PIV
Content Based Image Retrieval (CBIR) - Histogram based descriptors

Joana Justo Guillaumet
joana.justo@estudiantat.upc.edu

Víctor Moreno Borràs
victor.moreno.borras@estudiantat.upc.edu

Abstract

In this report, we present a study of the use of a Content Based Image Retrieval (CBIR) algorithm, characterizing the images by histograms. Content-based image retrieval [2] is the application of computer vision techniques to the image retrieval problem, that is, the problem of searching for digital images in large databases. Here we study the performance of this algorithm when descriptors are based on histograms. We evaluate the results obtained in the overall operation of the image search system based on quantitative measures such as the F-measure and the computational cost.

I. INTRODUCTION

The main objective of this report is to present the results of a practical assignment aimed at developing a Content-Based Image Retrieval (CBIR) system. The system is designed to operate on a database containing 4,000 images of 500 different objects. The goal is to retrieve the 4 images that correspond to the same object as the input image provided to the system.

The practical assignment consisted of developing a CBIR system in MATLAB, using the histogram from the images. The system was tested on a set of query images, and the retrieved images were compared to the expected output to evaluate the system.

II. OVERALL SYSTEM DESCRIPTION

The system takes an input file, "input.txt", which contains the name or names of the images to be analyzed. For each image specified in the input file, the system does the following:

Loads the matrix **H**[1] and extracts the histogram **h**, from the current image. Calculates the distance vector **d**, between **h** and all histograms in **H**. Then, finds the 10 smallest values in the **d** vector and it stays with their respective indices to later be able to write the 10 image names to the output file, "output.txt".

Overall, the algorithm performs a similarity search for each image in the input file, by comparing its histogram with all histograms in the dataset and selecting the 10 images with the smallest distances. The selected images are then written to the output file.

Algorithm 1: CBIR System

```
Input: input.txt
Output: output.txt
load H from dataset
foreach img in input.txt do
    h = extractHistogram(img)
    for i ← 1 to size(H) do
        $d_i = distance(\mathbf{h}, \mathbf{H})$;
    end
    for i ← 1 to 10 do
        index = minIndex(d)
        d(index) = max(d)
        nameImg = searchDataSet(index)
        write (output.txt , nameImg)
    end
end
return output.txt
```

The proposed system offers some flexibility in the practice for two specific points: the number of bins in the histogram and the distance function.

Bins

By varying the number of bins in the histogram, the system can achieve a balance between the level of detail and the computational cost required for processing the images.

The first histogram below was generated using 16 bins, while the second one used 32 bins. By using a larger number of bins, we can capture more fine-grained details in the distribution of pixel intensity values, at the cost of increased computational complexity.

Figure 1: Image 'ukbench00000.jpg' from the dataset

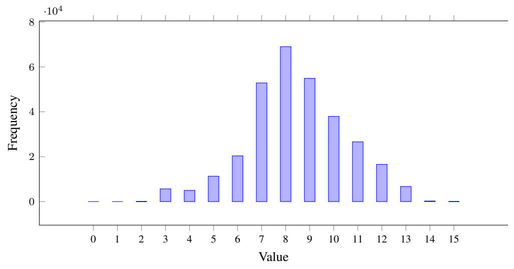

Figure 2: 16-bin histogram of image 'ukbench00000.jpg'

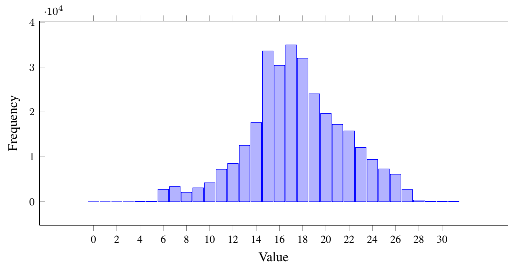

Figure 3: 32-bin histogram of image 'ukbench00000.jpg'

Distances

We have used three different distances to calculate the difference between histograms: the Mean Squared Error (MSE), the Bhattacharyya distance, and the Hellinger distance.

The MSE is defined as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(h_i - g_i)^2 \tag{1}$$

where h_i and g_i are the i-th bins of the two histograms being compared, and n is the number of bins.

