MO446 – Introduction to Computer Vision
Project 4

Breno Leite
Guilherme Leite

26/10/2017

Question 3 - Image Descriptor

To create a retrieval system we need to have image descriptors, in this question we extracted these descriptors for all the images in the dataset. This descriptors were saved in the disc, so the next time the program was executed it could load them, and perform faster, instead of recalculating every descriptor which is a high cost operation.

An image descriptor consist of all the regions in an image and their information. To extract these regions, we firstly used OpenCV's implementation of Kmeans algorithm, to creoia ate a k-colored image. Figure **??** shows the result of this image.

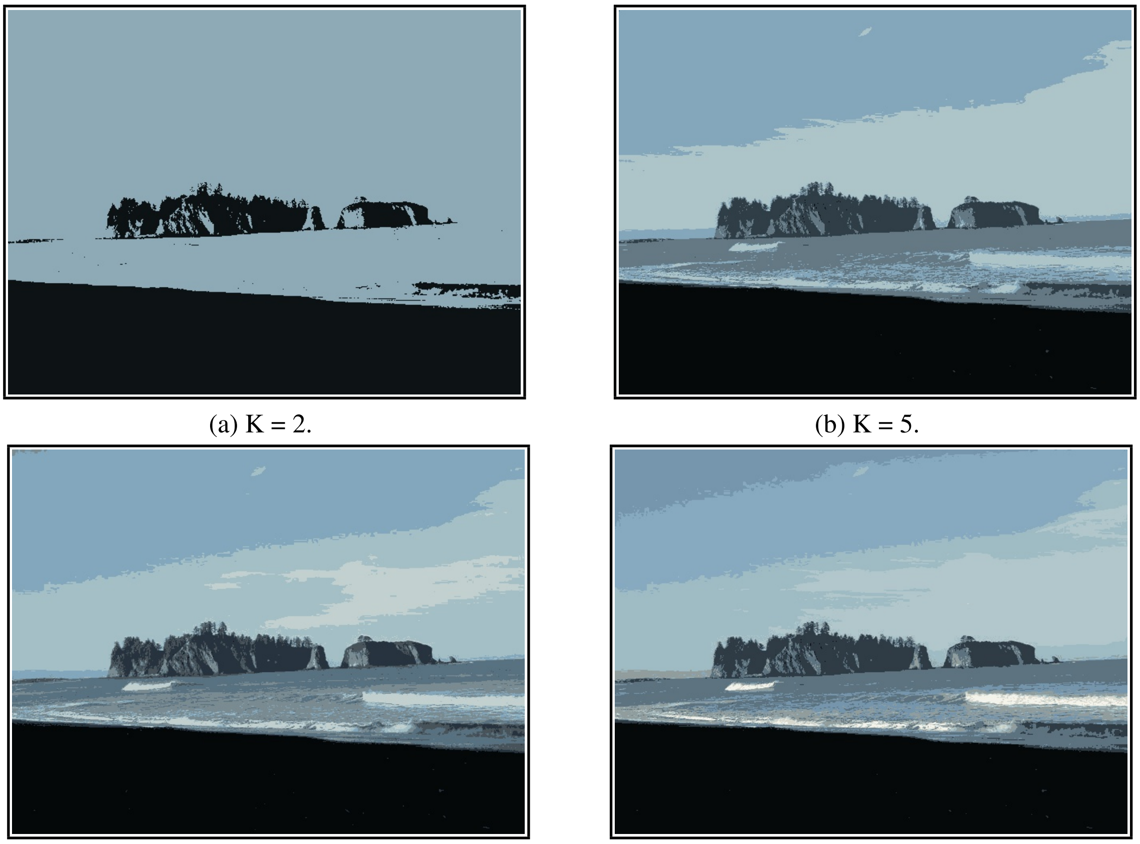
The algorithm returns K regions extracted based on their pixel intensity, in Figure **??** each color represents a region. The number of regions (K) was selected by trial and error, a small K yields big generic regions, as in Figure **??** for example the sky and the ocean were blended into a single region, which is not helpful to discriminate between images. In contrast a big K yields to many small regions as seen in Figure **??**, small regions don't increase matching accuracy, the reasoning about it will be explained later on, although they increase the computation time significantly. A $K = 5$ yielded to better results, as seen in Figure **??**, and the rest of the work will be using this value.

