ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

----------❖----------

BÁO CÁO ĐỒ ÁN CUỐI KỲ

NHẬP MÔN THỊ GIÁC MÁY TÍNH

Đề tài: Xây dựng hệ thống truy vấn hình ảnh thông qua đặc trưng màu sắc

Lớp: CS231.M22.KHCL
Tên sinh viên thực hiện: Đặng Xuân Mai
MSSV: 19521820

TP.Hồ Chí Minh, năm 2022

Mục lục

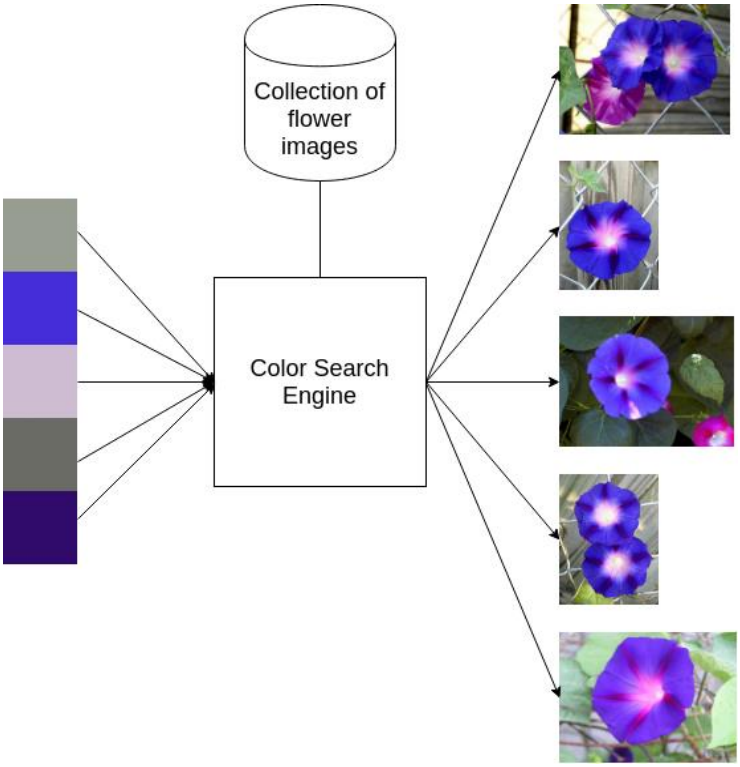
Danh mục hình ảnh

Danh mục đoạn chương trình

Danh mục biểu đồ

1. Tổng quan về bài toán

Bài toán xây dựng một hệ thống tìm kiếm hình ảnh bằng một sắc sẽ có input (đầu vào) là một hoặc một loạt những màu sắc và output (đầu ra) là những hình ảnh có những màu sắc input chiếm phần lớn trong bức ảnh. [1]

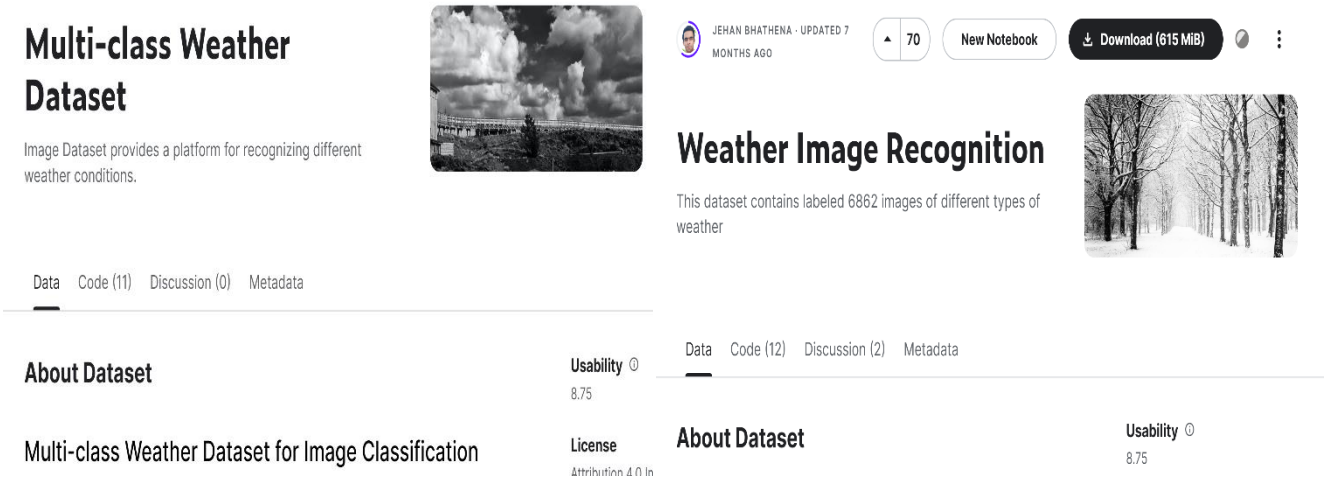


Hình 1 - Hệ thống truy vấn hình ảnh theo màu sắc [1]

Dựa theo những thông tin trên, nhóm sẽ đặt mục tiêu trên hết khi xây dựng đồ án này là mô hình xây dựng được có thể nhận một bức ảnh đầu vào, sau đó trả về top 20 bức ảnh có những màu sắc mà mô hình cho là giống với bức ảnh truy vấn nhất.

