Laboratory 7: Textons and Classifiers

Javier Coronel
Universidad de los Andes
Biomedical Engineering
jd.coronel30@uniandes.edu.co

Luis Carlos Rivera
Universidad de los Andes
Biomedical Engineering
lc.rivera10@uniandes.edu.co

Abstract

Textons were first defined as elements used by humans to make a linear representation of an specif texture. This initial definition was useful for computer vision specially for semantic segmentation, that's because texture gives additional information of an object. In the present laboratory will be presented two different methods for textons classification over the dataset provided by Ponce Research Group, the classifiers are Nearest Neighbor (NN) and Random Forest (RF). After a quantitative evaluation over the dataset the result shows that Nearest Neighbor represent the best alternative to classify textures, obtaining an Average Classification Accuracy of 68.2%.

1. Introduction

In the present work will be analyzed the performance in one of the main task of computer vision like is semantic segmentation. It consist in classify an image using as main resource the texture information that characterizes an object as a hole element in a picture. For that task, two different kind of classifiers are implemented: Nearest Neighbor and Random Forest. All the training and testing of the performance will be made over the Ponce dataset, and for evaluation of the result will be presented the confusion Matrix that compare each one of the categories and the average accuracy of the method [1][2].

2. Methodology

2.1. Database

For this laboratory the dataset used is from the Ponce Research Group of the Illinois University. This specific dataset is for texture analysis, counting with test and training data. For the training data there are a total of 750 images in JPG format with 25 different categories, each one with 30 images. For the testing data, a total number of images are 250 representing the same categories, but with a different number of images per category, this time are 10 images per each one.

For the creation of the dictionary of textons, first, we defined the filter bank, this was using the function $fbCreate$. This function creates a filter bank with two options parameters defined by the user or default parameters, for this specific laboratory we decided to use the default parameters of the function. The default parameters are represented in table 1.

Table 1. Default values filter bank.

Number of orientations	8
Starting Sigma	1
Number of Scales	2
Scaling Factor	$\sqrt{2}$
Length value	2

After the creation of the filter bank, the following step was the creation of a textons dictionary. For that 50 images were selected from the training data, 2 per each category, this two images were selected randomly from the dataset to avoid overfittig and memory issues. The most discriminative filters are the oriented filters (Figure 1), this because most of the images in the dataset are composed of oriented and long lines.

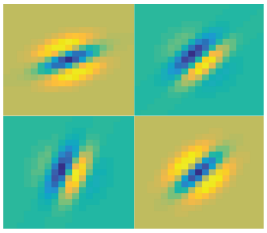

Figure 1. Most discriminative filters.

Following by that, the filter bank was applied over the 50 concatenated images using the function $fbRun$. From the output of the filters over the training images we applied the function $computeTextons$ where there will be applied k-means by a set number of clusters. In this case, was decided to try different number of clusters, first 50 and finally

90, the initial value its because we have 25 categories and we selected two different images we expected that the minimal number of textons to be used were 50 and the second value was selected bigger than the initial to identify how will change the performance of the semantic segmentation.

2.2. Nearest Neighbor

After the creation of the textons dictionary, a distribution of textons for each category was implemented using the function *assignTextons*. To make this, 20 images of each category in the training set were selected to obtain its textons distribution using the textons dictionary. Then, a normalized histogram of the generated map was achieved and was stored with the corresponding category label. This way, a dictionary of categories was obtained, in which the textons distribution for each category is listed. Finally, a matrix of 500x3 is obtained, 20 histogram distributions per category, first column with map of textons, second column with histogram distribution of textons and third column with corresponding label of the category.

For a new input image, it is assigned its textons and then, the normalized histogram of textons distribution is calculated. This histogram is necessary for comparing and define its category. For comparing histograms, *Histograms Intersection* and *Chi-Square* measures is implemented [1], the lower the measure, most similar are the histograms. Thus, the new histogram is compared with each histogram in the dictionary of categories, the category for the new image corresponds to the category of the most similar histogram.

For this Nearest Neighbor classifier, the most important parameters are:

- ***k:*** Number of textons in dictionary and number of bins for histograms, is critical for algorithm performance.
- ***Textons Dictionary:*** It is necessary to assign a new textons for a new image.
- ***Categories Dictionary:*** Contains the histogram distribution for each category, it is necessary for comparing new histograms and assign new labels.
- ***New image map:*** Generated image with new textons, necessary to calculate the histogram distribution and compare in Categories Dictionary.
- ***Distance Measures:*** Histograms Intersection and Chi-Square, necessary to identify the corresponding category for a new image.

