Image Filtering and Object Detection

Dario Russo (1714011), Pooja Jambaladinni (1911273), Vedat Katirci (1908990), Dilara Isikli

October 2020

1 Exercise 1.d

An image (27x27) with only a central pixel as non-zero value can be used to find out impulse responses for different combinations of 1D Gaussian filter G_x and its 1D derivatives filter D_x.

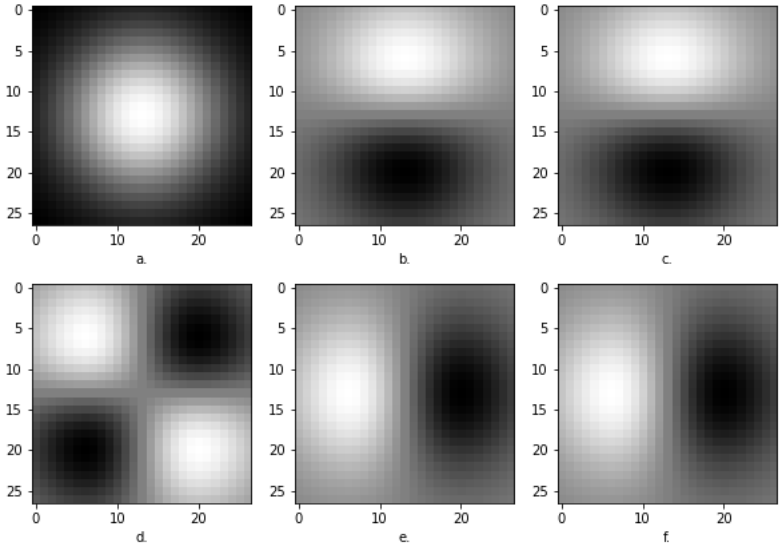

Figure 1: Image changes after filters

a. The filter blurs the image using an impulse response as a Gaussian. It represents a 2-Dimensional Gaussian kernel in zero derivative order, thanks to the linearity of this operation it is possible to break the 2D convolution in two 1D kernels, reducing the complexity.

b. It is possible to take advantage of the properties of linear systems, to smooth and find edges. It is possible to combine the first derivative and the Gaussian filter in one operation. In this way, it is possible to smooth with 1D Gaussian kernel in one direction and then apply all in once the filter and smoothing on the other one, finding in this case the horizontal edges.

c. In this example we obtain the same result that we got in 2., because we just inverted the order of the applied filters. This is possible, because we are working with only linear filters.

d. The filter blurs the image using an impulse response as a Gaussian. It represents a 2-Dimensional Gaussian kernel in dx dy. This is the same as a combination of a 2D Gaussian blur and a derivative filter applied in both directions. In this way it is possible to find edges in both directions (although it is not considered the best practice).

e. It is possible to combine the first derivative and the Gaussian filter in one operation. In this way, as the first step a smoothing combined with a first derivative filter is applied in vertical direction and then a 1D Gaussian blur is applied on the other one. These operations lead to the vertical edges.

f. In this example we obtain the same result that we got in 5., because we just inverted the order of the applied filters. This is possible, because we are working with only linear filters.

2 Exercise 1.e

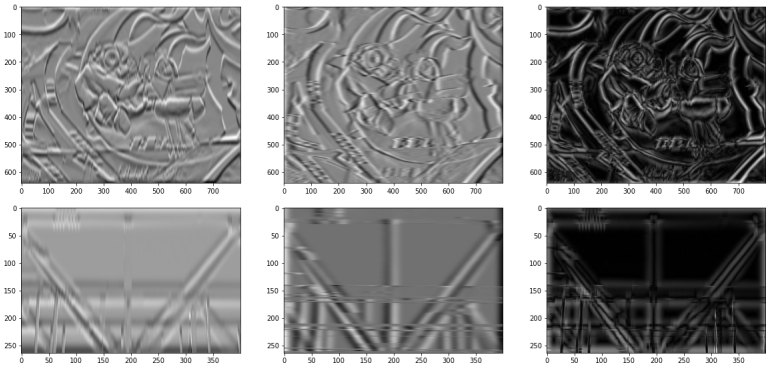

Figure 2: Examples

Considering the images graf.png and gantrycrane.png, applying the derivative of Gaussian filter in y-direction emphasized the edges which were along the y-direction (vertical edges) and applying the derivative of Gaussian filter in x direction emphasised the edges along the x-direction (orthogonal to y direction).

