Utilizing visual information to search relevant images using Content-Based Image Retrieval (CBIR) system

Sanghyun Lee
University of Michigan
shleec@umich.edu

December 12, 2019

Abstract

I explored to understand how we can utilize the extracted visual information (HSV color space) instead of any textual annotations to search relevant images. I used *Content-Based Image Retrieval (CBIR)* system that extracts features from and relies solely on the contents of images. I analyzed two different datasets, which consist of environmental presentations and object presentations, to learn how the content of images affect the accuracy of the results.

1 Introduction

How can we utilize the extracted visual information (color features) to help search for relevant or similar images?

Throughout many years, I was always interested in learning how to utilize visual information such as images, videos etc. Therefore, for my final project, I dissected image search engines to understand how it is possible to search relevant images without using any keywords, title or meta description. I analyzed two different datasets, which consist of environmental presentations and object presentations, to learn how the content of images can affect the accuracy of the results. Also, I focused on color features of photos to perform image searches.

2 Content-Based Image Retrieval system

When we talk about what an image search engine is, there tend to be three different approaches: search by meta-data, by image example or by a combination of the two. [1] Image search engines that extract features from and rely solely on the contents of images are called *Content-Based Image Retrieval (CBIR)* system. In past years, CBIR model have been a major topic of research and have been explored in many different approaches. [4]

Content-Based Image Retrieval (CBIR) is a method based on feature signatures [2], which comprise an expressive summary of the content of a multimedia object that is significantly more compact than the object itself, allowing for an efficient comparison between objects. It quantifies given images based on a defined image descriptor [1], and stores the feature vectors into a database. When a query image, or an input image is given, the model compares the similarity of the query and the stored photos by means of a distance function. [2] In order to use Content-Based Image Retrieval system to build a simple image search engine, I used `Python` and `OpenCV`, which is an open-source BSD-licensed computer vision library.

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

2.1 The 4 steps of CBIR system

When constructing a Content-Based Image Retrieval system, there are typically four distinct steps: Define image descriptor, index the dataset, define similarity metric and search relevant photos based on query image.

2.1.1 Define image descriptor

First, we need to decide what specific feature of the image content we want to extract to store and compare. Therefore, we need to define *an image descriptor* [1], which governs how the image is represented and quantified. We can use RGB colors, shapes of objects or texture of images to extract the visual information. [3]

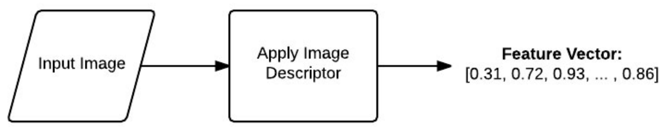

Figure 1: The pipeline of image descriptor, (2014) *The complete guide to building an image search engine with Python and OpenCV*

2.1.2 Index the dataset

After defining the image descriptor, we now need to apply the image descriptor to each image in our dataset and describe the certain property of image numerically and store the extracted vectors in a new database for similarity comparison with the query image. Typically, the numbers of extracted features, called *feature vectors* [2] are indexed in a CSV file, RDBMS, etc. [1]

2.1.3 Define similarity metric

With the stored feature vectors, we define similarity metric, which will help us to compare visual similarity between the query and stored images. The popular choices are Euclidean distance, cosine distance and chi-squared distance. [3] Since the metric is decided based on the type of the dataset and the types of extracted features. [1]

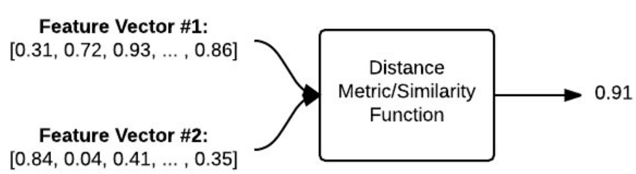

Figure 2: Comparing two feature vectors based on distance metric/similarity functions. Adrian Rosebrock, (2014) *The complete guide to building an image search engine with Python and OpenCV*

2.1.4 Search relevant photos based on query image

Now, it is time to actually search the relevant images. When a user submits a query image to the Content-Based Image Retrieval system, it extracts visual information using the same image descriptor. Then, it applies the similarity function to compare the feature vectors of query to the indexed vectors. Based on the similarity metric or the similarity function, it returns the most relevant results. (Figure 3.) [1]