The adjustment implemented for this classifier were the number of images used for training and the parameter k. In an early stage, only 10 images were selected from training set; increasing this number to 20 resulted in a best performance for the algorithm, but obviously resulted in more processing time. The number of textons also were modified, first was set to 50, but it was decided to increase until 90. With this number a better description of textons was obtained in the images, and was more easy to identify differences or similarities in histograms.

[1] *Histograms Intersection* and *Chi-Square* measures were obtained using the functions in *Histogram distances* by B. Schauerte, 2009.

Table 2. Time performance.

Method	Training Time (s)	Test Time (s)
Nearest Neighbor	521.9	220.8
Random Forest	1000	500

2.3. Random Forest

After the creation of the textons dictionary, the definition of the random forest will give an output of a prediction model. For the creation of this model there are needed two inputs; the first one is the training data, for creating the input we applied all over the 750 training images the function $assigntextons$, for each one of the textons map obtained as the function was applied we create a texton histogram and concatenated with the respectably label (1-25), each one represent a semantic category but for the present report we refer with a number to a category. As final result, a matrix of 750x51 is obtained, the last column is the label class of each histogram. For the creation of the model we used the Matlab function $TreeBagger$ which receive as parameters number of trees, training data and labels class. The relevant parameters for this method were:

- ***k:*** The number of textons in the dictionary, we tested the performance in $k = 50$ and $k = 90$ to decide how will affect the performance of the method.
- ***Texton Dictionary:*** It's the crucial factor to determinate in how many ways a texture could be represented.
- ***Texton Histogram:*** It's the way of representing the result of $assignTexton$, this info will be used to create the prediction model over the training dataset. The function used to represent the texton as an histogram was using Matlab function $histcounts(textons, numberofbins(K))$.
- ***Number of trees:*** We decided that the best parameter was setting the same number of tress as the number of categories, this will lead to the creation of a random forest for each of the data in the training.
- ***Prediction Model:*** This model was training by using the histogram representation of the textons map for each of the categories in the training dataset, its is important to remark that the only adjustment made over

the process is a normalization over the gray level over all the images creating images that were between 0-1 in each pixel value.

- ***Predicted Label:*** This is the output of any new data that pass trough the prediction model, in this report we treated each category as a number between 1-25.

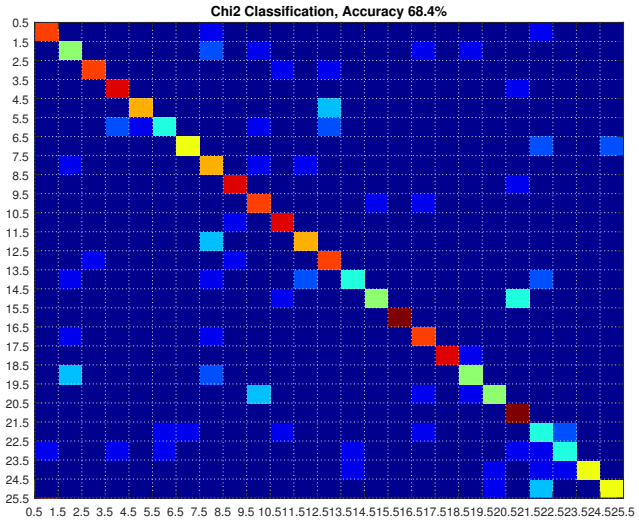

Figure 2. Confusion Matrix for Chi-Square measure in Nearest Neighbor.

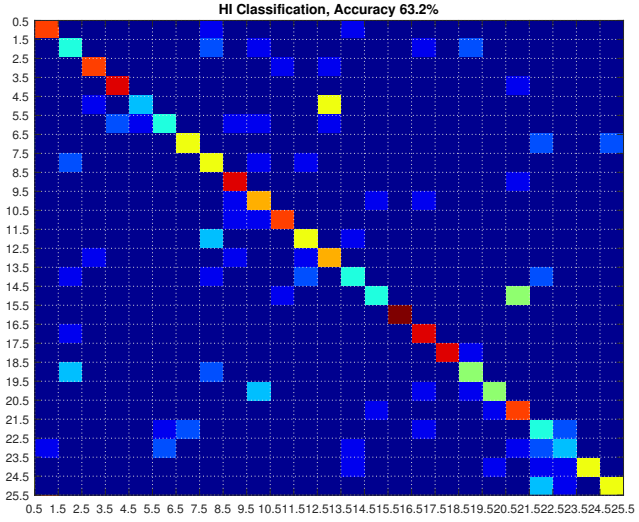

Figure 3. Confusion Matrix for Histogram Intersection in Nearest Neighbor.

