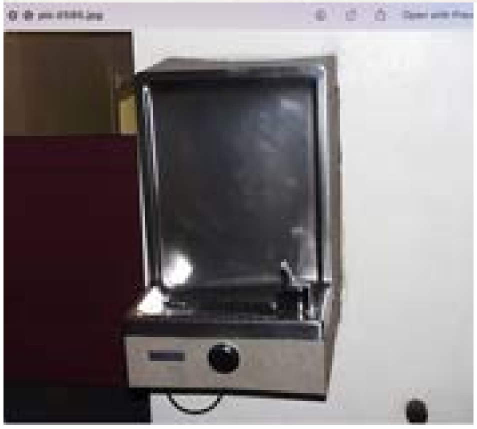


```
mac@Mac build % ./imgmatch ../olympus/pic.0287.jpg features.csv 4 6
Reading features.csv
Finished reading CSV file
Top 4 matches:
1. ../olympus//pic.0287.jpg (distance: 0.00)
2. ../olympus//pic.0289.jpg (distance: 0.44)
3. ../olympus//pic.0547.jpg (distance: 0.53)
4. ../olympus//pic.0585.jpg (distance: 0.57)
```









Both matching results are not ideal in terms of identifying similar objects compared to only using the DNN Embeddings. The reasons might be: 1. Feature Weight Balance Issues: current equal weighting (40-40-20) might be diluting ResNet's effectiveness. 2.Histogram Features Limitations: use basic color-texture histogram which does not capture spatial relationships. May add noise rather than useful information.