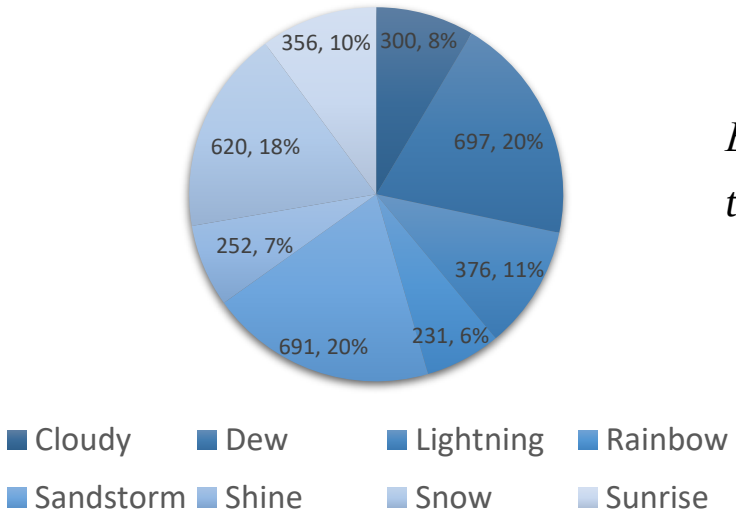
2. Các bộ dữ liệu sử dụng

Nhóm lựa chọn sử dụng 2 bộ dữ liệu trên Kaggle [2], [3] về nhận diện thời tiết để làm tập ảnh huấn luyện cho hệ thống.

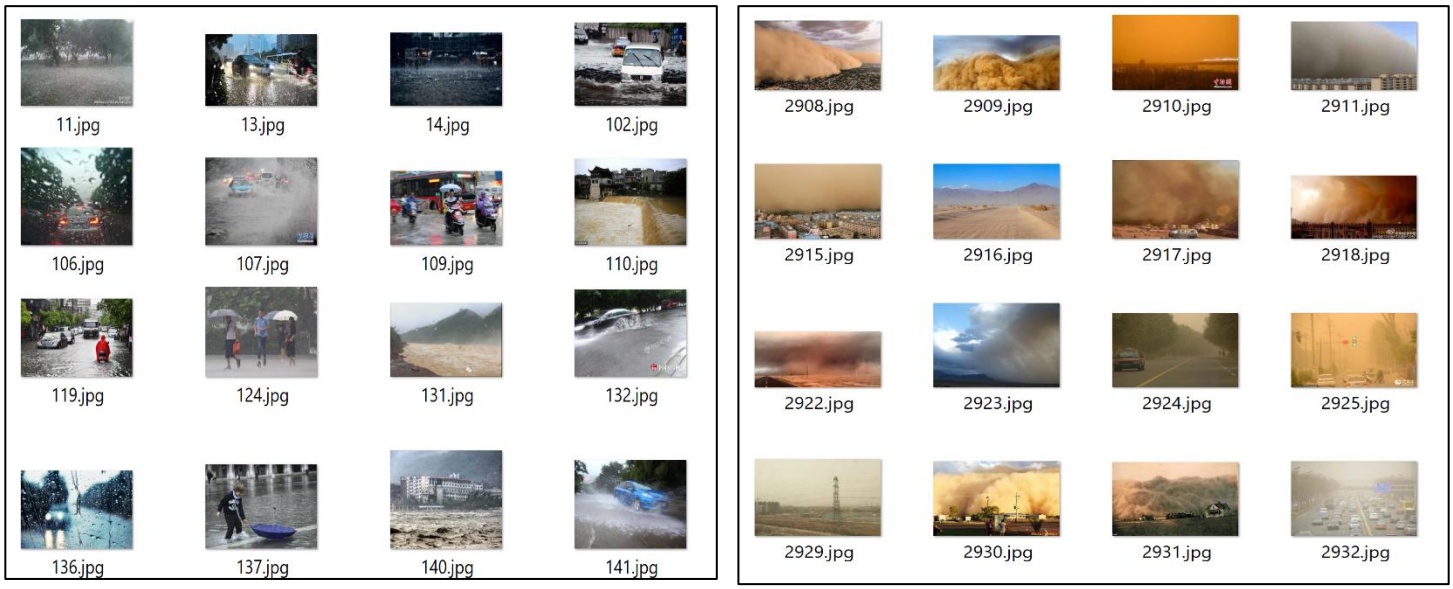


Hình 2 - Các tập dữ liệu sử dụng

Hai tập dữ liệu trên tổng hợp lại gồm 14 lớp dữ liệu thời tiết khác nhau, như: snow (tuyết), lightning (sấm chớp), …. Tuy nhiên trong quá trình sử dụng, nhóm quyết định chỉ lấy 3523 ảnh từ 8 lớp như hình 3, nguyên nhân là có nhiều nhóm hình ảnh, tuy các hình ảnh cùng một nhóm, nhưng màu sắc lại không có độ tương đồng (nếu trong trường hợp này thì nên dùng content-base để xây dựng model sẽ thích hợp hơn), nên nhóm đã tự chọn lấy 8 nhóm có độ tương đồng màu sắc giữa các ảnh trong nhóm để thực hiện.