$$\frac{\partial}{\partial x} I(x,y) = I_x \cdot D_x$$
$$\frac{\partial}{\partial y} I(x,y) = I_x \cdot D_x$$

The third result for the images graf.png and gantrycrane.png shows the magnitude of the gradient (ie. the intensity of the vector in the direction orthogonal to the edge) which measures the edge strength that is calculated using the following formula : It is also important to notice that the image has been smoothened before applying the derivative filter. It is mainly because the first derivative operator while applying the derivative filter is affected by noise and also, the numerical derivatives can amplify noise (particularly higher order derivatives).

3 Exercise 3.c

With a simulation a sample made of 300 observations has been randomly generated. For randomly, we mean that the combination *histogram type - distance type - number of bins* is selected by random sample. For each iteration we work out the percentage of correct matches as score, that we are going to use to draw some conclusion about how these parameters affect the ability to properly identify the pictures.
As first step, we have decided to study how the different histograms and the dimensions affect the image recognition process, analyzing the average score.

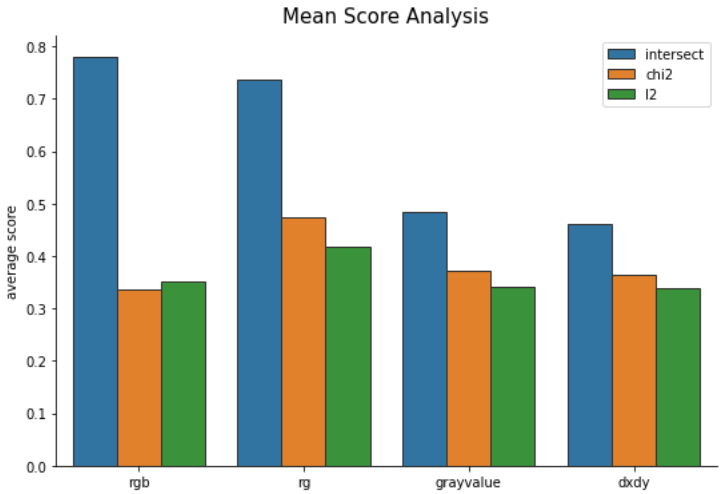

Figure 3: Mean Score vs Different Distances and Histograms

It is clear how important is the kind of histogram that is used for the identification. It is clear that the more information about the picture there are, the better the task is carried out. For this reason, the luminance histogram computed on the gray scaled image performs worse. Going on this way, it is clear that the histogram computed on a filtered image can only be worse, because with the Gaussian blur we have removed frequencies. However, this conclusion must not be drawn without thinking that removing high frequencies means getting rid off noise and in general it may improve a model. Furthermore, the filter does not only blur the image but it also makes the derivative of the function, which highlights the edges, unfortunately this operation doesn't help in image identification, because most of the times the AI is not able to interpret them well. Following the reasoning done up to this point, it is easy to understand why the RGB histogram and RG performs quiet better than the other ones.
For what concerns the distance types, we know that both euclidean distance and chi-squared distance are focused on the differences and that the latter is more discriminant because weights are not the same. This is the reason why chi-squared performs in average always better than the euclidean distance, except for the RGB histogram, in which they have almost the same performance, but probably it is due to the too small sample, maybe for a bigger number of observations the pattern would be followed. Finally, the intersection distance is the one that performs well because it is the best in terms of robustness.

As second step, we focus our attention on the bin size. and for the sake of completeness, also in this case we plot this information combined with the other ones. This time, we compare our variable directly with the score.

