Textons report - Lab4 - Computer Vision

Nicolas Florez, Yesith Juez

***Index Terms*—Textons, Nearest Neighbour Classifier , Random Forest Classifier, Bank filters**

I. Introduction

THERE are several ways to represent an image: through its color, texture or shape. The texture is very important because it allows to see the regular patterns that exist inside an image, and thus detect the edges that are inside it [1]. The repetitive elements inside an image are called textons, which are possible to find by means of the local texture representation using filters [2]. The present report it will be shown how it is possible to train and evaluate two classifiers based on the representation of textons and compare which classification method is more accurate.

II. Materials and Methods

A. Description of the Database

The database used for the realization of this laboratory consists of a set images with different types of textures. The database is divided into two parts: Training and testing.The set of training images are in total 750 images divided into 25 categories, and the set of test images are in total 250 images divided unto 25 categories.

B. Image Representation

The textons allow us to represent an image based on its texture. The textons classify the images by means of histograms, where each histogram indicates the vectors associated to the responses of the filters for each pixel of the image. When comparing the Euclidean distances between the test and train histograms, each image can be associated to a specific class.

The representation of a texton in an image shows how is the shape and orientation of the edges contained in it. The creation of the dictionary of textons was carried out the following procedure:

- A Gaussian-type bank filter was created.
- Due to the large number of images that were used for the training stage (750 images of 480x640), only one region of each image was taken, considering that the texture is uniformly distributed.
- The convolution of the image regions was carried out with each of the bank's filters.
- Then, each pixel of the image regions is represented by a vector formed from the responses of filter banks.
- The vectors are clustered using the K-means method. Each cluster center represents a texton in the dictionary.

Depending on the orientation and shape of the edges of an image, each filter can be more discriminative than another.

```
addpath('lib/matlab')
tic
%%Create a filter bank with deafult params
[fb] = fbCreate;
A=dir('train');
imBase=[];
vect={};
for j=3:numel(A)
D=dir(['train/' A(j).name '/*.jpg']);

for i=1:numel(D)
    D(i).name
    imtemp = double(imread(D(i).name))/255;
    imtemp = imtemp(1:150,1:150);
    imBase=[imBase imtemp];
    labelstrain{end+1}=A(j).name;

end
end
k = 16*8;

%Apply filterbank to sample image
filterResponses=fbRun(fb,imBase)

%Computer textons from filter
[map,textons] = computeTextons(filterResponses,k);
histograma=[];
A=dir('train');
for j=3:numel(A)
    D=dir(['train/' A(j).name '/*.jpg']);
    for i=1:numel(D)
        tmapBase1 = assignTextons(fbRun(fb,double(imread(D(i).name))/255),textons');
        histogramatrain=[histograma histc(tmapBase1(:),1:k)/numel(tmapBase1)]
    end
end
toc
save('hsitograma_train50.mat')
```

C. *Classifiers*

1) Nearest neighbor: It an supervised machine learning method, its largely used in the pattern recognition problem to classify objects based on an training of near elements in the space [3].

We implemented the classifier in MATLAB using the ”ficknn” function available in the software, for the usage we need to pass for parameters the matrix as the predictors variables, the labels for the supervised training and finally the amount of Neighbors the algorithm will use to classify the data.

```
tic
Mdl = fitcknn(histogramatrain',
    labelstrain,'NumNeighbors',3);
predict(Mdl,histogramatest')
toc
```

The ”tic” and ”toc” functions were used to calculate the time the algorithm will take for the learning process.

The Nearest neighbors classifier has a hyperparameter (number of nearest neighbors). The choice of this value can be very varied depending on the quantity and types of data to be classified. It should not be a very small value (risk of noise), nor very large (No differentiation between similar classes).

2) Random forest: It’s another supervised machine learning algorithm, the key point is that it to average multiple models to reduce the noise generated by the individual calculations [4]. For the implementation we used the ”TreeBagger” function of MATLAB. It receives as parameters: 1. The ammount of trees, 2. the matrix of the training points, 3. the supervised labels, and finally the desired method in this case Classification.

```
for i=1:100
tic
Mdl = TreeBagger(5,histogramatrain',
    labelstrain,'Method','
    classification')
predict(Mdl,histogramatest')
times(i)=toc;
end
mean(times)
```

The Random forest classifier has a hyperparameter (Number of trees). The choice of an appropriate value of this parameter depends on the data to be used. In general, the more decision trees are used, the classifier will be more accurate. It’s necessary to be careful not to overfit by using a lot of trees.