Biểu đồ 1 - Phân bố dữ liệu theo dạng số ảnh, % chiếm



Hình 3 - Sự khác biệt giữa nhóm ảnh không có sự tương đồng (trái), và có sự tương đồng (phải)

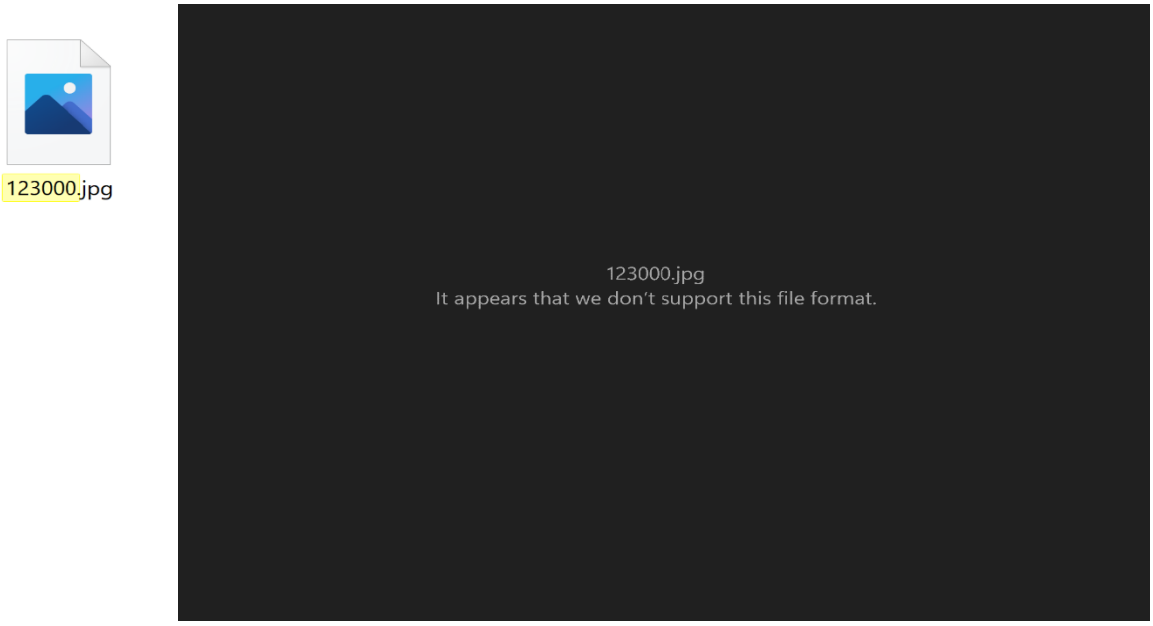
Như hình 3, nhóm ảnh bên phải là nhóm ảnh của lớp thời tiết trời mưa, những màu sắc của áo mưa, xe đi đường, đèn xe, … sẽ khiến cho bức ảnh có nhiều màu sắc hơn, từ đó khó nhận diện được màu chủ đạo nên nhóm không sử dụng hình ảnh của lớp thời tiết đó, ngược lại, nhóm hình ảnh bên phải có nhóm màu chủ đạo chung là vàng – cam – nâu (nhóm hình ảnh của thời tiết bão cát), điều đó sẽ khiến nhóm dễ thực hiện bài toán của mình hơn.

Tuy nhiên, sau lần báo cáo trên lớp thứ nhất vào ngày 17/5/2022, nhóm đã được thầy Dũng gợi ý là sử dụng một tập dữ liệu có sẵn thích hợp hơn cho bài toán này là tập dữ liệu Holiday [4], và sau đó nhóm đã được thầy giúp đỡ để download tập dữ liệu này về. Nên trong báo bài báo cáo này thì các phương pháp làm lần báo cáo thứ nhất sẽ thực hiện trên tập dữ liệu do nhóm tự chuẩn bị (Kaggle), còn các phương pháp đã được góp ý để cải thiện thì sẽ thực hiện trên tập dữ liệu Holiday.

Tổng quan về tập dữ liệu Holiday gồm 1490 ảnh, được tác giả của bộ dữ liệu này gán nhãn các hình ảnh có liên quan đến nhau một cách cụ thể.

Về tập ảnh truy vấn để đánh giá hệ thống:

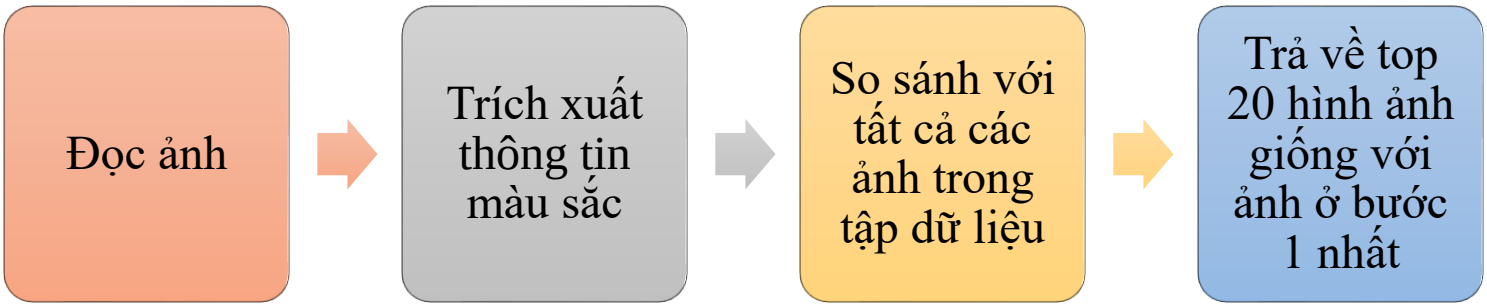
- Đối với bộ dữ liệu nhóm tự chuẩn bị: 8 ảnh từ 8 lớp thời tiết khác nhau nhóm đã chọn, 8 ảnh này không đưa vào tập dữ liệu để trích xuất đặc trưng, đảm bảo là hệ thống chưa từng được xử lý.
- Đối với bộ dữ liệu Holiday: tập 499 (ban đầu gồm 500 ảnh, tuy nhiên có một ảnh trong tập dữ liệu không thể đọc được, nên loại bỏ ảnh đó đi - ảnh "123000.jpg"), 499 ảnh này từng được xuất hiện, trích xuất đặc trưng trong quá trình xây dựng hệ thống.