A comparison between the sky in Figure **??** and Figure **??** shows a reasoning for the better results using $K = 5$, in Figure **??** it is possible to distinguish around two colors for the sky region, a light and a dark blue, as in Figure **??** the sky was sub-divided into roughly three regions, but a pixel by pixel analysis reveals more regions describing the sky which won't really improve the matching process and would make the algorithm less time efficient.

The ideal K value can vary from image to image, and there is a method to choose the best value depending on a set of tests, this method is called silhouette analysis and we did not explored this automatic approach in this project.

In order to use these regions as an image descriptor, it is desired that descriptive enough regions. The regions returned by Kmeans are still too generic, some of them are even disconnected,

Important note: *The borders seen in the figures are not part of the image, they are figurative information about the starting and ending points of the image. Moreover, all the image scales in this report were changed in order to make the text more readable.*



(a) K = 2.

(b) K = 5.

(c) K = 8.

(d) K = 10.

Figure 1: K-colored image formed by the kmeans algorithm

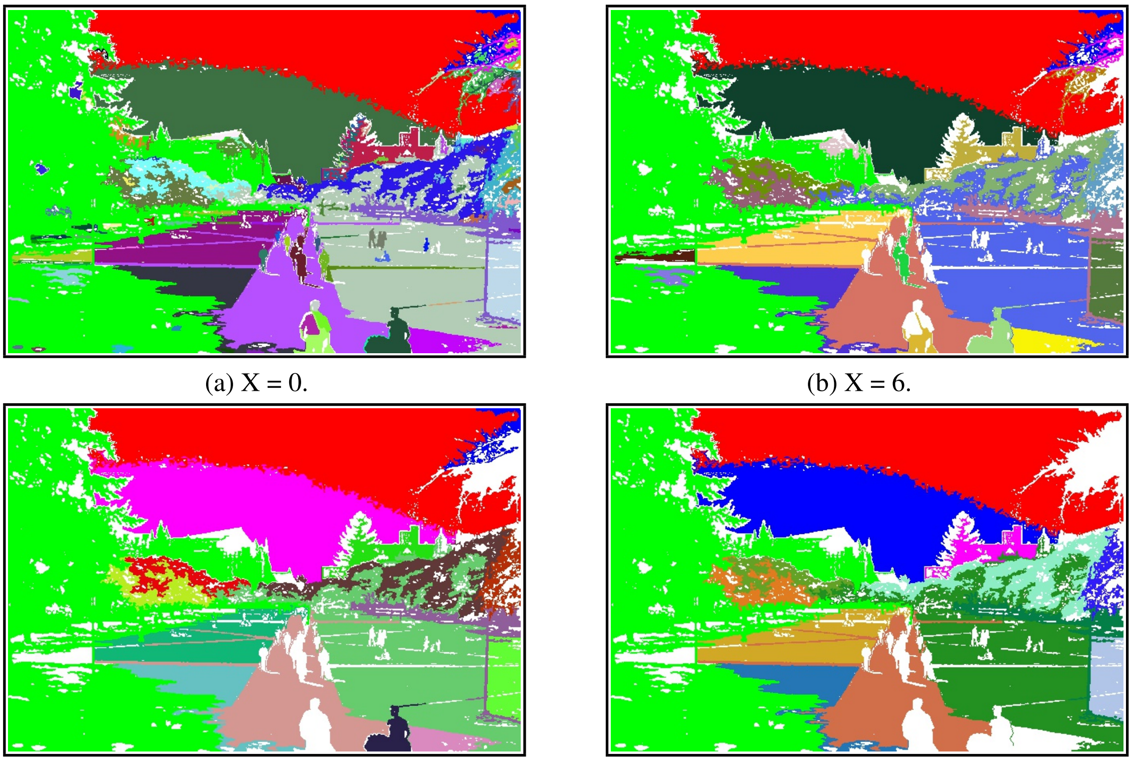
as in Figure **??** the sky and some waves are in the same region, even though they are disconnected and represent different objects in the scene.