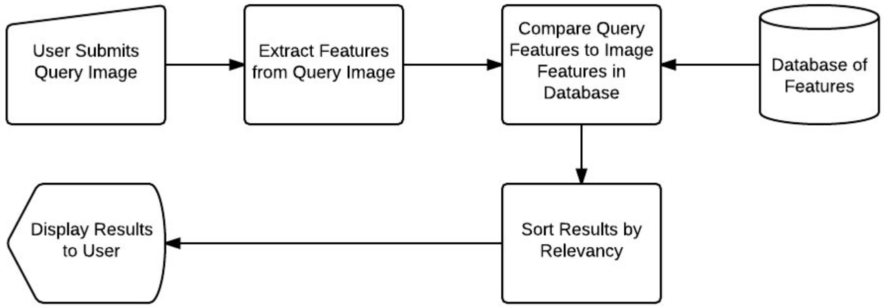

Figure 3: Performing a search on a CBIR system. Adrian Rosebrock, (2014) *The complete guide to building an image search engine with Python and OpenCV*

3 Datasets

After I decided to use color features to extract and implement the CBIR system into my image search engine, I found two different datasets: *INRIA Holidays* [5] (environment-based photos) and *Stanford Dogs Dataset* [6] (object-based presentation photos) to explore if the different types of images and visual information affects the result of relevant photos.

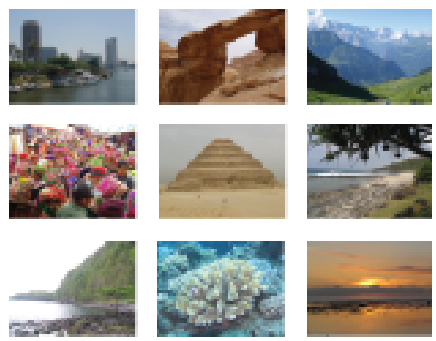

Figure 4: INRIA Holidays dataset

3.1 INRIA Holidays

The *INRIA Holidays* dataset is a set of 1491 JPEG images that contains holiday photos including a variety of scenes, object and landmarks around the world. It includes natural, man-made, water and fire effects types of images. However, this does not have any distinct meta-data or titles that can be used to search by keywords. [5]

3.2 Stanford Dogs Dataset

The *Stanford Dogs Dataset* contains images 120 breeds of dogs from around the world. It has around 150 images per breed and in a total of 20,580 images. [6] This dataset also didn't include any meta-data or keywords embedded in each image.

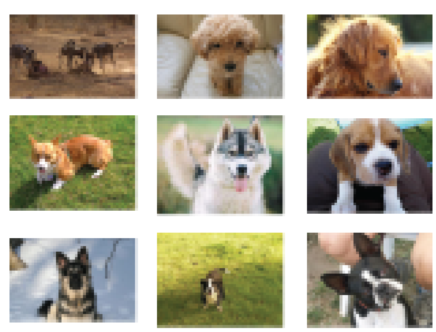

Figure 5: Stanford Dogs Dataset

However, for the simplicity of the data processing, I scaled down the number of photos to a total of 1480. Also, since I wanted to measure how well the CBIR search image using color features can perform to find the same breed as the query image. Therefore, I used a condensed dataset, which includes a total of 9 breeds with about 150 images each: African hunting dogs, Toy poodle, Pembroke Corgis, Eskimo dogs, German shepherds, Border Collies, Golden Retrievers, Beagles and Boston Bulls.

4 Implementation

Image descriptor When we consider to "similar" images, it means different based on the content of images. For instance, if we were to use dog photos, we would expect the CBIR model to find same or similar breed. However, when we were to find relevant scenic photos, we would want to find images with similar atmosphere as the query image. Instead of relying on multiple features, I decided to rely on only color features of the images in my datasets and to understand how this approach can affect the results.