Hình 4 - Ảnh bị lỗi và thông báo lỗi khi mở ảnh

3. Quá trình thực hiện

Quá trình thực hiện chung của bài toán này chính là:



Biểu đồ 2 - Các bước thực hiện chính

Ở bước *Trích xuất thông tin màu sắc,* nhóm sử dụng hai phương pháp khác nhau: dùng các hàm trợ giúp để tự tìm màu chủ đạo trong bức ảnh và tính histogram để tìm được đặc trưng màu sắc của bức ảnh.

3.1 Trích xuất thông tin màu sắc

3.1.1 Tìm màu chủ đạo không dùng Histogram

Theo như những gì nhóm đã tự tìm hiểu được, việc trích xuất màu chủ đạo có thể dùng K-Means Clustering để gom cụm hình ảnh thành từng vùng màu khác nhau [5] hoặc sử dụng một thư viện colorthieft [6] để trích xuất được các màu sắc đặc trưng của ảnh. Áp dụng các kiến thức đã được biết trong quá trình học thực hành, nhóm dùng 2 đoạn code để trích xuất K-Means và colorthief như sau:

```
def KMeansExtract(img, number_of_colors=5):
  nrow, ncol, nchl = img.shape
  g = img.reshape(nrow*ncol,nchl)
  clf = KMeans(n_clusters = number_of_colors, random_state = 0)
  k_means = clf.fit(g)
  colors = []
  for each in clf.cluster_centers_:
    color = ConvertRGBToDecimal(each)
    colors.append(color)
  res = np.array(colors).reshape(1, -1)
  return res
```

Code demo 2 - Trích xuất dùng K-Means

```
def ColorthiefExtract(image_path, number_of_colors=5):
  img = ColorThief(image_path)
  dominant_colors = img.get_palette(color_count=number_of_colors)
  colors = []
  for each in dominant_colors:
    color = ConvertRGBToDecimal(each)
    colors.append(color)
  res = np.array(colors).reshape(1, -1)
  return res
```

Code demo 1 - Trích xuất dùng colorthief

Tuy nhiên nếu áp dụng số màu chủ đạo quá nhiều thì mỗi đặc trưng lại trở nên quá riêng biệt đối với từng ảnh, số màu chủ đạo quá ít thì cũng không thể tìm được những ảnh thật sự tương đồng với ảnh truy vấn vì đặc trưng quá bao quát. Sau khi tìm hiểu thì nhóm quyết định chọn 4 bộ trích xuất đặc trưng gồm:

- K-Means 5 màu
- K-Means 6 màu
- Colorthief 5 màu
- Colorthief 6 màu

Trong quá trình xử lý để thực hiện trích xuất đặc trưng, nhóm cũng phải có thêm những xử lý khác để khiến cho bài toán dễ dàng tính toán hơn. Ví dụ như xử lý từ hệ màu RGB sang một số nguyên [7] để ma trận tính toán chỉ còn một mảng số nguyên với số phần tử là số màu muốn trích xuất.

```
def ConvertRGBToDecimal(triplet):
  R = int(round(triplet[0], 0))
  G = int(round(triplet[1], 0))
  B = int(round(triplet[2], 0))
  return R * 65536 + G * 256 + B
```

Code demo 3 - Hàm chuyển đổi RGB sang số nguyên

3.1.2 Tìm màu chủ đạo thông qua tính Histogram

Thực hiện tính histogram như trong bài lab 1 nhóm đã được làm trên lớp, tuy nhiên thì nhóm sẽ thay đổi thông số là đọc 3 kênh màu RGB, với mỗi giá trị màu sẽ nằm trong khoảng từ 0 – 256.

```
def CalHistogram(image_path, bin=[8,8,8]):
  img = cv.imread(image_path)
  if img is None:
    img = plt.imread(image_path)
    img = img[:,:,:3]
  hist = cv.calcHist([img],[0, 1, 2],None,[bin[0], bin[1], bin
[2]],[0,256, 0, 256, 0, 256])
  hist = hist.reshape(1, -1)/ hist.sum()
  return hist
```

Code demo 4 - Hàm tính histogram

Dòng code có câu lệnh *if img is None* sẽ được dùng để đọc những hình ảnh không thể đọc bằng câu lệnh imread (đây là trường hợp phát sinh trong quá trình code, được viết ra để xử lý riêng).

Tham số bin sẽ được dùng để khảo sát mức độ chính xác của hệ thống. Nhóm sẽ chọn những bin trong khoảng từ 8 – 64 để đánh giá [8] hệ thống gồm:

- bin = [8, 8, 8]
- bin = [12, 12, 12]
- bin = [30, 30, 30]
- bin = [64, 64, 64]

Ngoài ra thì nhóm cũng sẽ thử đánh giá hệ thống trên hệ màu HSV [9], bin = [8, 12, 3] để so sánh kết quả thu được.

3.2 So sánh ảnh truy vấn và những ảnh khác trong tập dữ liệu

Vì việc nhận 1 đầu vào của người dùng rồi trả về những kết quả liên quan sẽ phù hợp với tính chất của một hệ thống truy vấn hơn là phân loại tấm ảnh đầu vào rồi trả về những kết quả cùng lớp. Vì thế nhóm sẽ dùng độ tương đồng cosine để đo lường sự giống nhau của những bức ảnh, từ đó rút ra top 20 kết quả ảnh có độ tương đồng cao nhất.

Ngoài ra, nhóm còn sử dụng độ đo mAP để đánh giá một hệ thống truy vấn thông tin [10].