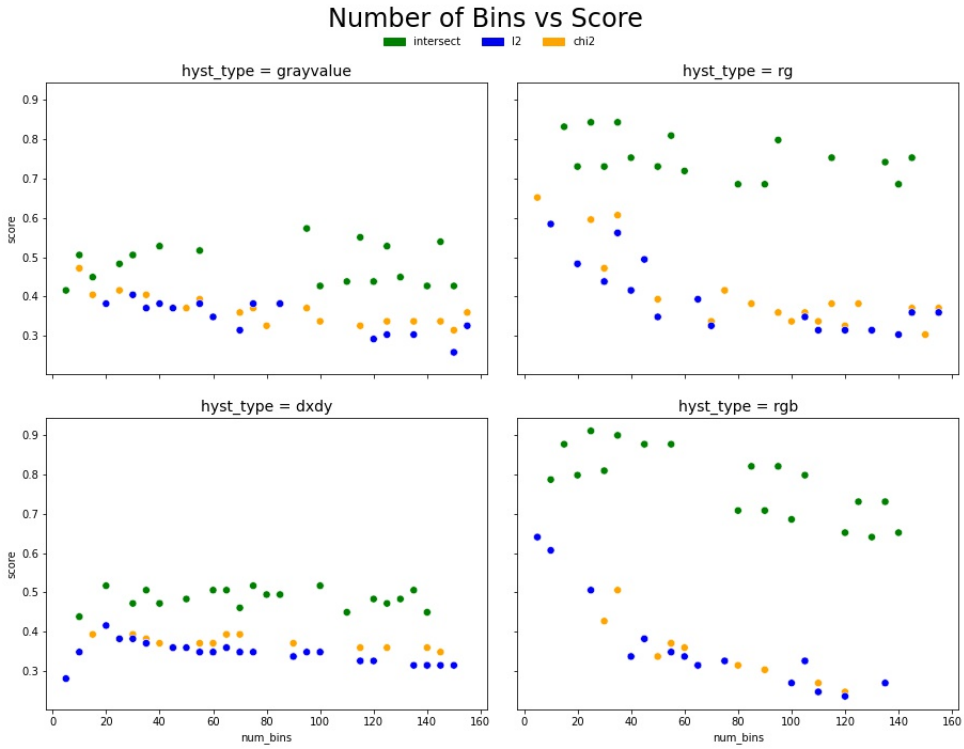

Figure 4: Score vs Number of Bins

For what concerns the gray value histogram and the dxdy histogram, it seems that the number of bins doesn't affect the performances of the identification process, which are in general very low. Instead, for the RGB and RG histograms, the number of bins strongly affects the final result. It is easy to see that with a number of bins higher than 20 the performances sink, it seems that this effect is smoothed when the intersection distance it is used, maybe because of its robustness. The idea behind this trend is clear, the more bins we have, the more details we have, but when they become too many, it becomes harder to find similarities and easier to find differences . There is a formula to work out the optimal number of bins, so we expect that that the score should have a max and then decrease, this kind of pattern can be appreciate in the bottom left subplot for the intersection function. The general idea is to find a number of bins that represents well our image, neither with a loss of information, neither with too much noise.
Thanks to a Bayesian Optimization Process we have found the best combination in terms of our score for the identification task. Anyway it is doesn't bring us to a very good result and this is because this kind of score doesn't take in account the False Positive. To obtain a better result is

recommended to use some other kind of score such as F1. In addition, it seems that this model has some troubles with the shapes, because it matches very well colours, but not shapes, check the Q2 result to see that.

Figure 5: Best Parameters Score

4 Exercise 4.b)

For calculating the precision and recall values we have used the formulation which is shown below:

Precission=TP/(Iteration+1)
Recall=TP/(Queries)

The intuition behind the precision is,fraction of retrieved images that are relevant to query images and recall intuition is the fraction of the relevant images that are retrieved. The steps for calculating precision and recall shown below:

$$D = \begin{bmatrix} 0.0022924 & \dots & \\ & \ddots & \\ & \dots & 0.00754249 \end{bmatrix} \quad L = \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix} \tag{1}$$

$$D = \begin{bmatrix} 0.0022924 & \dots & 0.00754249 \end{bmatrix} \; L = \begin{bmatrix} 1 & \dots & 1 \end{bmatrix} \tag{2}$$

$$d = \begin{bmatrix} sorted & smallest & to & highest \end{bmatrix} \; l = \begin{bmatrix} sorted & with & D.argsort() & indexes \end{bmatrix} \tag{3}$$