Before the representation of the images in textons, a data adjustment was made. Due to the number of images that were taken of training (750 images), and that each image was of big size (480x650), if all these images were used in their original form, the virtual machine did not have the sufficient storage to obtain the representation in textons. The solution proposed for this problem was to use only one region of the image (150x150). This is possible because of the uniformity of textures that were in the images.

III. Results

A. *Image Representation*

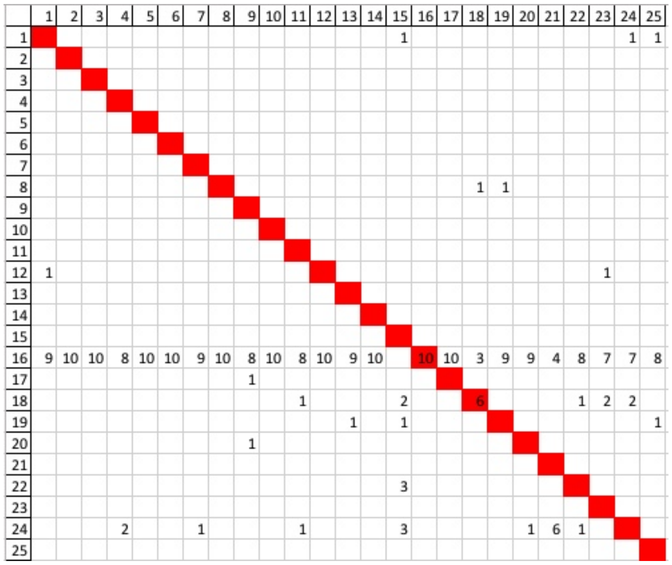
For the image representation we tried different sizes of the train images to create the textons. We croped the iamges to the initial portion of 50 by 50 pixels and 150 by 150 pixels. and with that the algorithm has different time respones. In average the time consumed by the computational cluster was 27 minutes for the 50 by 50 pixels train images(750) and in the second case (150 by 150 pixels) the program in average lasts 71 minutes.

B. *Nearest neighbor*

For the time of the algorithm predicting the 250 images of the database using the 750 images train model we found that the average time was:

AvgTime: 0.0485s

In the Results of the 150 by 150 pixels cropped-image the confusion matrix can be found in the Fig. 1 It can be shown that the predominant image predicted is from the category 16 that corresponds in the database to T16_glass1 it says that they could have the strongest features for the algorithm and the resulting predictor is arranging new images from the test database to the hyperspace corresponding to the glass.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	🟥														1									1	1
2		🟥																							
3			🟥																						
4				🟥																					
5					🟥																				
6						🟥																			
7							🟥																		
8								🟥										1	1						
9									🟥																
10										🟥															
11											🟥														
12	1											🟥											1		
13													🟥												
14														🟥											
15															🟥										
16	9	10	10	8	10	10	9	10	8	10	8	10	9	10		10	10	3	9	9	4	8	7	7	8
17									1								🟥								
18											1				2			6				1	2	2	
19													1		1				🟥						1
20									1											🟥					
21																					🟥				
22															3							🟥			
23																							🟥		
24				2			1				1				3					1	6	1		🟥	
25																									🟥