Trong phần đánh giá hệ thống với tập dữ liệu Holiday, tuy đã được thầy cung cấp cho tool để đánh giá hệ thống, tuy nhiên lúc cài đặt thì nhóm bị lỗi và không thể chạy được chương trình đó. Thế nên nhóm tự viết một đoạn code để tính toán độ đo mAP của cả 2 tập dữ liệu (nhóm tự chuẩn bị và Holiday).

Các bước thực hiện tính mAP của bộ dữ liệu Holiday là đọc file perfect_result.dat => tách từng cặp query và một mảng gồm những kết quả trả về phù hợp với query đó => tính mAP. (Các bước chi tiết cụ thể hơn được trình bày trong colab của nhóm)

```
def CalculateAP_Holiday(query_gt, ground_truth, n=20):
  precisions = []
  for index in range(len(ground_truth)):
    each = ground_truth[index]
    if each in query_gt:
      precision = (len(precisions) + 1) / (index + 1)
      precisions.append(precision)
  n = min(n, 20)
  average_precision = sum(precisions) / n
  print("Precision ~", len(precisions) / n)
  return average_precision
```

Code demo 5 - Tính mAP

Dòng code n = min(n, 20) được dùng để lấy giá trị min giữa số ảnh "liên quan" đến query và top-20 đang xét. Trong trường hợp chỉ có 5 ảnh được xem là liên quan thì n = 5, như vậy sẽ đảm bảo được công thức tính mAP.

4. Phân tích đánh giá hệ thống

Các kết quả trả về

Hàm trích xuất	Số màu	Max AP	Min AP	mAP	Thời gian thực hiện trích xuất
Kmeans	5	0.77	0.01	0.22	3 tiếng 20 phút
Kmeans	6	0.44	0.002	0.16	4 tiếng 30 phút
ColorThief	5	0.71	0.007	0.25	12 phút
ColorThief	6	0.81	0.004	0.21	11 phút

Hình 5 - Kết quả trả về khi trích xuất không dùng histogra

So sánh kết quả trên, cùng một tập dữ liệu nhưng thay đổi hàm trích xuất đặc trưng, dùng histogram thì sẽ có kết quả như sau:

Hàm trích xuất	Thời gian thực hiện	mAP
K-Means (cluster = 5)	3 tiếng 20 phút	0.2291
Histogram (bin=[30, 30, 30])	20 phút 24s	0.4295

Hình 6 - So sánh khi dùng K-Means và histogram

Từ kết quả trên, ta có thể thấy được là khi dùng K-Means (hoặc colorthief) thì tốn nhiều thời gian thực hiện hơn, và còn không mang lại kết quả tốt bằng khi dùng histogram. Ngoài ra, mỗi một lần chạy, bộ 5 (hoặc 6) màu chủ đạo sẽ bị lệch một ít (do K-Means clustering không cố định, mỗi lần sẽ ra một điểm center khác nhau).

⇨ Phương pháp trích xuất không dùng histogram không thiết thực

Sau đó nhóm tiến hành thực hiện trích xuất trên các bin khác nhau để xem số lượng màu sắc ảnh hưởng đến kết quả thế nào, và đây là kết quả thu được:

bin	mAP
[8, 8, 8]	0.6332
[12, 12, 12]	0.4583
[30, 30, 30]	0.4579
[64, 64, 64]	0.4484

Hình 7 - Kết quả khi thay đổi số bin

Có thể nhận thấy rằng trong tập dữ liệu này, số lượng màu càng lớn sẽ càng khiến cho hệ thống khó xác định được độ tương đồng giữa các ảnh.

Ngoài ra, khi sử dụng thông số bin và hệ màu HSV như bài tham khảo mà nhóm tìm thấy, kết quả thu về được là:

```
mAP_hsv
```