3. Results

After implementing the exposed methods on the training set, the performance on the test set was measured using a confusion matrix and the average classification accuracy (ACA). For the Nearest Neighbor, the confusion matrix is represented in Figures 2, 3. The ACA for the Nearest Neighbor classification using Chi-Square metric is 0.683 and for Histogram Intersection metrics is 0.632.

Using the Random Forest classifier, the confusion matrix is presented in Figure 4. The ACA for this method is 0.548.

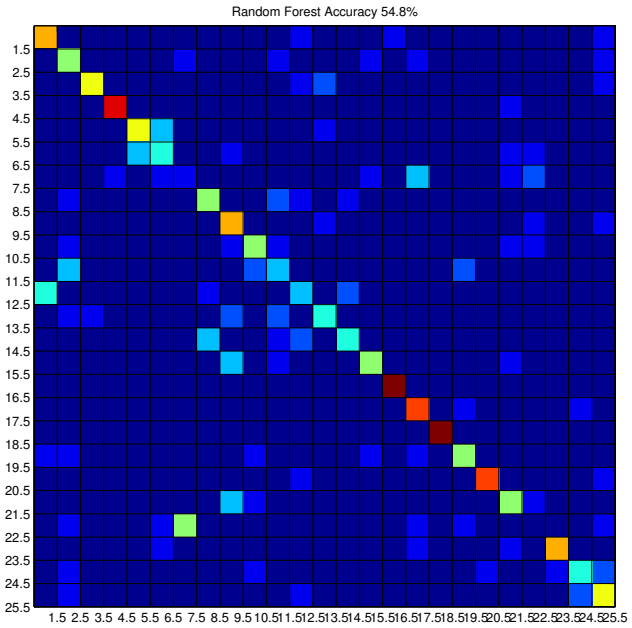

Figure 4. Confusion Matrix Random Forest Method.

4. Discussion

First of all, the parameter k implemented in the classifiers were critical to the classification performance, by increasing this parameter it was noticeable a best classification. This is because the more number of textons, the better the description of the image, and for the case of Nearest Neighbor classifier, the histograms had more robust information to compare and the distribution was more different between categories.

The necessary time to train and execute the algorithms are presented in Table 2. A mean of 0.88 seconds it took for the Nearest Neighbor method to classify a new image and assign its category in comparison with RF that took 2 seconds per image, more than two times the required time for NN. It is important to stand out the training time, for NN the training time is about 2 times faster than RF. For the ACA,

Table 3. Performance by category in Nearest Neighbor Classifier with Chi-square.

Category	Precision	Recall	Accuracy
1	0.89	0.80	0.99
2	0.45	0.50	0.96
3	0.89	0.80	0.99
4	0.75	0.90	0.98
5	0.87	0.70	0.98
6	0.67	0.40	0.97
7	0.86	0.60	0.98
8	0.41	0.70	0.95
9	0.82	0.90	0.97
10	0.57	0.80	0.86
11	0.75	0.90	0.97
12	0.7	0.70	0.97
13	0.57	0.80	0.96
14	0.67	0.40	0.96
15	0.83	0.50	0.97
16	**1**	**1**	**1**
17	0.67	0.80	0.96
18	**1**	**0.99**	**1**
19	0.62	0.50	0.96
20	0.71	0.50	0.99
21	0.71	1	0.96
22	**0.4**	**0.28**	**0.94**
23	0.57	0.40	0.98
24	1	0.60	0.96
25	0.75	0.60	0.95

Table 4. Performance by category in Random Forest.