To eliminate these disconnected regions we ran a Breadth-first search (BFS) in the entire image. The BFS obtains the connected regions separately, which receives a new label, eliminating disconnected regions altogether, Figure **??** shows the new regions generated by the algorithm. The borders remained unchanged during this process, but reducing their noise could improve the expression of some regions, increasing the accuracy of the solution.

In Figure **??** each color represents a connected region and Figure **??** shows their bounding boxes. Regions that are too small don't really describe enough of a feature to be relevant, and thus are discarded, in Figure **??** the white color is used to represent the discarded regions.

A region is considered too small if its size is smaller than the multiplication of a factor X by the average region size, this value was chosen based on our analysis of the yielded results given by the input dataset.

In Figure **??** and Figure **??** the value of X is set to zero and it is possible to observe regions as big as one pixel line being taken, which is counter productive towards our goal as they don't describe something relevant. In Figure **??** and Figure **??**, with $X = 6$, smalls non-discriminative regions are still being selected, like the tree on the right upper side of the image, the human carrying a bag in the bottom center (Figure **??** shows that only the bag and pants were selected) and a few smalls patches of grass in the left bottom side, all of those describes non-discriminative features



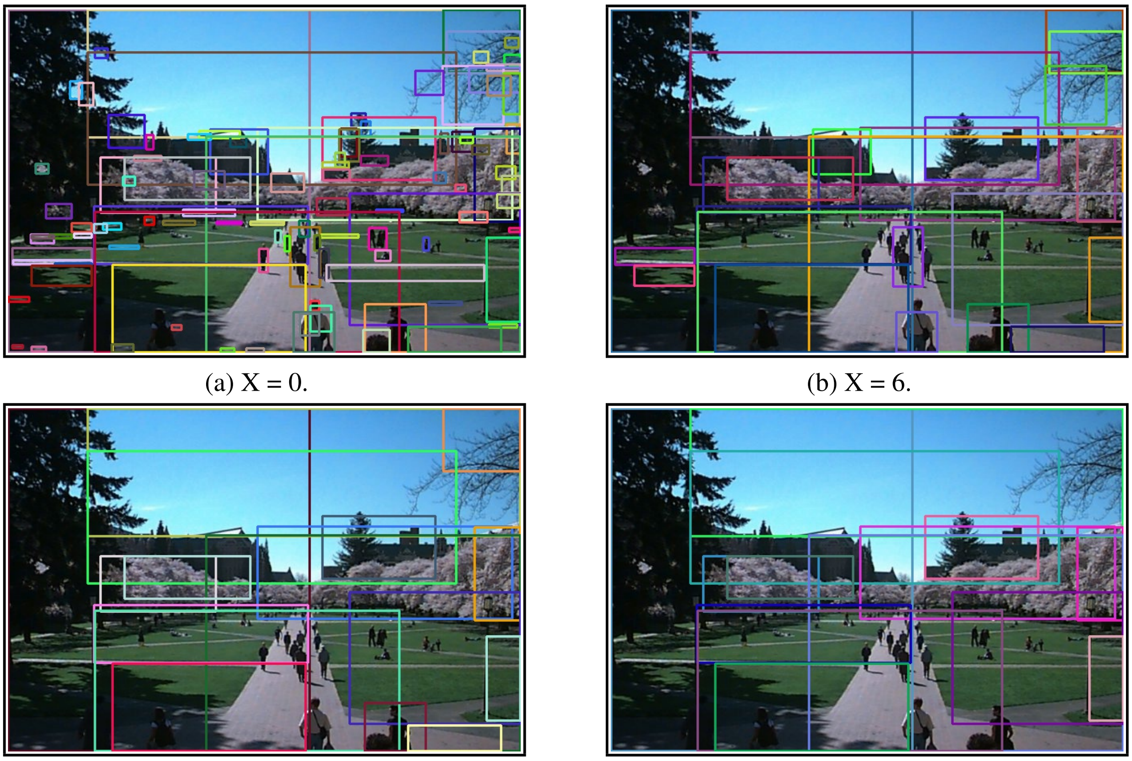
(a) X = 0.