The Chi-Squared distance[3] is defined as:

$$\chi^2(h,g) = \sum_{i=1}^{n}\frac{(h_i - g_i)^2}{h_i + g_i + \epsilon} \tag{2}$$

where h_i and g_i are again the i-th bins of the two histograms and ϵ is a small term to avoid division by zero.

Finally, the Bhattacharyya distance[1] is given by:

$$B(h,g) = -\ln\sum_{i=1}^{n}\sqrt{h_i g_i} \tag{3}$$

where h_i and g_i are the probabilities[2] of the i-th bin in each histogram.

III. RESULTS

Firstly, we will evaluate the system for the input file provided by the lab professor working with 256 bins. To do so, we will visualize the Recall-Precision graph and calculate the F-measure. This will allow us to compare the performance of the system when using three different distance measures.

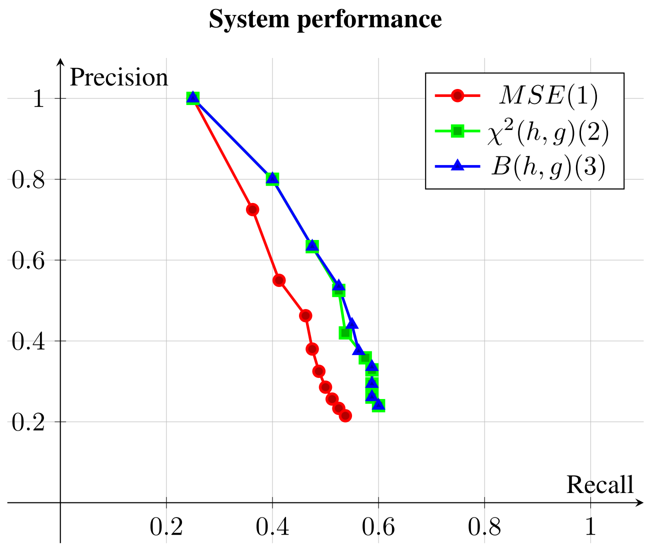

Figure 4: This precision-recall graph shows the performance of our system using three different distance metrics: Euclidean distance (red curve), Bhattacharyya distance (blue curve), and Chi-squared distance (green curve).

The performance curves for the Chi-squared and Bhattacharyya distances are similar, while the mean squared error (MSE) curve show a significantly worse performance.

This is due to the fact that the Chi-squared and Bhattacharyya distances both measure the similarity between two probability distributions, which is particularly useful in image retrieval applications when images are represented as histograms. In contrast, the MSE measures the mean square difference between two continuous variables and is less suitable for comparing histograms. As a result, the MSE show a poor performance compared to the other two distances,

	F-score	**Execution time**
MSE	0.4833	628.114 ms
Chi-squared $\chi^2(h, g)$	0.5429	179.137 ms
Bhattacharyya $B(h, g)$	0.5429	285.244 ms

Table 1: Results for 20 images of the input.txt

Note: execution time has been measured on our own computer..

From the results, we can see that the χ^2 and Bhattacharyya distances perform similarly with an F-score of 0.5429, which is slightly better than the F-score of MSE at 0.4833. However, it is important to note that the execution time for MSE is significantly higher than the other two metrics. This suggests that if execution time is an important consideration, then one of the other two metrics may be more suitable.

Now, we will evaluate the CBIR system using different bin levels for the histograms. By comparing the results obtained with different bin levels, we can assess the impact of the bin size on the performance of the system. This evaluation will help us determine the optimal bin level that balances the level of detail in the image representation and the computational cost required for processing the images.

Figure 5 shows the F-score as a function of the number of bins for three different distances, namely the mean squared error (MSE), the chi-squared (χ^2), and the Bhattacharyya distance (B). The x-axis represents the number of bins, ranging from 32 to 256, while the y-axis represents the F-score. The figure shows that for all three distances, the F-score increases as the number of bins increases, reaching a maximum at 224 and 256 bins. However, there are differences in the performance of the distances, with B outperforming the others for a lower number of bins (32-96) and χ^2 performing better for a higher number of bins (128-256).

[1]The **H** matrix is a collection of histograms extracted from the dataset.