Category	Precision	Recall	Accuracy
1	0.58	0.70	0.97
2	0.33	0.50	0.94
3	0.85	0.60	0.98
4	0.90	0.90	0.99
5	0.67	0.60	0.97
6	0.40	0.40	0.95
7	0.15	0.10	0.94
8	0.55	0.50	0.96
9	0.41	0.70	0.95
10	0.50	0.50	0.96
11	0.27	0.30	0.94
12	0.30	0.30	0.94
13	0.50	0.40	0.96
14	0.57	0.40	0.96
15	0.62	0.50	0.97
16	0.91	1	0.99
17	0.53	0.80	0.96
18	**1**	**1**	**1**
19	0.55	0.50	0.96
20	0.89	0.80	0.99
21	0.45	0.50	0.96
22	**0**	**0**	**0.94**
23	0.87	0.70	0.98
24	0.57	0.40	0.96
25	0.43	0.60	0.95

Nearest Neighbor with Chi-Square measure represented an improvement of 13.6% respect to Random Forest classifier. Because of the previous reasons, the best classifier is the Nearest Neighbor using Chi-Square measure(X^2).

To identify which categories cause the most confusion it is necessary to look at Figures 2, 3 and 4. First, there is evident that for both measures (Chi2 and HI), categories 13-5 and 21-15 represent the most bright values on Figures 2 and 3 in the upper triangle section. For RF classifier the most confused categories are 12-1 and 22-7, these are represented in the most bright values on the lower triangle in Figure 4.

The confusions obtained are because similarities in categories between representation space of the methods. For example in Nearest Neighbor classifier, for categories 15 and 22 (Fig. 5), the confusion may be explained not only by similar shapes in the image, but also because the distribution of textons on its histograms are similar. In Figure 5 are noticeable that four peaks (nearly of the same magnitude) are represented in both histograms despite its different category.

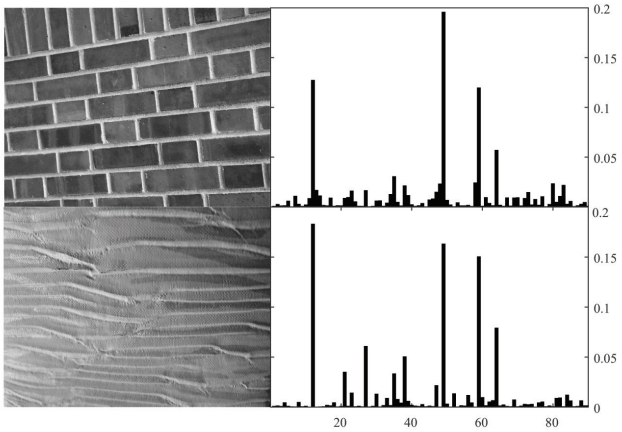

Figure 5. Comparison between confused categories, left: categories 15 and 21. right: Histograms of textons distribution.

4.1. Nearest Neighbor

In table 3 are represented Precision, Recall and average accuracy of each one of the categories for Nearest Neighbor Classifier using Chi-Square measure. In red is highlighted the worse classified category, and in green are the categories classified with the more efficient score, this indicates that

categories 16 and 18 are the most discriminative categories for this classifier and the best performance is obtained.

4.2. Random Forest

In table 4 are presented Precision, Recall and average accuracy of each one of the categories for Random Forest Classifier. Similar to table 3 for NN Classifier, the best performance was in category 18 , and the category with worst performance by this method was category 22.

One of the limitations of random forest is the computational resources that it consume to create the prediction model, making difficult to work with datasets that contain more that 25 categories will lead to maybe memory issues or take long time to make the model.
One of the possible improvements that could be made to this methodology is changing the correlation between the trees in the forest to identify which factor of correlation work better to create a segmentation over this specific dataset, that's because in this report the correlation factor is the one by default in the Matlab function.

The limitations of the database is the restricted number of patterns, this not represent the the most common textures in real life. Other limitation is the color of the images, it is true that a gray-level image is more easy to analyze, but reducing a multi-color image to gray-level also remove special features that might have crucial information for classifying.

5. Future Work

As a future work could be relevant to analyze how each of the classifiers behave when more textons are implemented, this to identify the critical point where the prediction model present overfitting and underfitting. Was evident that the number of textons k was important for classification; it would be interesting to inspect how the ACA an the training/testing times varies depending on this parameter and if there is a saturation in the classification performance. Also could be relevant the implementation of another classifiers like SVM to compare the performance of different machine learning techniques over texture information.

References

[1] B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290(5802):91–97, 1981.

[2] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Computer Vision–ECCV 2006*, pages 1–15. Springer, 2006.