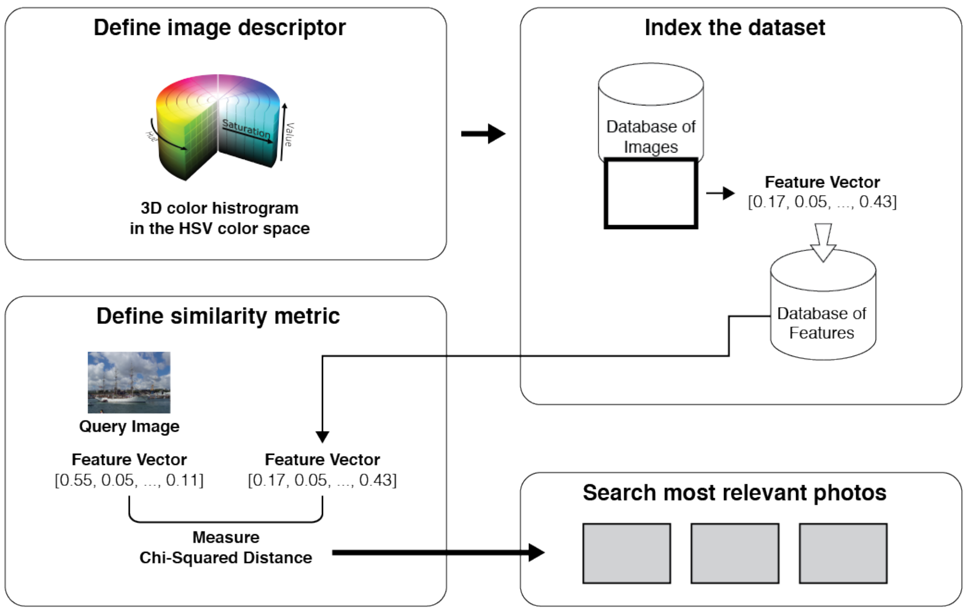

Figure 6: The brief process of CBIR system

To make a more powerful and robust color feature based CBIR, I defined a 3D color histogram in the HSV color space (Hue, Saturation, Value) to be my image descriptor. While RGB (Red, Green, Blue) values are simple to understand, the RGB color space fails to mimic how humans perceive colors. [1] Therefore, I utilized the the color distribution of HSV color space, which maps pixel intensities more robustly. Lastly, I did some experiment with the number of bins of my color histogram descriptor and used 8 bins for the Hue channel, 12 bins for the Saturation channel and 3 bins for the Value channel, yielding a total feature vector of 288 dimension.

Figure 7: Region-based of Image

Index the dataset Like how Adrian Rosebrock from pyimagesearch [1] suggested, I divided each image into different regions and applied my image descriptor to each region instead to the entire image. (Figure 7.) By computing color histogram based on regions rather than globally, I was able to simulate locality in a color distribution. For instance, if we were to use global-based image descriptor, it would be difficult to determine where in the image the blue colored sky is or where the yellow sand is. Therefore, I divided each of my image into five different regions: the top-left corner, the top-right corner, the bottom-right corner, the bottom-left corner and the center of the image.

However, when I used Stanford Dogs Dataset to perform the search, I instead focused on the center region. As I mentioned earlier, the goal of using the dog images was to find out if CBIR can successfully search same or similar breed. Therefore, it was logical to focus on the object of images, which is typically located in the center of images, instead of the environment. (Figure 8.)

Figure 8: Region-based of Image

Lastly, applying the image descriptor to each image in the datasets, the CBIR system creates a CSV file and writes the feature vectors with IDs of corresponding images.

Similarity metric For the similarity metric, I had many different choices to evaluate if the photos are relevant or not. Since I decided to use histograms of HSV color space, I defined my similarity function using *chi-squared similarity*. As this metric is used to describe the distribution of a sum of squared random variables [7], I utilized it to compare the probability distribution of color histogram of each images.

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

Figure 9: Chi-Squared Formula.

Search relevant photos When a query image is submitted to the CBIR search engine, the system will loop over every single feature vectors written in the CSV file and compare to the vectors of HSV color space histogram of the query. Then, the system returns the five most relevant images after applying the chi-squared similarity function with the lowest value

4.1 Script details

In order to implement the Content-Based Image Retrieval system, I used four different `Python` scripts using `OpenCV` library. The first script is called `colordescriptor.py`, which is dedicated to perform the image descriptor. This script stores `ColorDescriptor` class, which extracts the HSV color histogram features from each image in the dataset. It extracts a 3D color histogram from the five segmented masked region of the image, using the number of bins per channel.

The second script is called `index.py` script. It initializes `ColorDescriptor` from `colordescriptor.py` and indexes the features of the stored dataset photos. Also, it constructs the argument parser and creates path to the dataset directory so that the system can grab feature vectors with corresponding images when matched with the query image.

The `measure_distance.py` is used to compare distance between two features vectors of the queried image and indexed photos in the dataset. Then, it produces the five most relevant images with the lowest sum value.

Lastly, the `search.py` is the script, which actually perform the "search" when a query image is submitted. It first extracts the feature vectors from the query using `ColorDescriptor`. Then, it compare the distance of the feature vectors to find the five most relevant images. After finding the photos, it shows the results to the user.

The code in this project was modified and was originally demonstrated in an article called, *The complete guide to building an image search with Python and OpenCV*. [1]

5 Results

After defining my image descriptor, indexing my datasets and setting the similarity metric, I ran my Content-Based Image Retrieval search engine with preselected query images.

INRIA Holidays From the INRIA Holidays photos, I tested with six different scenery photos I randomly selected. One of query images was a sunset scene as shown in Figure 9. Since I divide five different regions to compare the feature vectors, the bottom half of output images are very dark and have a mixture of orange and yellow colors. Also, the top half of results has a gradient color from dark blue to light blue. As the images of INRIA Holidays are environment focused photos, the outputs have similar atmosphere and mood as the query image.

Figure 10: The result of using INRIA Holidays dataset (An example of environmental-based photos

Looking at the Table. 1, it shows the actual Chi-squared similarity value of results. The measures show that the first image has a very low Chi-squared sum value, whereas next fours images have much larger value compared to the first one. Even though actual output images look very similar in terms of color distribution, the sum values are surprisingly larger than expected.