(b) X = 6.

(c) X = 10.

(d) X = 20.

Figure 2: Connected regions extracted using the BFS explorer for different X values.



(a) X = 0.

(b) X = 6.

(c) X = 10.

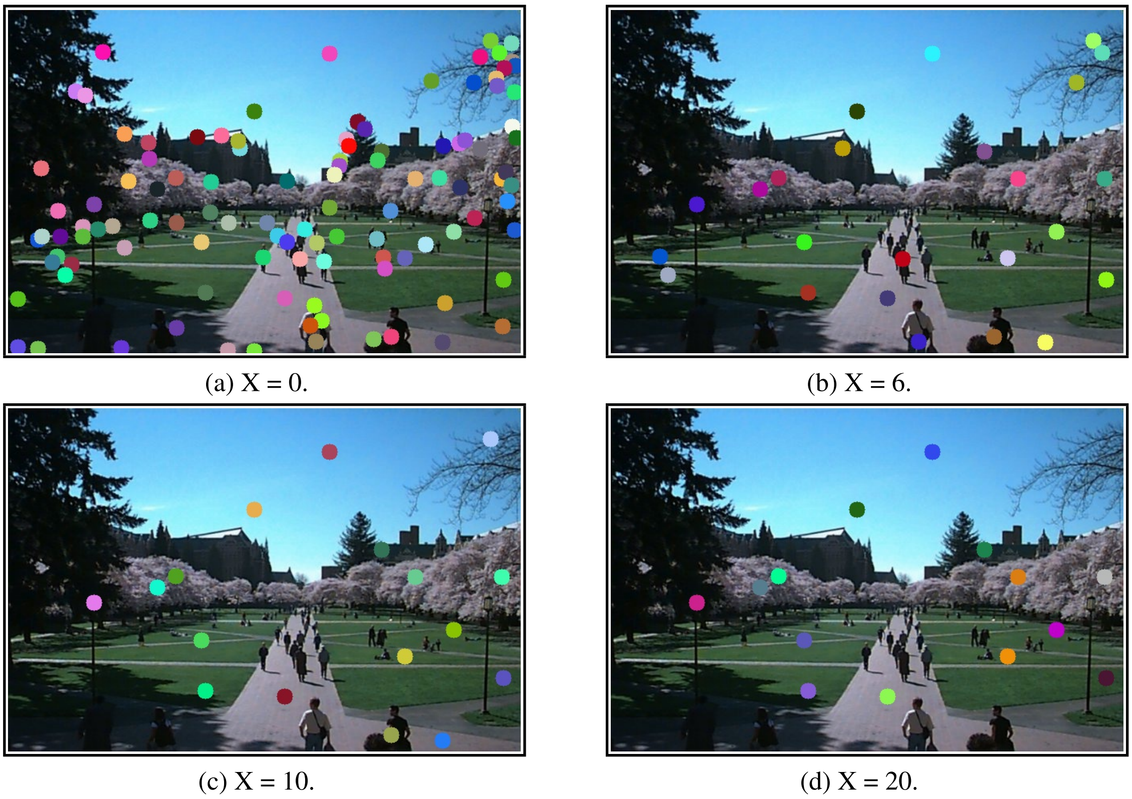
(d) X = 20.

Figure 3: Bounding box of each region calculated with different X values.

that could hurt the final result.

In Figure **??** and Figure **??**, where $X = 10$, there are still some non-discriminative regions being selected, like half of the tree in the right upper corner, but they are few and contrasted with a majority of somewhat big, discriminative regions, like the cherry trees, the buildings in the back and the path in the center. In Figure **??** and Figure **??**, with $X = 20$, almost all the small ignorable regions were discarded and it performed well in this image, but since it was a big value it could start eliminating discriminative regions on other images, this and the inefficiency of the small values to determine good sized regions was the reason why we chose $X = 10$.