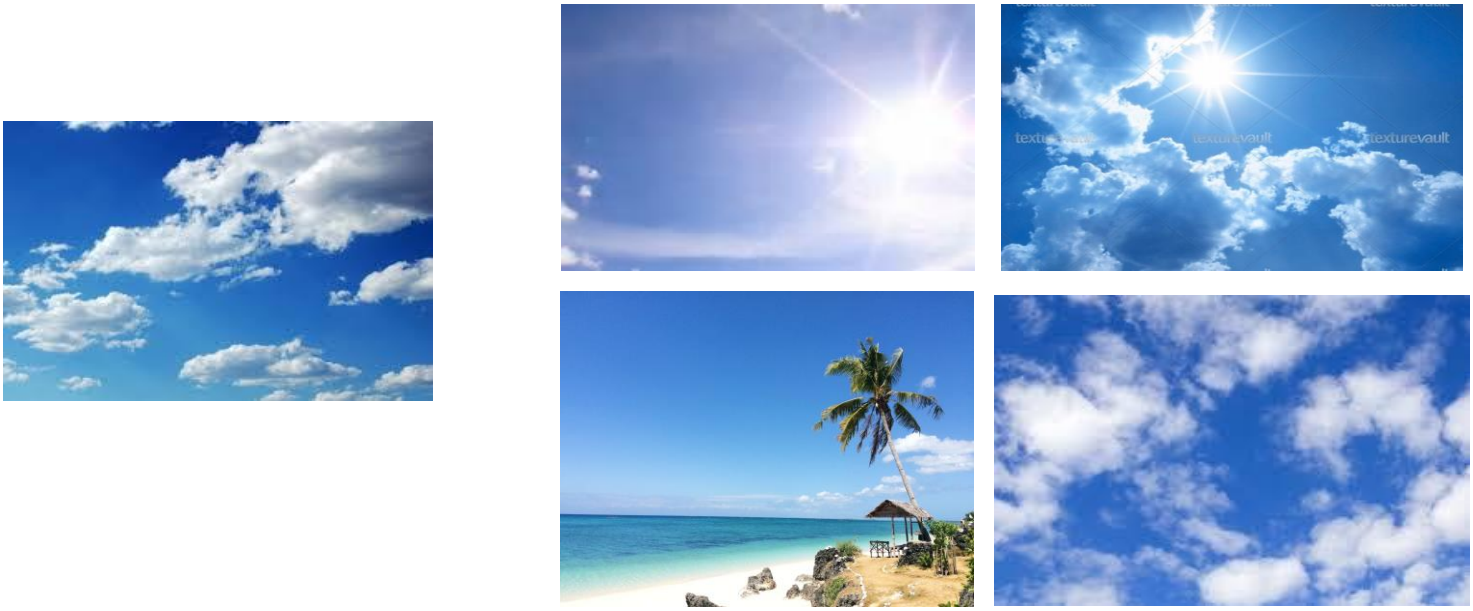
```
0.6218108387276904
```

Hình 8 - Kết quả khi dùng hệ màu HSV

Từ những kết quả trên, nhóm quyết định sẽ dùng mô hình trích xuất bằng histogram trong hệ màu RGB, số bin là [8, 8, 8] để xây dựng ứng dụng.

Dưới đây là một số kết quả thực tế mà hệ thống thu được khi chạy những mô hình trên.

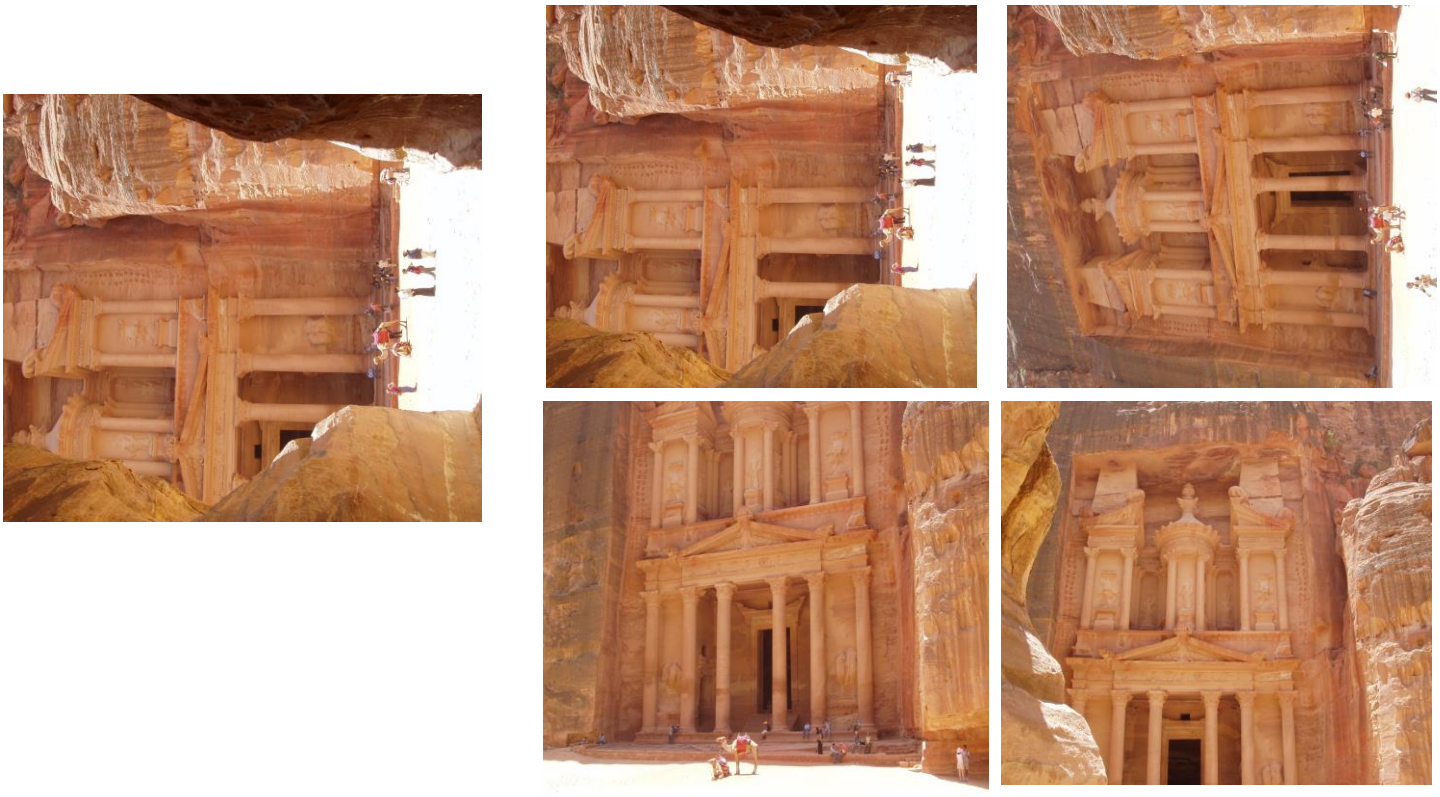
Hình 9 - Kết quả khi dùng tập dữ liệu Kaggle, dùng KMeans trích xuất



Hình 10 - Kết quả khi dùng dữ liệu Kaggle, dùng KMeans trích xuất

Hình 9 là hình có AP là 0.77 cho ảnh truy vấn.