Table 1: Chi-squared similarity measures of 127502.png query image

Result Image	Chi-squared
127503	0.23
126500	2.94
126400	3.42
127501	4.16
127500	4.53

Stanford Dog Dataset From the Stanford Dog Dataset, I selected one image from each breed. Therefore, I had one each from African hunting dogs, Toy poodles, Pembroke Corgis, Eskimo dogs, German shepherds, Border Collies, Golden Retrievers, Beagles and Boston Bulls as the query images. As you can see from Figure 10., even though the color distribution of brown and white in the Pembroke Corgi image is similar to the color distribution of the result images, the actual result of breeds does not match with the query image. The output shows two Pembroke Corgis, two Golden Retrievers and one Toy Poodle. Not even a half of the most relevant images was not actually "similar" in terms of defining objects. However, because I focused on the content in the center of the image, I could at least find similar colored dog images from the dataset.

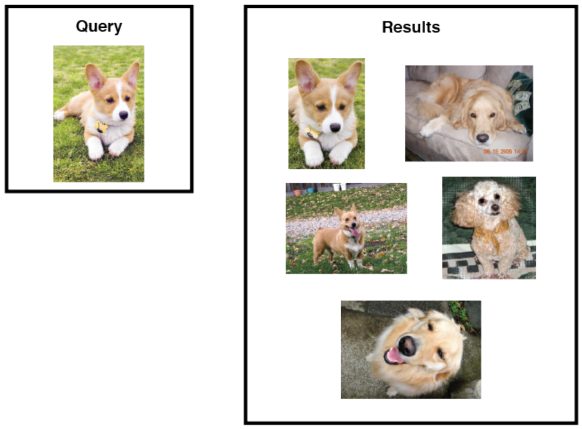

Figure 11: The result of using Stanford Dog Dataset (An example of object-based representation photos

Looking at Table. 2, which shows the Chi-squared similarity measures of Pembroke Corgi query image, I found an interesting fact. Compared to the similarity measure of the Holidays results, the measures of the results of the Corgi have much lower value. Even though it doesn't feel like the CBIR search engine found the relevant images due to the difference in breed, they have much lower sum value of errors.

Table 2: Chi-squared similarity measures of Pembroke Corgi query image

Result Image	Breed	Chi-squared
n02113023_3913	Pembroke Corgi	0.67
n02099601_7304	Golden Retriever	1.35
n02113023_4893	Pembroke Corgi	1.38
n02113624_3103	Toy Poodle	1.43
n02099601_544	Golden Retriever	1.44

6 Conclusion

The CBIR system demonstrated its advantage of searching images with similar visual information when the image descriptor was set properly. The bins of color histograms had to be tuned to set the right density of pixel intensities in an image. The image search engine using this system was able to find similar mood or tone of images, because it solely relied on the whole content of the image. Therefore, it produced a great result when using environmental-based images.

However, using the Content-Based Image Retrieval to search the most relevant images, I found a limitation of finding same or similar object. When analyzing Stanford Dog Dataset, which contains object-based presentation photos, the accuracy of searching a same breed was very low. Even when I specifically target the area of where the object is located in the photo, it could not find the same breed.

Acknowledgments

I would like to thank professor Paramveer Dhillon and GSI Yulin Yu for guiding me in the SI 671 Data Mining course.

References

[1] Adrian Rosebrock (2014) The complete guide to building an image search engine with Python and OpenCV *https://www.pyimagesearch.com/2014/12/01/complete-guide-building-image-search-engine-python-opencv/* Pyimagesearch

[2] Christian Beecks & Tomas Seidl (2013) Distance based similiarity models for content based multimedia retrieval. *PhD thesis, Aachen, 2013. Zsfassung in dt. und engl. Sprache; Aachen, Techn. Hochsch., Diss.*

[3] Preeti Chouhan, & Mukesh Tiwarii (2015) Image Retrieval Using Data Mining and Image Processing Techniques *International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering Vol. 3, Issue 12* IJIREEICE

[4] Ioannis Brilakis & Lucio Soibelman (2005) Content-Based Search Engine for construction image database. *Automation in Construction, Volume 14, Issue 4, August 2005, Pages 537-550*

[5] Herve Jegou, Matthijs Douze, & Cordelia Schmid (2008) Hamming Embedding and Weak geometry consistency for large scale image search. *Proceedings of the 10th European conference on Computer vision*

[6] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, & Li Fei-Fei (2011) Novel dataset for Fine-Grained Image Categorization *First Workshop on Fine-Grained Visual Categorization (FGVC), IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* Stanford University

[7] Robert Kissell, & Jim Poserina (2017) Chi-squared Distribution *Optimal Sports Math, Statistics, and Fantasy* ScienceDirect