These regions will be used as descriptors of the image, and compared later on, to do so it is necessary to describe each region, in a data structure, a feature vector with the following information were used: region size, region mean color, region centroid, and some texture features for the region patch. The patch of the texture features were extracted using the delimitations of their bounding boxes and are composed of: contrast, correlation, dissimilarity, energy and entropy. These components were extracted using the co-occurrence matrix, applied to the region patch, and they tend to describe correlation between pixels in the region, Figure **??** illustrates the centroids position of each region.



(a) X = 0.

(b) X = 6.

(c) X = 10.

(d) X = 20.

Figure 4: Centroids of each region calculated with different X values.

Question 4 - Distance Measure

In order to create a content-based image retrieval, we developed a distance measurement for our collected descriptors that are based on the regions obtained by the connected components algorithm. The distance formula used was a simple difference method over the features size, mean color and the textures features for each region. Moreover, euclidean distance was used to compute the centroids distances.

Every region is described by a single value that is composed by the average between all the differences found in the features descriptors for that region. Two different average processes are taken, the first one between the feature vector for the textures obtained by the co-occurrence matrix, and, the other one is an average operation between all the features for that region. Equations **??** and **??** shows the equations for the texture features, and the average of all features, respectively.

$$feat\_weight = \frac{contrast + correlation + dissimilarity + energy + entropy}{5} \tag{1}$$

$$region\_distance = \frac{size + mean\_color + centroid + feat\_weight}{4} \tag{2}$$

As images might contain different number of regions, a region is compared to all others and the minimum distance is chosen to represent the region. After determining all the distances for the regions, a average is performed between them, giving a single value to classify the similarity between both images. Smaller values mean more correlation, while higher values describe more distinctive images. These averages are important, they normalize the proportion of each feature. Figure **??** shows a result for some queried images.

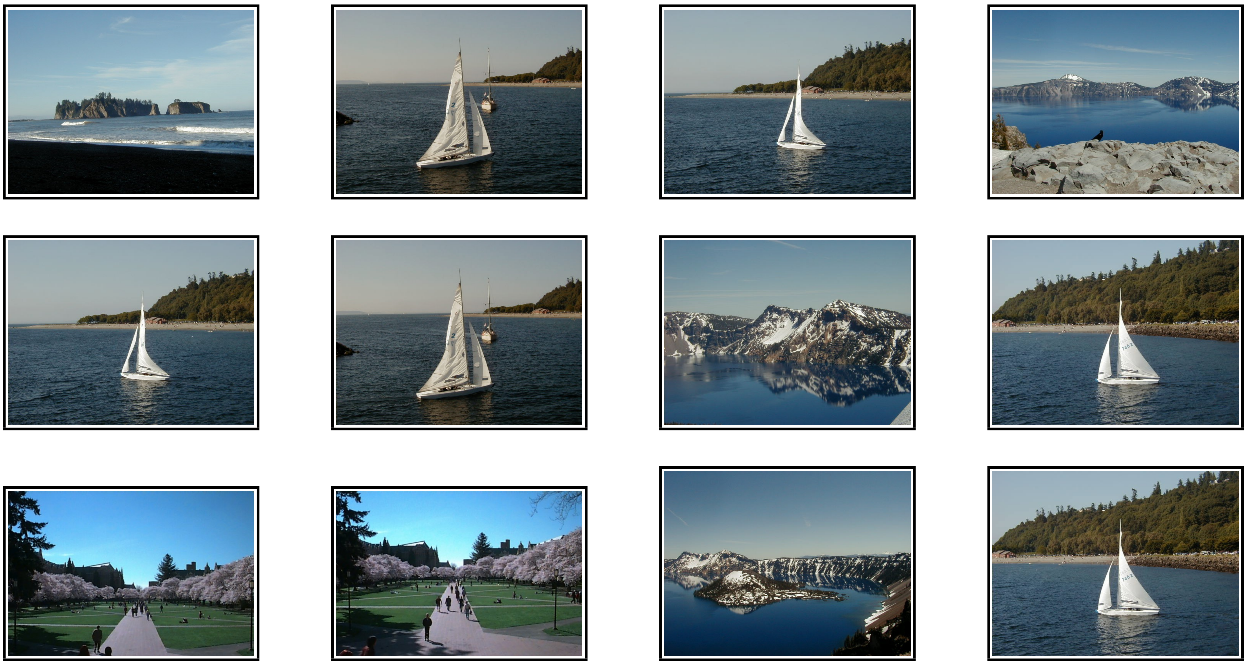
Figure **??** shows that in some cases the results are pretty good, for example for the query (boat_5) which is represented by the second row, two of the images has a boat in it. Moreover, the second ranked image, is also composed of a land and a river. However, in the third row the second ranked (crater_3) image is not so similar to the queried image (cherry_3).