Hình 10 là hình có AP là 0.013 cho ảnh truy vấn



Hình 11 - Kết quả khi dùng histogram, bin [8, 8, 8]

Hình 11 có AP là 1, tương đương với việc tất cả các ảnh được xem là có liên quan đều được trả về, và đồng thời những ảnh đó đều nằm ở những kết quả trả về đầu tiên.

Hình 12 - Kết quả khi dùng histogram, bin [8, 8, 8]

Hình 12 có AP là 0.39, kết quả trả về thấp hơn hình số 11 dù có cùng bộ trích xuất.

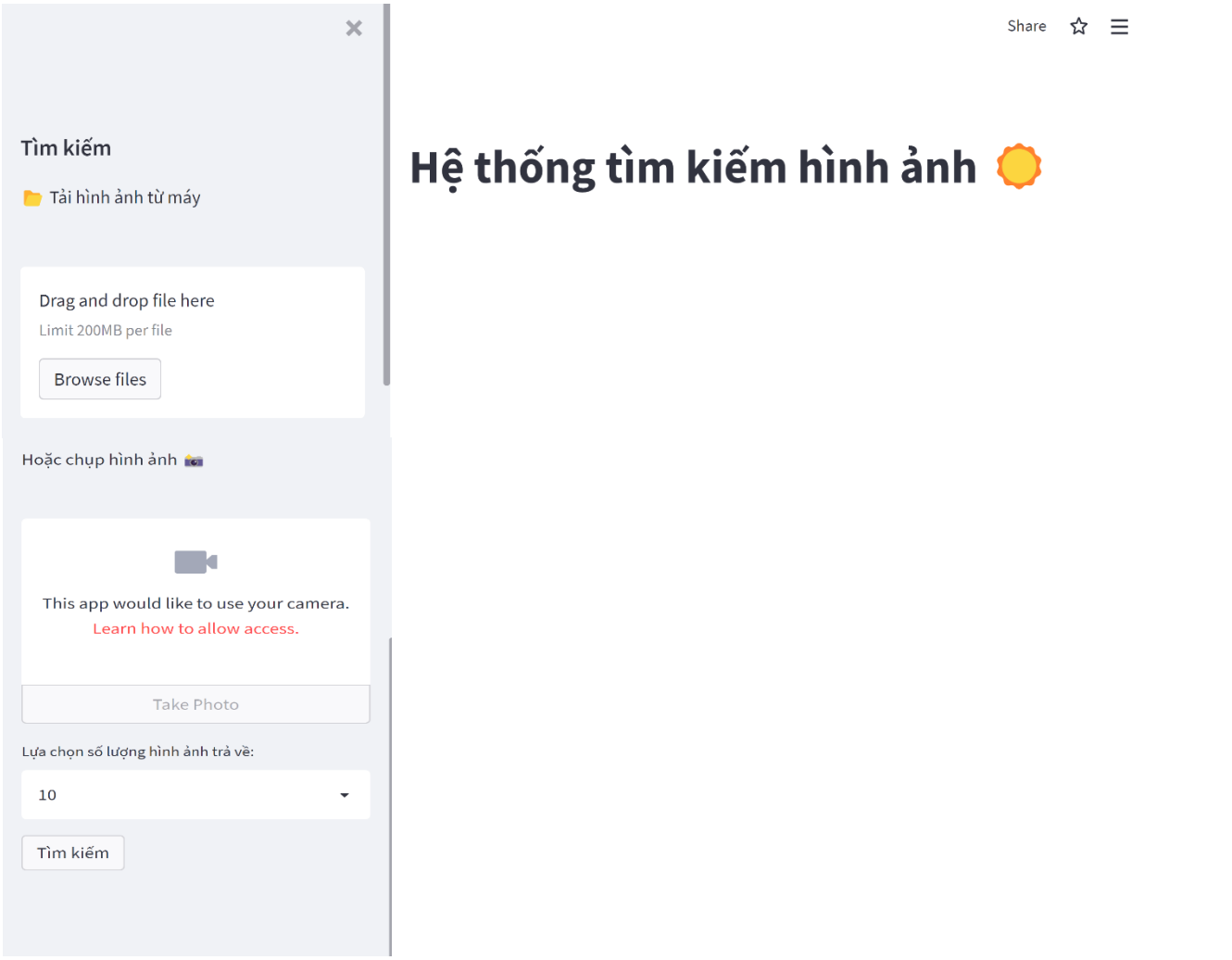
Theo những gì thu được, kết luận của nhóm là tuy đã có bộ dữ liệu Holiday thích hợp hơn cho bài toán này, tuy nhiên những kết quả trả về chỉ dựa trên màu sắc như vậy là không đủ, không hợp lý. Như hình 10 và hình 12, theo cảm quan về màu sắc thì những bức ảnh được trả về cũng đang có cùng tông màu sắc chủ đạo với ảnh truy vấn, tuy nhiên thì những ảnh đó được gán nhãn là không liên quan. Để cải thiện vấn đề này, nhóm tìm hiểu được là nên phát triển thành content based search engine, như vậy thì sẽ có thể trả về những kết quả tốt hơn, còn trong bài toán nhóm đang hướng tới là trả về những bức ảnh cùng tông màu, thì sau khi thay đổi phương pháp trích xuất sang dùng histogram, mAP của hệ thống đã cải thiện lên được 0.63. Tuy kết quả vẫn chưa gọi là quá tốt, nhưng đây là tất cả những gì nhóm đã có thể

tìm hiểu và thực hiện, nhóm sẽ cố gắng phát triển hệ thống này ở những môn học sau (nếu có thể).