Fig. 1. Confusion matrix of the knn algorithm

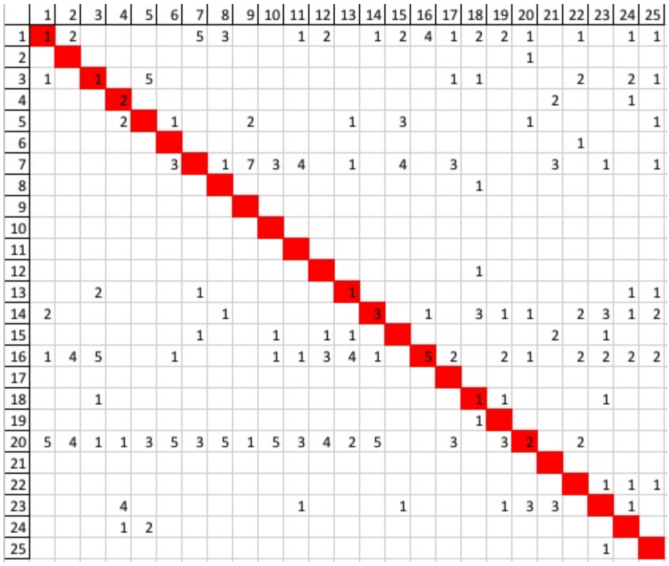
The previous statement and result tells us that the algorithm is not so robust to small cropped and low textured images as the done ones of the preprocessing step(crop the image to the initial 150 by 150).

C. *Random forest*

For the time of the algorithm predicting the 250 images of the database using the 750 images train model we found that the average time was:

AvgTime: 0.1773s

In the Results of the 150 by 150 pixels cropped-image the confusion matrix of the Random forest algorithm can be found in the Fig. 2 It can be shown in this time that there is not a predominant category for the images, this shows an overall better performance than from the Tree Bag Algorithm than the Nearest neighbor one, it is consistent with the expected output as this algorithm focuses on remove noise from the model by calculating different ones.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	1	2					5	3			1	2		1	2	4	1	2	2	1		1		1	1
2																				1					
3	1		1		5												1	1				2		2	1
4				2																	2			1	
5				2		1			2				1		3					1					1
6																						1			
7						3		1	7	3	4		1		4		3				3		1		1
8																		1							
9																									
10																									
11																									
12																		1							
13			2				1						1											1	1
14	2							1						3		1		3	1	1		2	3	1	2
15							1			1		1	1								2		1		
16	1	4	5			1				1	1	3	4	1		5	2		2	1		2	2	2	2
17																									
18			1															1	1				1		
19																		1							
20	5	4	1	1	3	5	3	5	1	5	3	4	2	5			3		3	2		2			
21																									
22																							1	1	1
23				4							1				1				1	3	3			1	
24				1	2																				
25																							1		

Fig. 2. Confusion matrix of the Random Forest algorithm

IV. THINGS TO CONSIDER

It's important to express that this laboratory could have better results as the iteration of multiple of hyper-parameters to get the best classification models also the change of the crop size beyond 150 by 150 pixels or the usage of a method of sub-sampling to reduce the size of the original image. We tried to do the sub-sampling of the original image to 250 by 250 pixels but the computational cluster died several times in the week that this laboratory was expected to be running. On Thursday 01/03/2018 the machine died for the last time when the algorithms had 2hours running calculating this iteration. We expect that the sub-samplig pre-processing method could provide us better results, but because of the situation was really impossible to do the expected iterations.

V. CONCLUSION

- The hyperparameters of the classification algorithms are a key factor that will determine the accuracy of the final calculated model for the predictors. Those parameters are sensible to the specific problem data.
- The textons dictionary are the most time and resources consuming algorithms, so it's necessary to determine a good database and preprocessing algorithms to optimize the computation time for the best results
- As expected the Random forest classification algorithm was the best one as stated in their confusion matrix, the algorithm must have removed several noise from the calculated models.
- It's important to have an excellent computational machine that could handle the textons dictionary calculation for real problems with tons of images the machine will determine the time that we can classify images.
- Cropping a texture image could enhance the time that the algorithms would run creating the textons dictionary but will affect directly the correct classifications showed in the confusion matrix of each machine learning algorithm.
- Machine learning algorithms can be used in the help of day to day problems as the organization of images by a classification.

VI. REFERENCES

[1] R. Szeliski, ”Computer Vision: Algorithms and Applications”, Springer, 2011.

[2] D. Forsyth, J. Ponce, ”Computer Vision: A Modern Approach”, Prentice Hall, 2011.

[3] D. Sontag, ”Nearest Neighbor Methods”, New York University.

[4] G. Louppe, ”Understanding Random Forests”, University of Liege, 2014.