Those results shows us that large regions are being largely used for comparing the images, however, it is hard to choose the best features to use. In this way, in order to create a more robust system we decided to change the simple average to a weighted average (in both averages processes), this way we could optimize the weights for the average choosing the best weights for our dataset. Equations **??** and **??** shows the new average processes.

$$feat\_weight = \frac{w_1 * contrast + w_2 * correlation + w_3 * dissimilarity + w_4 * energy + w_5 * entropy}{w_1 + w_2 + w_3 + w_4 + w_5} \tag{3}$$

$$region\_distance = \frac{w_6 * size + w_7 * mean\_color + w_8 * centroid + w_9 * feat\_weight}{w_6 + w_7 + w_8 + w_9} \tag{4}$$

In order to optimise these weights, the mean distance between one specified image with all images in the same class were used. We searched in a space between 0 and 2 for each feature, as we have 9 values a total of 19683 different weights were tested and the following weights had the



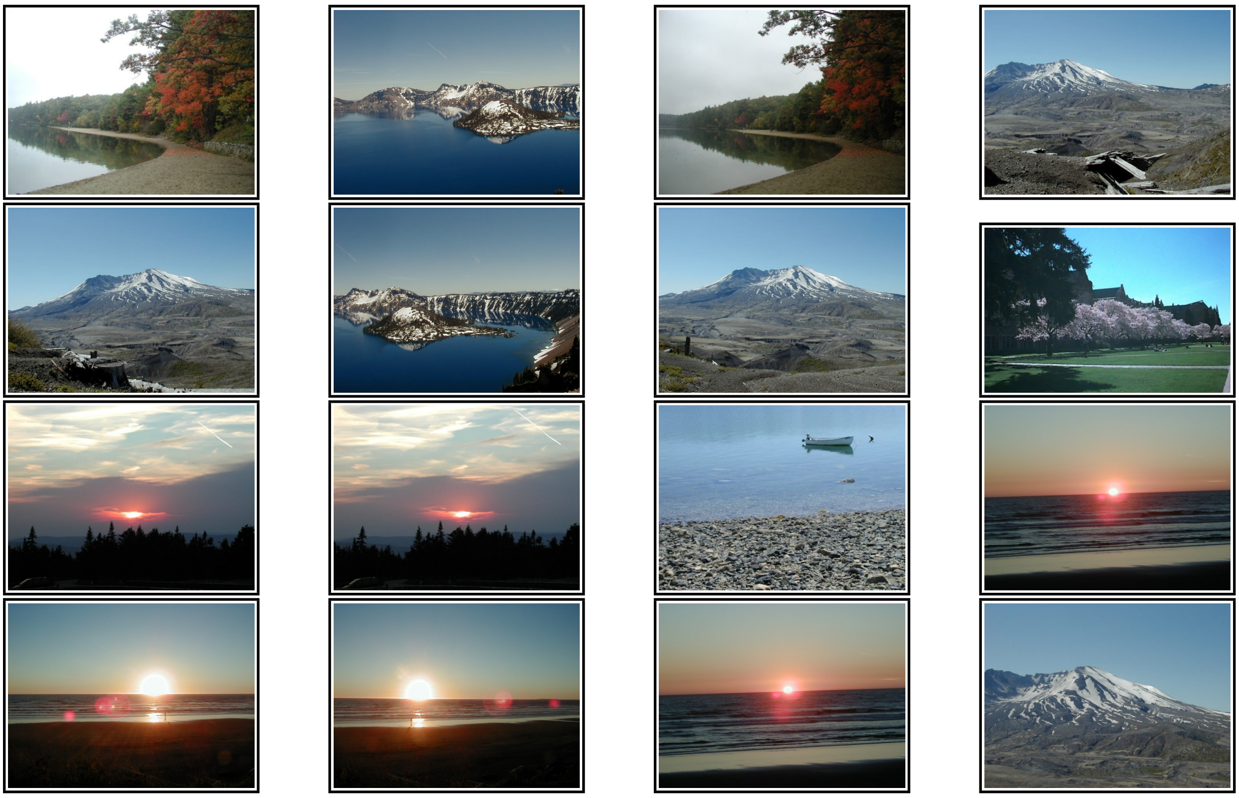
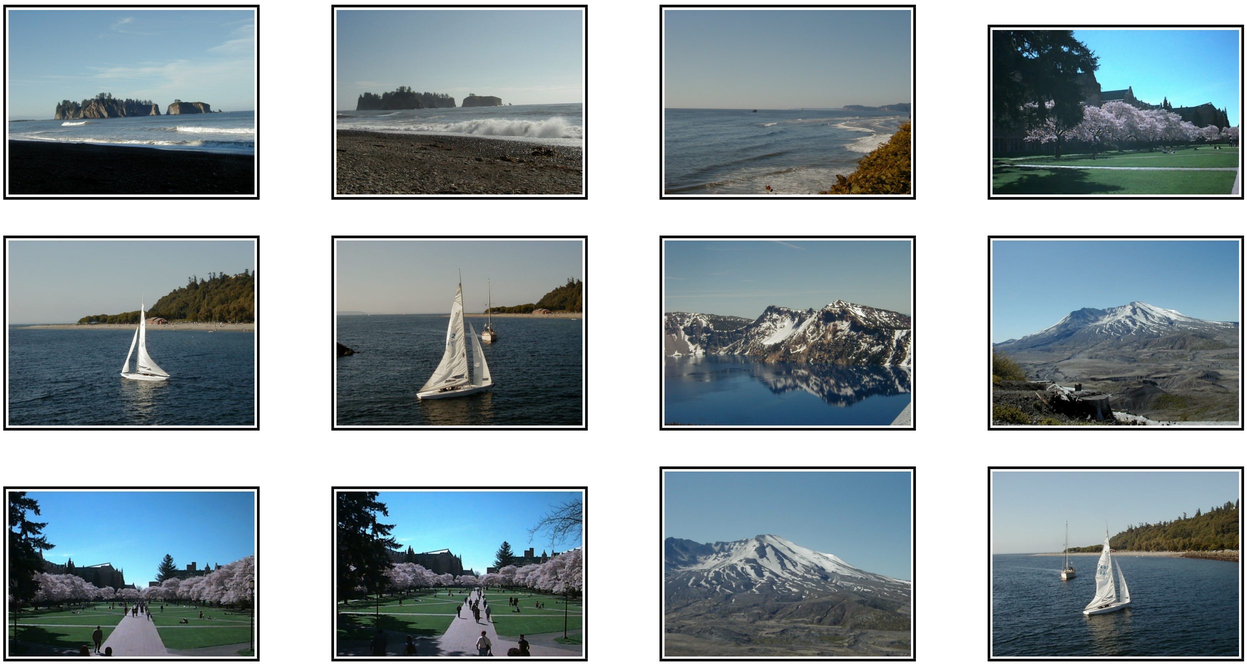


Figure 5: Retrieved images using the left column as queries, in order the columns are Queried image, 1^{o} rank, 2^{o} rank, and 3^{o} rank.

best results (smaller mean distance for images of the same class): $w_1 = 0$, $w_2 = 0$, $w_3 = 0$, $w_4 = 0$, $w_5 = 1$, $w_6 = 1$, $w_7 = 1$, $w_8 = 1$, $w_9 = 1$. These weights shows that the optimisation decided that some of the co-occurrence features were not improving the overall result, the only feature used was entropy which seems to not help a lot the algorithm to not be based just on color intensity. Figure **??** shows the retrieved images for the same queries as before.