[2]With the probabilities we mean the normalized histogram

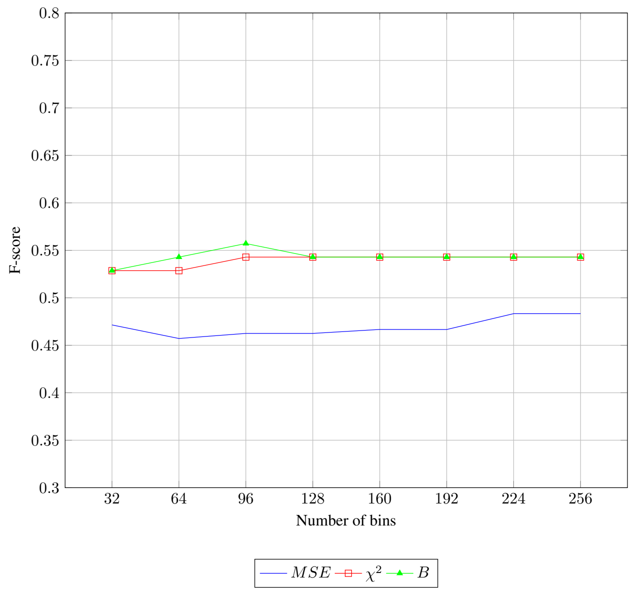

Figure 5: F-score as a function of the number of bins for three different distances.

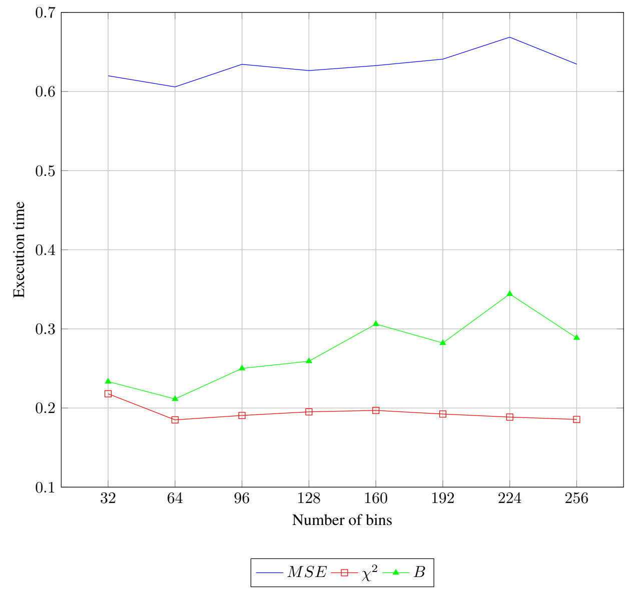

Figure 6: Execution time as a function of the number of bins for three different distances.

Figure 6 shows the execution time as a function of the number of bins for the same three distances. The y-axis in this case represents the execution time in seconds. The figure shows that the execution time increases very softly as the number of bins increases just for Bhattacharyya distance (3). In the other cases, the execution time remains constant against expectations.

IV. CONCLUSIONS

It is worth noting that the results presented here are based on a specific experiment, specifically the images of the input.txt, and may not be generalizable to other contexts. Therefore, the results should be interpreted with caution and further experimentation with others input.txt may be needed to confirm the conclusions.

The optimal configuration for our system without taking into account the runtime for this specific input file appears to be using the Battachary distance for a low number of bins. However, we believe that for other input.txt files, the performance may not be as high. In some way, we would be overfitting the system to always work with the same input data.

On the other hand, taking into account the runtime, we could opt for the chi-squared distance since it has slightly lower performance results than the Battachary distance but reduces the runtime by 62.8%. Therefore, in applications where the system's runtime is critical, the chi-squared distance is the better option. Finally, the option of working with the MSE would be totally discarded.

In conclusion, the optimal configuration for the system is based on a specific experiment and may not be generalizable to other contexts. Further experimentation with other input files may be necessary to confirm the conclusions.

Improvements

One possible improvement that could be implemented is the use of color histograms, as currently we are processing images in grayscale, resulting in a significant loss of information.

Another approach to tackle the problem would be implementing other algorithms instead of histograms, as they do not provide any spatial or geometric information.

References

[1] Authorless. Bhattacharyya distance. `https://en.wikipedia.org/wiki/Bhattacharyya_distance`.

[2] Authorless. Content based image retrieval. `https://en.wikipedia.org/wiki/Content-based_image_retrieval`.

[3] Xiaopan Chen Fengbin Zheng Wei Yang, Luhui Xu and Yang Liu. Chi-squared distance metric learning for histogram data, 2015.