5. Ứng dụng thực tiễn

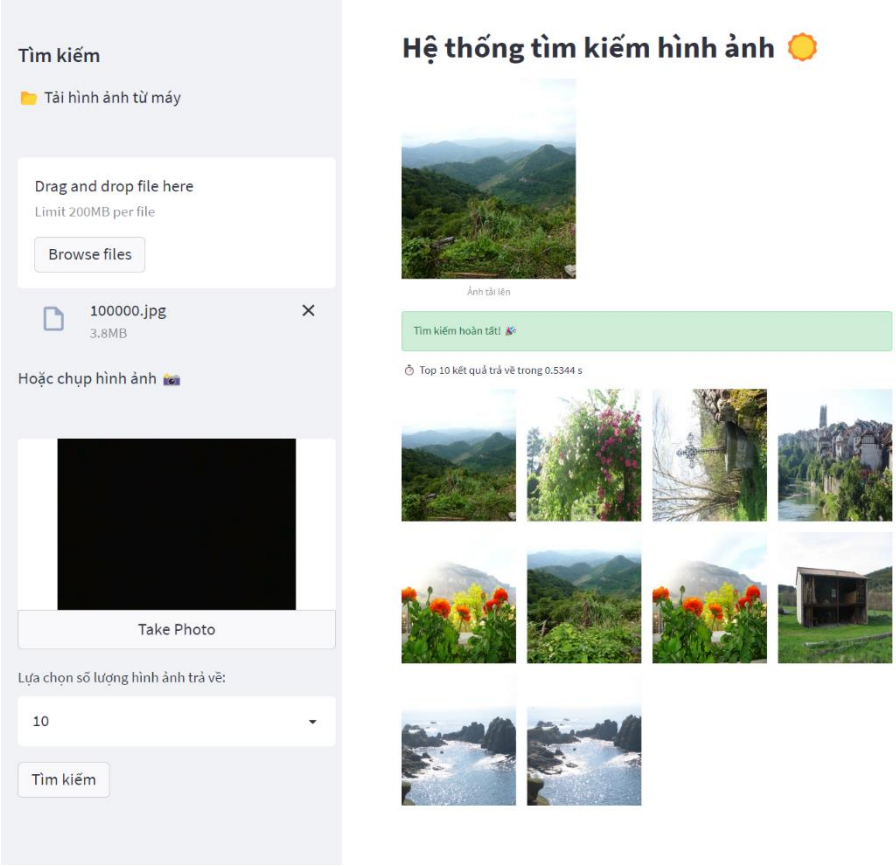
Link trang web ứng dụng của nhóm đã được xây dựng với nền tảng streamlit: https://share.streamlit.io/dxmai/cs231.m22.khcl/main

Trang web sẽ sử dụng mô hình tốt nhất đã trình bày ở trên (histogram, bin = [8, 8, 8]) và bộ dữ liệu Holiday để trả về hình ảnh.



Hình 13 - Trang web ứng dụng

Trên trang web này, người dùng có thể chọn hình ảnh từ máy hoặc cấp quyền để trang web có thể chụp hình ảnh từ thiết bị của người dùng (nếu không có ảnh nào được chọn, hệ thống sẽ báo lỗi). Ngoài ra, người dùng còn có thể chọn số lượng ảnh muốn trả về, nếu muốn tìm và trả về hết tất cả các ảnh trong bộ dữ liệu, chọn All.



Hình 14 - Kết quả chạy thử trang web

Ngoài ra còn các đường link liên quan đến đồ án như:

Google colab: https://colab.research.google.com/drive/Final_Project

Github: https://github.com/dxmai/CS231.M22.KHCL

6. Tài liệu tham khảo

[1]I. Memari, "A journey towards creating a color search engine," *Synthesio Engineering*, Nov. 26, 2019. https://medium.com/synthesio-engineering/a-journey-towards-creating-a-color-search-engine-194f1c388680 (accessed Jun. 27, 2022).

[2]"Multi-class Weather Dataset." https://www.kaggle.com/datasets/pratik2901/multiclass-weather-dataset (accessed Jun. 28, 2022).

[3]"Weather Image Recognition." https://www.kaggle.com/datasets/fceb22ab5e1d5288200c0f3016ccd626276983ca1fe8705ae2c32f7064d719de (accessed Jun. 28, 2022).

[4]"Download datasets." https://thoth.inrialpes.fr/~jegou/data.php#holidays (accessed Jun. 28, 2022).

[5]K. Bhanot, "Color Identification in Images — Machine Learning Application," *Medium*, May 24, 2019. https://towardsdatascience.com/color-identification-in-images-machine-learning-application-b26e770c4c71 (accessed Jun. 27, 2022).

[6]S. Feng, "Color Thief," Jun. 20, 2022. https://github.com/fengsp/color-thief-py (accessed Jun. 28, 2022).

[7]"How do you convert RGB to decimal? – Technical-QA.com." https://technical-qa.com/how-do-you-convert-rgb-to-decimal/#How_do_you_convert_RGB_to_decimal (accessed Jun. 27, 2022).

[8]"OpenCV Image Histograms (cv2.calcHist)," *PyImageSearch*, Apr. 28, 2021. https://www.pyimagesearch.com/2021/04/28/opencv-image-histograms-cv2-calchist/ (accessed Jun. 28, 2022).

[9]"Building image search an engine using Python and OpenCV," *PyImageSearch*, Dec. 01, 2014. https://www.pyimagesearch.com/2014/12/01/complete-guide-building-image-search-engine-python-opencv/ (accessed Jun. 28, 2022).

[10] J. Hui, “mAP (mean Average Precision) for Object Detection,” *Medium*, Apr. 03, 2019. https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173 (accessed Jun. 28, 2022).