Figure **??** shows some improvements for some classes as the beach class, however, it is still



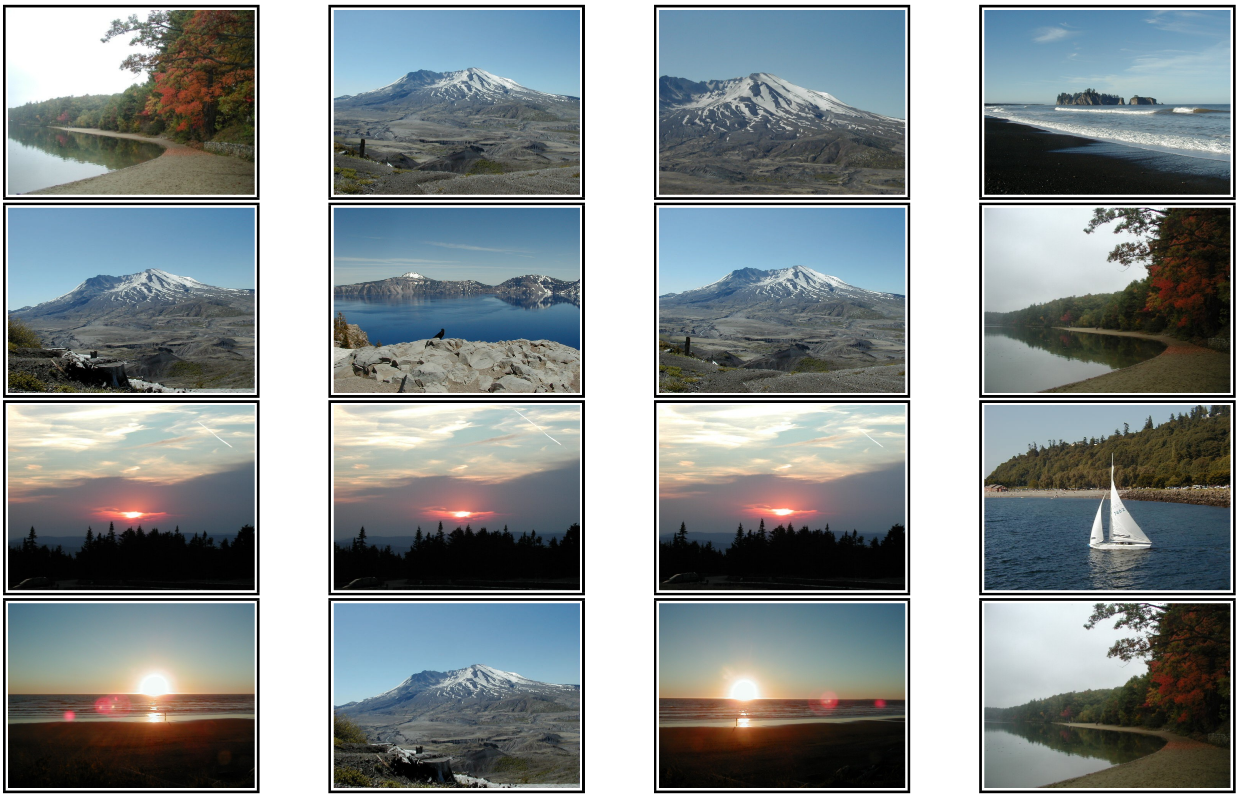


Figure 6: Retrieved images using weights, in order the columns are queried image, 1^{o} rank, 2^{o} rank, and 3^{o} rank.

guessing some weird results like the third rank for the beach query (cherry_5). It is also clear that the system is retrieving more similar images in the higher ranks compared to the other approach. Another important note is that the algorithm is highly related with colors, which some times confuses different image because of a large amount of blue sky or ocean.

In order to retrieve better results, we could have searched for betters weights in a larger space domain. However, this search is highly computer expensive. A better approach would be to use different features extractor, which are not too related with colors. A SIFT or HOG descriptor for each region could be applied, but it would turn the algorithm more time consuming.