After matrix reshaped into 1-d, by using np.argsort() (which returns the index of the element smallest to biggest one) on the D matrix, we sort both D and L matrix with the same indexes that we get from np.argsort(). After this point our assumption is like that ;

For each iteration we increase the index and evaluate this index whether it is a True Positive or False Positive.All the remaining indexes which are not yet reached evaluated as True Negative.So as mentioned in the homework thresholds are equal to values in the d matrix. After this assumption calculating recall and precision are easy because we know (TP+FN) will be equal total number of positive which is equal to query size and TP+FP will be equal to the reached index. Implementation result shown below:

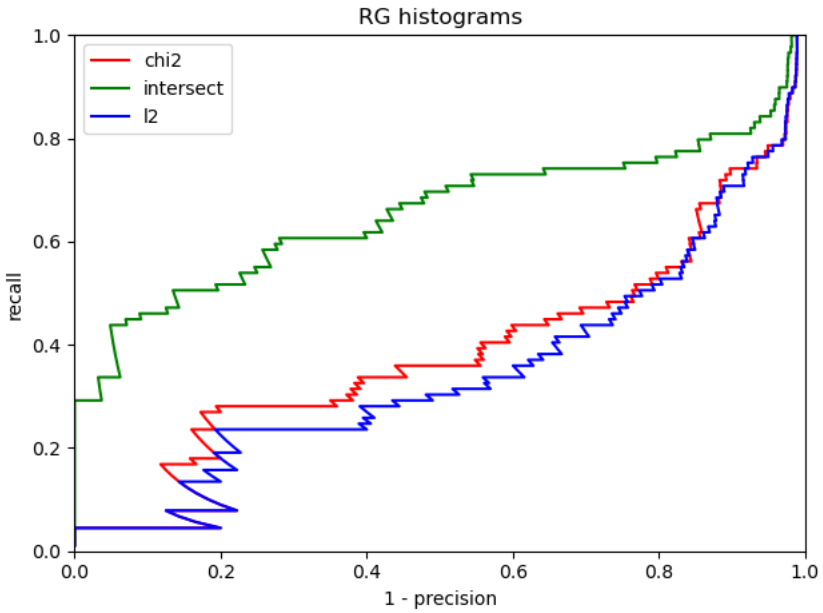

Figure 6: Mean Score vs Different Distances and Histograms

4-B) In this section we try to find the best combination according to F-scores. The combination that we try, can be seen below:

- Histogram types: RGB, RG ,DxDy
- Bin size : 5,7,10,12,15,17,20,25,30,35,40,45
- Distance : chi2, intersect, l2

Top 10 result according to F-scores is given below;

Combination	Recall	Precision	F-score
intersect-rgb-15	0.730337	0.833333	0.778443
intersect-rgb-12	0.719101	0.831169	0.771084
intersect-rgb-22	0.696629	0.861111	0.0.770186
intersect-rgb-17	0.651685	0.840580	0.734177
intersect-rgb-7	0.640449	0.838235	0.726115
intersect-rgb-5	0.629213	0.848485	0.722581
intersect-rgb-10	0.696629	0.738095	0.716763
intersect-rg-25	0.606742	0.870968	0.715232
intersect-rg-15	0.629213	0.823529	0.713376
intersect-rgb-20	0.606742	0.805970	0.692308

According to top result with respect F-scores, we have concluded that rgb-histogram method is more efficient than other 2 histogram methods which are rg-histogram and dxdy-histogram. Our intuition for that is RGB picture contains more information than RG picture. Moreover, dxdy-histogram after normalization detecting edges and loosing all the color information.That's why there is no dxdy-histogram value in top 10. Furthermore it seems like bins interval size is also has affect on the score.